UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE ECONOMÍA Y EMPRESA
CARRERA DE NEGOCIOS INTERNACIONALES

TEMA

Análisis de supervivencia de la producción del camarón segmentado por genética y alimentación.

AUTORES
Acán Guamán, Luis Slayder
Franco Vásquez, Carlos Antonio

Trabajo de titulación previo a la obtención del título de Licenciados en Negocios Internacionales

TUTOR
Ing. Carrera Buri, Félix Miguel, Mgs.

Guayaquil, Ecuador
13 de febrero del 2026

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE ECONOMÍA Y EMPRESA
CARRERA DE NEGOCIOS INTERNACIONALES

CERTIFICACIÓN

Certificamos que el presente trabajo de titulación, fue realizado en su totalidad por **Acán Guamán, Luis Slayder** y **Franco Vásquez, Carlos Antonio** como requerimiento para la obtención del título de **Licenciados en Negocios Internacionales**.

TUTOR

f. _____

Ing. Carrera Buri, Félix Miguel, Mgs

DIRECTOR DE LA CARRERA

f._____

Ing. Hurtado Cevallos, Gabriela Elizabeth, Mgs

Guayaquil, a los 13 del mes de febrero del año 2026

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE ECONOMÍA Y EMPRESA
CARRERA DE NEGOCIOS INTERNACIONALES

DECLARACIÓN DE RESPONSABILIDAD

Nosotros**, Acán Guamán, Luis Slayder**
Franco Vásquez, Carlos Antonio

DECLARAMOS QUE:

El Trabajo de Titulación, **Análisis de supervivencia de la producción del camarón segmentado por genética y alimentación**, previo a la obtención del título de Licenciados en Negocios Internacionales, ha sido desarrollado respetando derechos intelectuales de terceros conforme las citas que constan en el documento, cuyas fuentes se incorporan en las referencias o bibliografías. Consecuentemente este trabajo es de mi total autoría.

En virtud de esta declaración, me responsabilizo del contenido, veracidad y alcance del Trabajo de Titulación referido.

Guayaquil, a los 13 del mes de febrero del año 2026

LOS AUTORES:

f._____
Acán Guamán, Luis Slayder

f._____
Franco Vásquez, Carlos Antonio

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE ECONOMÍA Y EMPRESA
CARRERA DE NEGOCIOS INTERNACIONALES

AUTORIZACIÓN

Nosotros**, Acán Guamán, Luis Slayder**
Franco Vásquez, Carlos Antonio

Autorizo a la Universidad Católica de Santiago de Guayaquil a la **publicación** en la biblioteca de la institución del Trabajo de Titulación, **Análisis de supervivencia de la producción del camarón segmentado por genética y alimentación.**, cuyo contenido, ideas y criterios son de mi exclusiva responsabilidad y total autoría.

Guayaquil, a los 13 del mes de febrero del año 2026

LOS AUTORES:

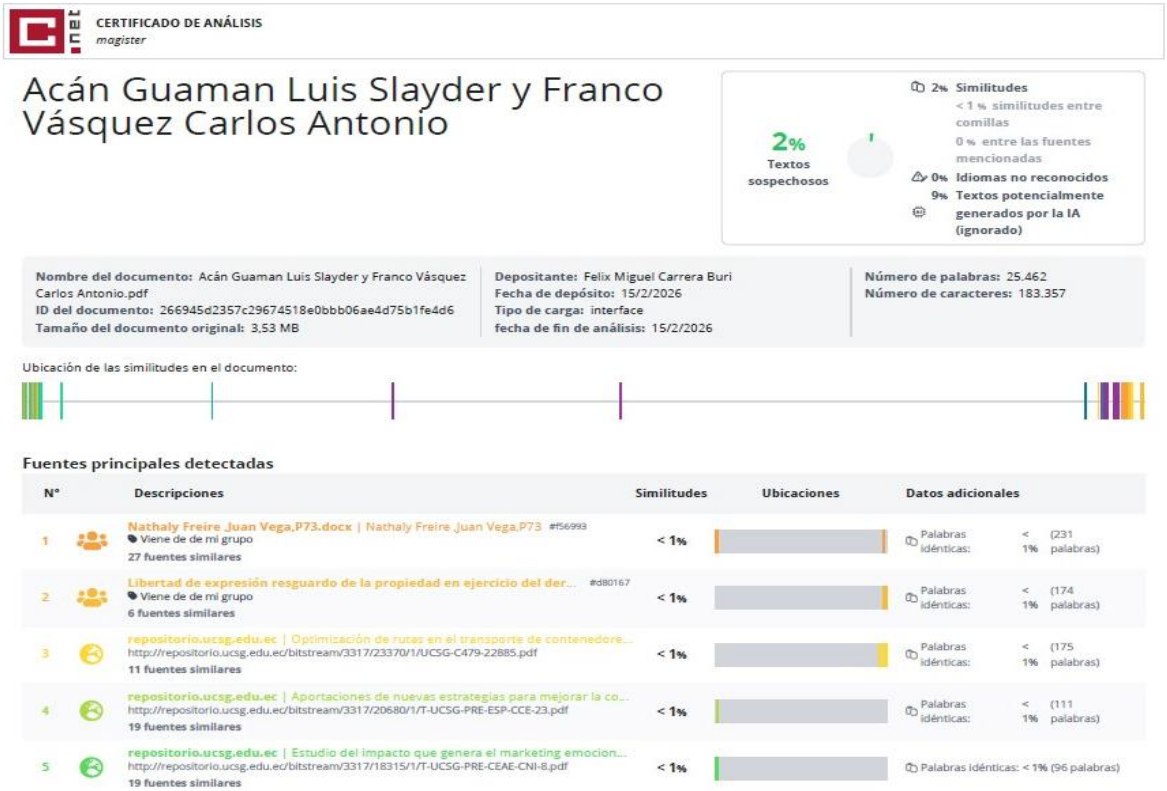
f._____
Acán Guamán, Luis Slayder

f._____
Franco Vásquez, Carlos Antonio

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE ECONOMÍA Y EMPRESA
CARRERA DE NEGOCIOS INTERNACIONALES

REPORTE COMPILATIO



CERTIFICADO DE ANÁLISIS
magister

Acán Guaman Luis Slayder y Franco Vásquez Carlos Antonio

2% Textos sospechosos

2% Similitudes
< 1 % similitudes entre comillas
0 % entre las fuentes mencionadas
0% Idiomas no reconocidos
9% Textos potencialmente generados por la IA (ignorado)

Nombre del documento: Acán Guaman Luis Slayder y Franco Vásquez Carlos Antonio.pdf
ID del documento: 266945d2357c29674518e0bbb06ae4d75b1fe4d6
Tamaño del documento original: 3,53 MB

Depositante: Felix Miguel Carrera Buri
Fecha de depósito: 15/2/2026
Tipo de carga: interface
fecha de fin de análisis: 15/2/2026

Número de palabras: 25.462
Número de caracteres: 183.357

Ubicación de las similitudes en el documento:

Fuentes principales detectadas

N°	Descripciones	Similitudes	Ubicaciones	Datos adicionales
1	Nathaly Freire Juan Vega,P73.docx | Nathaly Freire Juan Vega,P73 #f56993 Viene de de mi grupo 27 fuentes similares	< 1%		Palabras idénticas: < 1% (231 palabras)
2	Libertad de expresión resguardo de la propiedad en ejercicio del der... #d80167 Viene de de mi grupo 6 fuentes similares	< 1%		Palabras idénticas: < 1% (174 palabras)
3	repositorio.ucsg.edu.ec | Optimización de rutas en el transporte de contenedore... http://repositorio.ucsg.edu.ec/bitstream/3317/23370/1/UCSG-C479-22885.pdf 11 fuentes similares	< 1%		Palabras idénticas: < 1% (175 palabras)
4	repositorio.ucsg.edu.ec | Aportaciones de nuevas estrategias para mejorar la co... http://repositorio.ucsg.edu.ec/bitstream/3317/20680/1/T-UCSG-PRE-ESP-CCE-23.pdf 19 fuentes similares	< 1%		Palabras idénticas: < 1% (111 palabras)
5	repositorio.ucsg.edu.ec | Estudio del impacto que genera el marketing emocion... http://repositorio.ucsg.edu.ec/bitstream/3317/18315/1/T-UCSG-PRE-CEAE-CNI-8.pdf 19 fuentes similares	< 1%		Palabras idénticas: < 1% (96 palabras)

f. _____

Ing. Carrera Buri, Félix Miguel, Mgs
TUTOR

f._____

Ing. Hurtado Cevallos, Gabriela Elizabeth, Mgs
DIRECTOR DE LA CARRERA

AGRADECIMIENTO

LUIS SLAYDER ACÁN GUAMAN

Hoy, al concluir este último proyecto durante mi vida universitaria, deseo agradecer a Dios por haberme situado junto a las personas quienes me acompañan hoy en día y por ser un soporte en cada una de las decisiones que han ayudado a desarrollarme como estudiante, profesional, hijo y amigo.

Asimismo, agradezco a mi familia, mi abuelo por enseñarme que el trabajo y la constancia provocan que las cosas sucedan, ya que de las dificultades resultan lo mejor que nos ofrece la vida. Por aquellos instantes durante los cuales nos encontrábamos charlando al lado de una fogata, compartiendo anécdotas.

A mi madre, quien me enseñó a aceptar cualquier desafío que se presente en la vida, por las veces en que hemos discutido, jugado y reído juntos, por demostrarme que el campo es un lugar adecuado para despejar y aclarar las ideas.

Agradezco a mi compañero de tesis, con quien hemos cooperado mano a mano en la elaboración de este trabajo, por escuchar las sugerencias y las propuestas, por los momentos desafiantes y malas noches, por las veces que hemos debatido, meditado, agoviado y solventado.

DEDICATORIA

LUIS SLAYDER ACÁN GUAMAN

Este largo proceso se lo dedico a mis viejos, a las personas que llegue a conocer; grandes colegas, amigos y hermanos de vida, que se han cruzado en mi camino y me han brindado su apoyo y conocimientos en todos los aspectos de mi vida.

AGRADECIMIENTO

CARLOS ANTONIO FRANCO VASQUEZ

Quiero agradecerle antes que nada a mis padres, Carlos y Anabel; ya que por ellos soy y sin ellos nada de esto podría existir, agradecerles siempre por todo su apoyo en la universidad, en el colegio y en la vida en sí ya que es gracias a ellos que hoy en día sigo cumpliendo mis metas y forjando mi propio camino.

Agradecerlo así mismo a mis hermanas Emily y Mikelle quienes han estado para mí toda mi vida y han sido un pilar fundamental en mi formación personal.

Agradecerle a mi novia, María Fernanda “la tercera autora de esta tesis”, quién ha sido mi soporte en todo el proceso de titulación y durante toda la elaboración de mi tesis, por siempre estar conmigo, ayudarme a relajarme y a reír cuando necesitaba, por sus palabras de apoyo y cariño incondicional

Agradecerle también a mi compañero de tesis, Luis; con el que hemos sufrido de incontables horas de esfuerzo hasta poder concluir con nuestro trabajo y a mis grandes colegas y amigos de la carrera, en especial a Joselyne, “Gamboa”, Jeremy, Jennifer.

Finalmente, a mi tutor Félix, quien siempre ha estado dispuesto a ayudarnos y ayudarnos en ese camino y a todos los profesores que fueron parte de mi educación en la universidad y por su conocimiento compartido.

DEDICATORIA

CARLOS ANTONIO FRANCO VASQUEZ

Me gustaría dedicarles este título a las personas que más amo, a mis padres Carlos y Anabel, a mis hermanas Emily y Mikelle, a mi abuelito Ramiro que me cuida desde el cielo, a mí abuelita Nancy y a mí novia María Fernanda, ustedes tranquilos que yo salgo, Carlos

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE ECONOMÍA Y EMPRESA
CARRERA DE NEGOCIOS INTERNACIONALES

TRIBUNAL DE SUSTENTACIÓN

f. _____

Ing. Carrera Buri, Félix Miguel, Mgs

TUTOR

f._____

Ing. Hurtado Cevallos, Gabriela Elizabeth, Mgs

DIRECTOR DE LA CARRERA

f._____

Lic. Freire Quintero, César Enrique

COORDINADOR DEL ÁREA O DOCENTE DE LA CARRERA

INDICE GENERAL

INDICE FIGURAS

RESUMEN

La investigación, denominada "Análisis de supervivencia de camarón segmentado por genética y alimentación", busca determinar la probabilidad de supervivencia del camarón en el sector productivo ecuatoriano según variables productivas como la genética y el tipo de alimento. Esta investigación surge de la necesidad de crear una herramienta que permita establecer cuáles son los mejores proveedores que, mediante una base histórica, logran que las líneas genéticas alcancen cierto peso con determinado tipo de alimentación, además de identificar cuánto sería su tiempo óptimo de cosecha.

Metodológicamente, el estudio se abordó desde un enfoque cuantitativo, de modelado supervisado, específicamente de regresión logística, para predecir la probabilidad de supervivencia del camarón a partir de datos históricos de producción. Para verificar los resultados, se utilizaron métricas como la matriz de confusión, accuracy, precisión, recall, F1 score, curva ROC y AUC, las cuales midieron el rendimiento del modelo.

Además, se incorporaron indicadores productivos que refuerzan el análisis técnico.

Los resultados indican que la supervivencia no está ligada a las formas de alimentación, sino a la combinación de la genética y el proveedor que las desarrolla. ¡Este análisis hace evidente la necesidad de certificar tanto a los proveedores como a las genéticas con las que se ha trabajado.

Palabras Clave: Códigos genéticos, Supervivencia, Regresión, clasificación, logística, genética, producción, alimentación.

ABSTRACT

The research conducted, entitled “Survival Analysis of Shrimp Segmented by Genetics and Feeding,” aims to analyze the probability of shrimp survival in the Ecuadorian productive sector by examining key production variables such as genetics and type of feeding. This research arises from the need to develop a tool that makes it possible to determine the best suppliers who, based on historical data, enable specific genetic lines to reach a certain weight under a defined feeding scheme, as well as to identify their optimal harvest time.

Methodologically, the study focused on a quantitative approach based on supervised models, specifically logistic regression, in order to estimate the probability of shrimp survival using historical production data. To validate the results, performance metrics such as the confusion matrix, accuracy, precision, recall, F1 score, as well as the ROC curve and AUC were used, allowing the model’s performance to be properly evaluated.

Additionally, production indicators were integrated to complement the technical analysis.

The findings show that survival does not depend exclusively on feeding techniques, but rather on the combination of genetics and the supplier responsible for developing those genetic lines. This analysis highlights the need to evaluate and classify both suppliers and the genetic lines that have been used.

Keywords: Genetic codes, Survival, Regression, classification, logistics, genetics, production, feeding.

RESUMÉ

La étude intitulée « Analyse de la survie de la crevette segmentée par génétique et alimentation » vise à étudier la probabilité de survie de la crevette dans le secteur productif équatorien en analysant des facteurs productives clés comme la génétique et le type d'alimentation. Cette étude découle de la nécessité d'identifier les meilleurs fournisseurs qui, sur la base de données historiques, permettent à certaines lignées génétiques d'atteindre un poids donné sous un régime alimentaire donné et de définir leur temps de récolte optimal.

L'étude a utilisé une méthode quantitative basée sur des modèles supervisés, notamment la régression logistique, pour estimer la probabilité de survie des crevettes à partir de données historiques de production. Des indicateurs de performance comme la matrice de confusion, l'accuracy, la précision, le recall, le score F1, la courbe ROC et l'AUC ont été utilisés pour valider les résultats.

Par ailleurs, des indicateurs productifs ont été ajoutés pour compléter l'analyse technique.

Les résultats indiquent que la survie ne dépend pas uniquement des pratiques alimentaires, mais plutôt de la combinaison entre la génétique et le producteur de ces lignées génétiques, ce mois-ci alyse met en évidence la nécessité d'évaluer et de classifier à la fois les fournisseurs et les lignées génétiques utilisées.

Mots-clés : Codes génétiques, survie, régression, classification, logistique, génétique, production, alimentation.

1. INTRODUCCIÓN

Actualmente la industria camaronera a nivel global se encuentra como uno de los sectores con mayor auge dentro del sector acuícola, genera miles de millones de dólares en exportaciones y se constituye como la principal fuente de proteínas crustáceas aptas para el consumo humano según indica la publicación de (Fortune Business Insights, 2026) los productores han elevado los estándares de calidad para poder acoplarse a la exigencia del mercado.

El mercado camaronero al igual que los diversos mercados que forman parte del sector alimenticio enfocados en la producción de origen marítimo han conseguido establecer que la producción de origen acuícola sea mucho más amplia en relación a la extracción y pesca tradicional representando el 55% del total de los alimentos consumidos según reportes de la Organización de las Naciones Unidas para la Alimentación y la Agricultura. (Food and Agriculture Organization of the United Nations, 2024), sin embargo, este crecimiento trae consigo riesgos y oportunidades, tanto que mientras los avances tecnológicos en la optimización de la producción, los productores deben adaptarse a la volatilidad del mercado, los cambios en la calidad de insumos y riesgos fitosanitarios (Aqua Culture Asia Pacific, 2023)

En Ecuador esta industria constituye uno de los pilares económicos más relevantes y a su vez es uno de los referentes internacionales dentro del sector acuícola. El crecimiento ha sido progresivo; responde a un proceso histórico de resiliencia, adaptación e innovación constante. Comprender la evolución de la producción camaronera permite dimensionar tanto el crecimiento económico como el desarrollo de tecnologías para presentar a Ecuador como uno de los más establecidos a nivel mundial y su incidencia en el desarrollo económico global.

Ecuador comenzó las exportaciones de camarón desde 1968, cuando se dieron los primeros experimentos de producción en estanques artesanales en zonas costeras (manglares), durante 58 años, la industria comenzó a desarrollar sistemas de producción estandarizados, como lo son las piscinas de camarón (Tabares, 2023) en las cuales se desarrollaban las especies conocidas como camarones del pacifico, la ubicación del Ecuador en conjunto con el clima favorable, permiten que las camaroneras puedan cultivar tres veces al año siendo este una gran ventaja en comparación a otros productores del medio, también debemos tener en cuenta que estas piscinas se han desarrollado tanto en agua salada y dulce. En este período nacieron los primeros centros de investigación y se estableció un modelo extensivo de cultivo basado en procesos naturales, lo que impulsó los volúmenes de producción sin un aumento masivo en los costos de producción debido a que no se requería una tecnificación intensiva.

En el año de 1980 se presentó uno de los acontecimientos más importantes, el desarrollo de acelerado del sector Acuícola posicionando al país como uno de los principales exportadores de camarón a nivel mundial. El aumento de la inversión extranjera, Las mejoras dentro de la cadena logística y el interés internacional por productos del mar de excelente calidad contribuyeron a una expansión acelerada. Sin embargo, este punto de apogeo no estuvo libre de desafíos.

A finales de los años 90 y principios de los 2000, el sector enfrento una de sus crisis más severas debido al virus de la mancha blanca debido a su capacidad de contagio, Miles de hectáreas quedaron inactivas, y muchas empresas salieron del mercado. La industria pudo haber llegado a decrecer. (Piedrahita, 2018)

La industria camaronera ecuatoriana ha pasado de ser un proyecto experimental en los años sesenta a convertirse en un referente mundial de producción acuícola sostenible. Si en sus inicios el éxito del cultivo dependía casi exclusivamente de factores ambientales, clima,

disponibilidad de manglar y abundancia natural de larvas, en la actualidad su competitividad se sustenta en avances científicos, particularmente en la selección y mejoramiento genético del camarón. Con esta evolución se transformó el rendimiento de las piscinas y de igual manera se redefinió el modelo de producción del país mejorando a su vez su posición a nivel global.

Entre 1968 y finales de los años 80, la cual es considerada una etapa tempranera en el desarrollo del camarón, su producción se realizaba de manera extensa. Los productores dependían de la captura de larvas silvestres del *Penaeus vannamei* y las sembraban en estanques sin mayores controles técnicos. La genética del camarón era totalmente natural puesto a que no existía selección ni una mayor selección en la características del animal y de su producción. El crecimiento era lento, la sobrevivencia impredecible y el rendimiento por hectárea bajo. Sin embargo, la simple reproducción de un ciclo biológico en cautiverio marcó un hito para el crecimiento exportador del país. (Piedrahita, 2018)

El desarrollo estructutal llego en los años 90, en el momento que Ecuador empezó a desarrollar laboratorios de reproducción y producción de larvas (*hatcheries*). Por primera vez se implementaron estrategias para no depender de larvas silvestres y comenzó a manejar la genética del cultivo. El objetivo inicial era asegurar la oferta de semillas durante todo el año. Sin embargo, fue la crisis del virus de la mancha blanca (wssv), alrededor del 1999-2001, la que actuó como catalizador para el salto científico. La mortalidad masiva obligo a investigadores, productores y laboratorios a replantear el modelo productivo del camarón. Con la finalidad de adecuar la genética y mejorarla para resistir enfermedades y ser mejores al momento de desarrollarse y adaptarse a los cambios.

A partir de 2003, surgió una nueva etapa marcada por la selección genética sistemática. Los laboratorios comenzaron a identificar las familias de *Penaeus vannamei* que mostraban mayor resistencia a patógenos, mejor crecimiento y mejor conversión alimenticia. Este proceso

no consistió en modificar el ADN artificialmente, se centró en seleccionar para reproducción únicamente a los individuos con mejores características morfológicas, método el cual se emplean en diferentes sectores como el de la agricultura, Ecuador evoluciono de un cultivo extensivo y dependiente de factores del entorno, hacia un sistema estandarizado con bases científicas.

La genética es uno de los principales diferenciadores del camarón ecuatoriano, las larvas que se producen tienen características adaptadas y acopladas a distintos escenarios a lo cuales el camarón puede ser expuesto, como cambios en la temperatura, enfermedades y diferentes niveles de salinidad. La selección genética se apoya en análisis de trazabilidad y marcadores de líneas genéticas.

La camaronera objeto de estudio trabaja con diversos códigos genéticas desarrollados en diferentes laboratorios, estos códigos contienen una trazabilidad integral en cuanto a las características del gen, adicional a ello se analiza la supervivencia en combinación a otros factores biológicos que intervienen en la producción y desempeño de cada siembra. A pesar de la estandarización de procesos del cultivo y siembra o condiciones del agua para cultivo, no es acertado establecer que los diferentes tipos de genéticas tengan los mismos requerimientos de alimentación, oxigenación y demás variables que aportan a la crianza y desarrollo, pues se asume que todas las larvas poseen los mismos requerimientos nutricionales y ambientales. Cada línea genética puede expresar variaciones en su metabolismo, tolerancia al estrés, conversión alimenticia y ritmo de crecimiento. Por tanto, ignorar estas diferencias podría limitar el potencial productivo de los organismos y afectar la correcta interpretación de los resultados experimentales.

En este contexto las múltiples líneas genética, la producción camaronera enfrenta un gran desafío para adaptar sus procesos a las necesitades de las larvas, así también necesitan

transformar la gestión del desarrollo de los códigos genéticos, para poder estandarizar el desarrollo y supervivencia de los mimos camarones. Ya que a medida que se implementan nuevas líneas genéticas con diferentes comportamientos biológicos es ineficiente aplicar esquemas de manejo homogéneos que no tienen en cuenta la respuesta de cada población cultivada.

Es así que la evolución tecnológica y diversificación genética de la producción camaronera enfrenta un nuevo desafío el cual es adaptar la gestión del cultivo de un enfoque estandarizado hacia un modelo basado en el seguimiento a cada una de las genéticas, con esto se necesita analizar datos referentes al progreso del camarón y toma de decisiones fundamentadas en datos. A medida que los sistemas productivos incorporan líneas genéticas con comportamientos biológicos diferenciados, se vuelve insuficiente aplicar esquemas de manejo homogéneos que no consideren la respuesta especifica de cada población cultivada. Esto crea vacíos entre el potencial productivo de la genética existente y lo que realmente se logra en campo.

Históricamente, la productividad en cultivos de camarón se ha medido en términos de biomasa final de cultivo, crecimiento promedio semanal de peso o índice de consumo alimenticio. Aunque estas métricas dan una idea del resultado del ciclo, tienen serias limitaciones porque no consideran el comportamiento temporal de los organismos. Pero específicamente estos indicadores no logran precisar en qué fases del ciclo se producen las mayores pérdidas, ni cómo influyen la genética y la nutrición en la supervivencia diaria del camarón. Esta inexactitud impide la identificación temprana de errores en la manipulación y disminuye la capacidad de respuesta ante incidentes.

Por lo cual, estudiar la supervivencia del camarón en los sistemas productivos pretende aportar a la comprensión de cómo interactúan estas variables en el tiempo y cómo finalmente

se manifiestan en la productividad del cultivo. Al hacer un análisis segmentado de la supervivencia, con esto se pueden hacer mejores elecciones para el desarrollo sostenible del ambiente.

2. PROBLEMÁTICA

Uno de los mayores desafíos que enfrenta la producción camaronera se encuentra en la mala administración del alimento. En muchos casos, las raciones diarias se calculan de forma generalizada, sin considerar las necesidades reales del camarón ni las diferencias genéticas entre lotes. Este manejo errático provoca desequilibrios en los estanques que, con el tiempo, afectan gravemente la salud y la supervivencia de los organismos. (Boyd C. E., 1990)

En un mercado que se ha visto afectado de manera constante a lo largo del camino, un sector que en los últimos 50 años ha soportado crisis económicas, desastres naturales, fenómenos sanitarios, colapsos de los mercados, incluso litigios legales. (VISTAZO, 2022) no es de sorprender que se encuentre de cierta forma acostumbrada a los porvenires, sin embargo; la incertidumbre que se adoptan a la hora del proceso alimenticio de camarón es una problemática autoimpuesta a la cual buscaremos la manera de disminuir el impacto que impone ante la producción de este.

Cuando se sobrealimenta, los camarones no pueden consumir todo el alimento. El alimento no consumido se descompone rápidamente, cambiando las condiciones del agua y creando un ambiente favorable para bacterias y enfermedades. Estos microorganismos no solo degradan la calidad del ambiente, sino que también causan enfermedades que se propagan rápidamente entre los lotes. En pocos días, una infección causada por una mala práctica alimentaria puede extenderse por todo el organismo, echando a perder toda la producción. (Tacon A. G., 1988)

Las consecuencias no se limitan al plano sanitario. Las enfermedades derivadas del exceso de alimento afectan directamente la calidad del camarón, reduciendo su valor comercial y dificultando el cumplimiento de los estándares exigidos por los mercados internacionales. (Balcázar, 2019)Los compradores suelen exigir productos uniformes, saludables y con trazabilidad garantizada; cualquier desviación en esos parámetros puede traducirse en devoluciones, pérdida de contratos o incluso restricciones de exportación. En este sentido, el problema deja de ser solo biológico para convertirse en una amenaza financiera y reputacional. (Smith & Furness, 2006)

Una cosecha afectada por enfermedades o mortalidad elevada implica pérdidas significativas de inversión y tiempo. En el cultivo de camarón, los brotes sanitarios incrementan los costos de alimentación, mantenimiento y tratamientos sanitarios, mientras que la productividad disminuye. (Engle, 2010)En algunos casos, las empresas deben vaciar y desinfectar completamente los estanques, interrumpiendo la continuidad del ciclo productivo. Estas interrupciones no solo deterioran los indicadores financieros, sino que también impacta la imagen de la empresa ante socios, clientes y autoridades sanitarias. (Engle, 2010)

El origen de estas pérdidas se encuentra, en gran medida, en la falta de control y medición precisa del alimento. Según Tacon & Akiyama (1997), la ausencia de información clara sobre cuánto, cómo y cuándo alimentar según la genética y el comportamiento de los camarones, las decisiones se convierten en un riesgo operativo constante. Adicionalmente, la ausencia de un enfoque analítico impide anticipar los efectos de las prácticas alimenticias en la supervivencia general del cultivo. Las empresas que operan bajo este esquema enfrentan una doble amenaza: el incremento de los costos y la disminución de la calidad. Con el tiempo, esta mezcla puede deteriorar la capacidad competitiva de la empresa, sus relaciones con los mercados internacionales y su posibilidad de crecer en el sector.

Desde la perspectiva de la gestión productiva, varios autores indican que la eficiencia en el cultivo de camarón no se basa en la cantidad de alimento que se vierte en el estanque, sino en la habilidad del sistema para generar información confiable para medir y modificar las prácticas de manejo. La falta de indicadores y mecanismos de seguimiento convierte la nutrición en un proceso empírico, incapaz de predecir resultados y controlar la variabilidad (Tacon A. G., 2002).

En este sentido, la integración sistemática de datos sobre alimentación, crecimiento y supervivencia constituye una condición necesaria para avanzar hacia esquemas productivos más estables. (Fast, Andrew W. & Lester, Lawrence J., 1992)

Es por ende que es indispensable avanzar hacia un modelo de producción más medible y controlado. Entender la relación entre la alimentación, genética y porcentaje de supervivencia del camarón no solo contribuyen a reducir perdidas, sino que también permite explicar las deferencias del camarón, no solo constituye a reducir perdidas, a su vez permite explicar las diferencias en el desempeño productivo entre cada una de las piscinas a cultivar de tal manera que la alimentación deja de ser una práctica aislada y se convierte en un factor estratégico para sostener la calidad del producto final y la competitividad de la empresa en mercados que demandan trazabilidad y control.

3. JUSTIFICACIÓN

La producción camaronera representa una de las actividades económicas más sólidas y dinámicas del Ecuador. Su crecimiento constante, impulsado por la calidad del producto y la apertura de nuevos mercados, ha convertido al país en un referente mundial en el sector acuícola. (Guerra, 2025). Sin embargo, a pesar de los avances tecnológicos y las mejoras en los sistemas de cultivo, aún persisten desafíos estructurales que limitan la eficiencia de la

producción y la sostenibilidad de las operaciones. Entre ellos, el manejo del alimento y su relación con la genética del camarón se destacan como factores determinantes en la supervivencia de las poblaciones cultivadas. (Lightner, 2011)

La intensificación que ha experimentado la industria camaronera ecuatoriana ha incrementado la dependencia del alimento balanceado como insumo principal del sistema productivo. Este escenario ha elevado la dificultad del manejo y el nivel de riesgo asociado a decisiones inadecuadas especialmente cundo no se consideran la característica biológicas y genéticas de los organismos cultivados. Es por esto que la supervivencia del camarón se transforma en un indicador critico en el desempeño del sistema, debido a que denota los efectos acúmulos en el manejo de la alimentación, la calidad del entorno y la eficacia productiva (Boyd C. E., 1990)

El sector enfrenta la necesidad de comprender mejor cómo la interacción entre la genética y la alimentación influye en los resultados de cada ciclo productivo. Durante años, muchas decisiones en las camaroneras se han tomado con base en la experiencia o la costumbre, sin contar con herramientas analíticas que permitan medir con precisión las consecuencias de cada ajuste alimenticio. Esta ausencia de indicadores y mecanismos de seguimiento limita la capacidad de identificar las causas reales de las perdidas productivas y de diferenciar si estas responden a factores genéticos, al manejo del alimento o a la interacción de ambos, generando un margen de incertidumbre que afecta la rentabilidad y estabilidad del proceso productivo. (Carlos Joel Cabrera Peñaloza, Rosana de Jesús Eras Agila, & Margot Isabel Lalangui Balcazar, 2022)

Frente a esta problemática, la implementación de un análisis de supervivencia enfocado en estas variables puede ofrecer una visión más clara y práctica para el manejo del cultivo. Desde el punto de vista científico, el análisis de supervivencia ha sido ampliamente utilizado

en disciplina como la biología, la medicina y la ecología para estudiar la probabilidad de ocurrencia de eventos a lo largo del tiempo; sin embargo, su aplicación en sistemas acuícolas, particularmente en el cultivo de camarón, aun es limitada. Diversos autores señalan que el uso de métodos estadísticos orientados al tiempo permite una comprensión más precisa de los procesos biológicos y productivos, superando las evaluaciones tradicionales basadas únicamente en promedios o resultados finales de cosecha. Por esto, a través de la recopilación y el tratamiento de datos, se busca identificar patrones que ayuden a los productores a administrar mejor los recursos y a establecer estrategias más rentables. Con ello, se pretende reducir los márgenes de error en la planificación alimentaria, optimizar los costos de operación y mejorar la consistencia de los resultados en el tiempo (David G. Kleinbaum & Mitchel Klein, 2012)

Dentro del campo empresarial, los datos obtenidos de este tipo de estudios se convierten en un recurso primordial para tomar decisiones sólidas, prever comportamientos en distintos escenarios productivos y establecer políticas internas en procesos de nutrición, selección genética y planificación tanto de siembra y cosecha. La aplicación de este conocimiento no solo genera eficiencia, sino que también fortalece la competitividad de las empresas frente a un entorno global cada vez más exigente. (Engle, 2010)

Así mismo la información resultante del análisis de supervivencia segmentada por genética y tipo de alimentación puede convertirse en una herramienta estratégica para la gestión de crianza y desarrollo del cultivo acuícola. Contar con este tipo de evidencia permitiría optimizar la planificación alimentaria, reducir perdidas asociadas a manejos ineficientes y mejorar la estabilidad de los resultados productivos entre ciclos. De esta manera, la investigación no solo aporta al conocimiento técnico del sector camaronero, sino que responde a una necesidad real de sostenibilidad económica y competitividad en mercados cada vez más exigentes. (Jonathan Endara, 2023)

La incorporación de analsis técnicos en la toma de de decisiones impulsan una mejora hacia modelos de producción modernos y estables, los cuales miden, comparan y entienden los factores que inciden en la supervivencia del camarón. Esto es un paso hacia la profesionalización integral del sector, donde la productividad no depende únicamente de la experiencia, sino del respaldo que brindan los datos y la observación científica. Esta perspectiva apunta a un sistema de cultivo más controlado, rentable y equilibrado con su entorno natural.

4. ALCANCE

El presente trabajo titulado “Análisis de supervivencia de la producción del camarón segmentado por genética y alimentación”, tiene como objetivo comprender de manera integral los factores que influyen en la durabilidad, la salud y el rendimiento de las poblaciones camaroneras en los entornos productivos, enfocándose en como la genética del camarón y los diferentes regímenes alimenticios afectan su supervivencia para generar información que permita optimizar los cultivos.

Esta investigación está dirigida principalmente a tres grupos de interés. Por un lado, los productores y gestores de granjas acuícolas, quienes podrán utilizar los resultados para mejorar los protocolos de alimentación, incrementar la eficiente en el uso de recursos y reducir las mortalidades por enfermedades o depredación.

Por otro lado, los técnicos del sector camaronero encontraran en este estudio una herramienta para comprender de qué manera las diferencias genéticas y los hábitos de alimentación repercuten en la salud de los lotes y en la calidad del producto final.

Finalmente, los inversionistas responsables de la toma de decisiones en el sector acuícola podrán evaluar con mayor precisión los riesgos operativos relacionados con la supervivencia del camarón y basar sus decisiones financieras en indicadores técnicos más confiables.

5. OBJETIVOS

5.1. **Objetivo General**

Analizar la producción de camarón basado en la supervivencia segmentada por genética y alimentación.

5.2. **Objetivos Específicos**

- Revisar la literatura sobre el comportamiento de producción del camarón y diferentes conceptos de Machine Learning que se pueden implementar en este análisis.
- Emplear modelos basados en conceptos de machine learning para el análisis de datos basados en el histórico de producción de una empresa Camaronera.
- Comparar los resultados con estudios previos que utilizan metodologías similares.

6. MARCO TEORICO

6.1. **Modelos de Aprendizaje Supervisado**

El aprendizaje supervisado es un pilar estadístico fundamental al momento de construir buenos ensambles u combinaciones de los modelos a emplear, es cerciorándose de que los modelos individuales que se van a combinar sean precisos como múltiples o diversos. En esta parte se enfocará el problema de la exactitud de los modelos en conjunto con la teoría

estadística. Se mostrarán los modelos más representativos de los algoritmos de aprendizaje supervisado.

6.2. **Teoría de la decisión.**

Se plantea un dilema el cual se deberá resolver, para esto se deben introducir las bases teóricas sobre las cuales se sostiene el aprendizaje estadístico mostradas a continuación:

Sea $X \in \mathbb{R}^p$ un vector fortuito que se denomina comúnmente como un vector de características y que $Y \in \mathbb{R}$ que es una variable aleatoria de respuesta continua, siendo que p(x,y) su distribución conjunta. Según Devore, J "el objetivo en estos casos siempre será encontrar una función f(x) que sea capaz de predecir la variable respuesta y dado un vector de características x".

Es así que para poder conseguir esto, se debe emplear un proceso que requiere una función de perdida l que nos ayuda a identificar la relación de Y con f(X). O esta a su vez expondrá el error que ocurre en esta predicción dependientes de las variables a estudiar, es así como se puedo construir un modelo que minimice la cantidad de error.

6.3. **Regresión**

Para los problemas de regresión, debido a que la o las variables son continuas las funciones de perdida más usuales son el error cuadrático, $L\big(Y, F(X)\big) = (Y - f(X))^2$ y la normal dice que $L\big(Y, F(X)\big) = |Y - f(X)|$. Así mismo en el contexto de limitantes para la clasificación donde y como variable categórica. En este caso tomaría la siguiente forma $L\big(Y, F(X)\big) = \theta\big(-YF(C)\big)$ dado que θ es una función escalonada de heaviside.

$$\theta(x) = \begin{cases} 1\ SI\ X\ <\ 0 \\ 0\ SI\ X \geq\ 0 \end{cases}$$

Explicando así 0 cuando x se clasifica correctamente y 1 cuando no.

6.3.1. **ETE**

Ahora si bien es cierto que en un escenario totalmente controlado la lógica para aplicar la distribución $\mathbb{P}\ (X, Y)$ un posible factor para elegir la función f a predecir sería la minimización de su error de test esperado conocida también como (ete), esta según "se puede definir como el error que tendrá un modelo f al predecir con nuevos atributos (datos) generados bajo la distribución $\mathbb{P}(X, Y)$"

Matemáticamente esta se expresa de la siguiente manera

$$ETE(F) = \left[L\big(Y, F(X)\big)\right] = x_x y L(Y, F(X)\mathbb{P}(X, Y) DXDY$$

6.3.2 **Función de Pérdida**

Para poder emplear una regresión usualmente se deberá aplicar una función de perdida para poder emplear sus propiedades acordes a lo buscado y, se describe de la siguiente forma:

$$ETE(F) = \left[L\big(Y, F(X)\big)\right] = \int x_x y L(Y, F(X)\mathbb{P}(X, Y) DXDY$$

Para poder emplear una regresión usualmente se deberá aplicar una función de perdida para poder emplear sus propiedades acordes a lo buscado y, se describe de la siguiente forma:

$$ETE(F) = E[Y - F(X)]^2 = \int x_x y [Y - F(X)]^2 \mathbb{P}(X, Y) DXDY$$

Así, es como se factoriza la densidad conjunta de $\mathbb{P}(X,Y) = \mathbb{P}(Y|X)\mathbb{P}(X)$ y dividiendo la integral doble utilizando el teorema de fubini:

$$ETE(F) = \int_x (\int_y (Y - F(X))^2 \, \mathbb{P}(Y|X)DY)\mathbb{P}(X)DY$$

En el caso de que fuera condicionada se transformaría en lo siguiente:

$$ETE(F) = E_x E_{y|X}([Y - C]^2|\text{x})$$

A esta se debe minimizar específicamente en x:

$$F(X) = ARGMIN_C \, E_{y|x}([Y - C]^2|X)$$

Dando como solución:

$$F(X) = E[Y|X = X]$$

Esto se conoce como la función de regresión, e indica la mejor predicción posible para y en un punto x concreto es su esperanza condicionada a dicho x. Sin embargo, debemos tener en cuenta que la distribución conjunta $\mathbb{P}(X,Y)$ es completamente desconocida, lo unico certero en el caso serán el conjunto de entrenamiento finito con n datos.

$$S = [(x_1, y_1), \dots, (x_N, y_{\mathbb{N}})]\}$$

Aunque no se puedan emplear directamente las anteriores derivaciones expuestas, se podría aproximar el error en la hipótesis predicha con métricas como la función de riesgo:

$$RE\ (f) = \frac{1}{N}\sum_{\lambda=1}^{\mathbb{N}} L\big(y_i, f(x_i)\big)$$

Esta fórmula expresa que un modelo estadístico fijado en función de "x" se entrenó en un conjunto finito de s. Esto evalúa lo mal o bien que el modelo se ajusta a los datos de entrenamiento.

6.4. **Modelos Lineales Generalizados**

Es un conjunto de variables aleatorias independientes $y_1, y_2, \cdots, y_n$ cuya función de densidad, o función de probabilidad. (Ross, 2014)

6.4.1. **Componentes mlg**

Un modelo lineal generalizado está conformado por tres componentes:

1. Conjunto de η variables respuestas independientes, cada una de ellas está asociada a una distribución perteneciente a la familia exponencial.
2. Un vector de coeficientes β junto con la matriz de diseño X, los cuales definen el predictor lineal correspondiente a cada observación, expresado como $\beta^T x_i$.
3. Una función de enlace monótona y diferenciable que establece la relación entre la media μ_i de la variable respuesta y su predictor lineal, de la forma:

$$g(\mu_i) = \beta^T x_i \,.$$

6.4.2. **Componente aleatorio:**

La componente aleatoria está constituida por el vector aleatorio$Y = (y_1, y_2 \dots, y_n)$ cuyos componentes son independientes e idénticamente distribuidos, y cuya función pertenece a la familia exponencial. (McCullagh & Nelder, 1989)

La función de densidad asociada a la familia exponencial se expresa como:

$$p(y_1 |\theta_i , \phi) = exp\left\{\frac{y_i\theta_i - b(\theta_i)}{a_i(\phi)} + c(y_i, \phi)\right\}$$

Donde:

- $\theta_i =$ es un coeficiente natural o canónico
- $\emptyset =$ es un coeficiente adicional de escala o dispersión
- $a_i\,(\cdot), b(\cdot)\; y\; c\;(\cdot) =$ son funciones especificas

Si $\emptyset$ es conocido, el modelo pertenece a la familia exponencial lineal

Si $\emptyset$ es desconocido, el modelo se clasifica como un modelo de dispersión exponencial

6.4.3. **Componente sistemática:**

La componente sistemática está formada por las variables explicativas, cuya combinación lineal da lugar al predictor lineal. (McCullagh & Nelder, 1989)

Dicho predictor puede representarse como:

$$n_i = \sum_{j=1}^{p} \beta_j x_{ij} \ , \quad \text{i} = 1, \dots, n \ ,$$

- $n_i =$ Predictor lineal asociado a i-esima observación, Representa la combinación lineal de las variables explicativas para ese individuo.
- $\sum_{y=1}^{p} =$ Sumatoria sobre todas p variables incluidas en el modelo.
- $p =$ Numero total de predictores o variables explicativas del modelo.
- $\beta_y =$ Parámetro o coeficiente asociado a la j-ésima variable explicativa. Mide el efecto de x_J sobre la variable respuesta.
- $x_{ij} =$ Valor de la j-esima variable explicativa correspondiente a la i-esima observación.
- $\text{i} = 1, \dots, n =$ Indice que recorre las n observaciones de la muestra.
- $n =$ Tamaño de la muestra o numero total de observaciones.

6.4.4. **La función de enlace**

Con el fin de modelar relaciones no lineales entre la variable respuesta y las variables explicativas, se incorpora la función de enlace$g(\cdot)$, la cual establece la conexión entre la media de la variable respuesta y el predictor lineal de la forma:

$$g(\mu) = X\beta = n$$

Se denomina enlace canónico de una distribución a aquella función de enlace que satisface la relación (Cayuela, 2015)

$$\theta = g(\mu) \cdot$$

- $g(\cdot) =$ Función de enlace.

- $\mu =$ Media de la variable respuesta.
- $X =$ Matriz de diseño que contiene los valores de las variables explicativas para todas las observaciones.
- $\beta =$ Vector de parámetros o coeficientes del modelo.
- $X\beta =$ Predictor lineal.
- $n =$ Notación alternativa del predictor.
- $\theta =$ Parámetro canónico de la distribución.

6.5. **Regresión Logística**

La regresión logística es uno de los pilares fundamentales para poder realizar los componentes que se vinculan a cualquier tipo de dato, según (David W. Hosmer & Stanley Lemeshow, 2000), La regresión es uno de los métodos integrales más utilizables para poder describir la probabilidad de ocurrencia binaria o también conocidas como dicotómicas. Así también "La regresión logística es un algoritmo de machine learning supervisado en la ciencia de datos" (Lee, S-F.)

Así también debemos tener en cuenta que existen tres tipos de regresión logística, como son: "la regresión logística binaria, la cual se enfoca en identificar si la variable dependiente tiene una naturaleza dicotómica, la cual especifica que solo deberán existir dos posibles resultados. Según (Guillén & Alonso, 2020) la regresión logística binaria se representa como un modelo en el cual solo se pueden definir dos posibles resultados, ya sea positivo o negativo. Así mismo permite evaluar la probabilidad de ocurrencia.

6.5.1. **Diferencia entre regresión lineal y logística.**

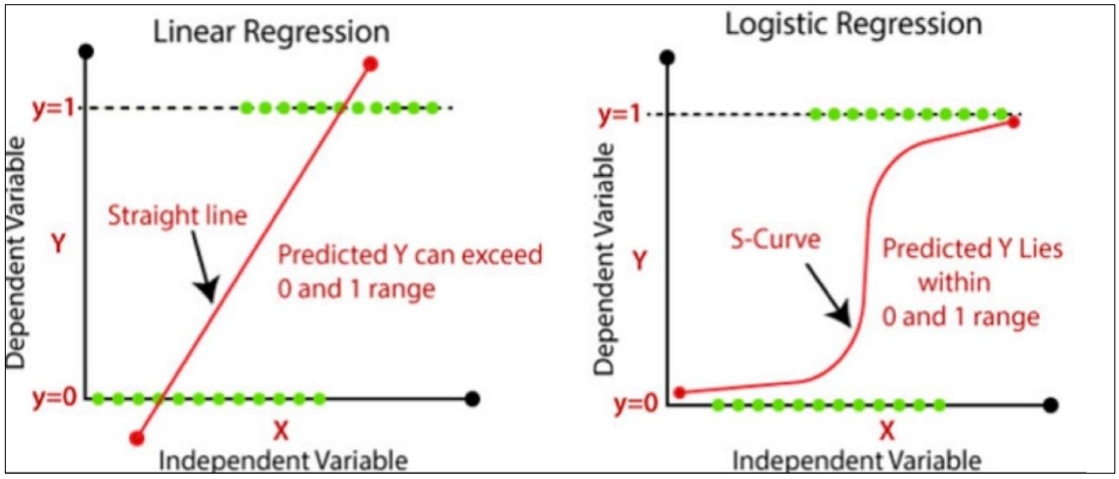


Figura 1*: Comparación entre modelo de Regresión Lineal y Regresión Logística.*

La regresión logística, en relación con el modelo lineal, es un modelo que tiene la tendencia lineal en donde se estudia la relación entre variables predictoras o independientes y un output. Ambas son empleadas para ratificar que lo predicho genere un valor p por consiguiente se puede evaluar que tan significativa es la relación de las variables. Así mismo una P valor bajo sugiere que contribuye significativamente al modelo y una de las fórmulas con las cuales podemos realizar esta evaluación es la siguiente:

En la regresión lineal:

$$y + \beta_0 + \beta_1 X_1 + \cdots + \beta_k X_k + \varepsilon$$

Aplicamos estadístico T-student: $t_1 = \frac{\widehat{\beta}}{SE(\widehat{\beta_i})}$

Donde:

$\hat{\beta}_i$ es la estimación del coeficiente

$SE(\hat{\beta}_i)$ es el error estándar del coeficiente

Así podemos nosotros identificar el P-valor:

$$p - valor = 2 * [1 - F_t(|t_i|, n - k - 1)]$$

Donde:

$F_t(*)$ es la función de distribución acumulada de T-Student

N es el número de observaciones

K es número de variables explicativas

Regresión logística:

$$\ln\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 X_1 + \cdots + \beta_k X_k$$

Donde empleamos el estadístico Wald este se encarga de analizar o contrastar la nulidad de los parámetros:

Este de debe aplicar a cada una de las variables del modelo. Se encarga de testar un parámetro especifico:

$H_0 = b_i =$ 0 sinifica que no es una variable significativa

$H_1 = b_i \neq$ 0 significa que es variable significativa en el modelo

Para calcularla se emplea la siguiente formula:

$$z = \frac{b_i}{S_{bi}}$$

Donde:

b_i es el coeficiente estimado de la variable X_i.

S_{bi} es el error estándar del coeficiente.

Para obtener en P-Valor se calcula de la siguiente manera.

$$P - valor \ = \ 2[1 - \Phi(|z|)]$$

P-valor se obtiene mediante el estadístico z, el cual, mediante una distribución normal, la función $\Phi(\cdot)$ representa la función de distribución acumulada. Y el contraste se realiza de forma bilateral. Por lo que la probabilidad obtenida de debe multiplicar por dos.

6.5.2. **Funcionamiento Regresión logística**

La regresión logística es un modelo utilizado para resolver problemas de clasificación, especialmente en escenarios en los cuales se desea estimar la probabilidad de pertenecer a una clase particular dadas ciertas variables predictoras. Específicamente busca representar de manera lineal la relación entre los predictores y los logits (log -odds) o razones de probabilidades, garantizando que estas probabilidades desarrolladas se encuentren en el rango de 0 y 1. Para esto se formula de la siguiente manera:

En un dilema de clasificacion de k clases, la regresión logística modela los logs odds de cada clase en correlación a una clase base, mediante expresiones lineales, el modelo esta definido por:

$$LOG\frac{PR(G = k|X = X)}{PR(G = K|X = X)} = \beta_{K0} + \ \beta_K^T X, k = 1,2,..,K-1$$

Donde:

G es igual a la clase verdadera

X es el vector de predictores

β_{K0} es el intercepto de la clase k

β_K es el vector de coeficientes para la clase k, se debe tener en cuenta que la clase k es una referencia.

Una de las propiedades importantes para tener en cuenta es que, las probabilidades siempre están entre 0 y 1, es así como la construcción de la fórmula es la siguiente:

$$\sum_{k=1}^{k} PR(G = k|X = X) = 1$$

Así también existen casos especiales en los cuales se pueden presentar regresiones logísticas binarias, esta tiene como objetivo modelar la probabilidad de la concurrencia de un evento y =1. Ya que las probabilidades están enmarcadas en 0 y 1.

Una vez entendido esto, podemos llegar a la formulación base de las expresiones generales para las probabilidades:

$$PR(G = K|X = X) = \frac{EXP(\beta_{K0} + \beta_K^T X)}{1 + \Sigma_{L=1}^{k-1} EXP(\beta_{K0} + \beta_K^T X)} \quad PARA\ K = 1, \dots, K-1$$

Y así para la clase base es:

$$PR(G = K|X = x) = \frac{1}{1 + \Sigma_{L=1}^{k-1} EXP(\beta_{K0} + \beta_K^T X)}$$

Donde:

β_K^T es la combinacion lineal

Exp es exponencial

K es la clase especifica

X son vectores de los predictores

G es la variable de clase o etiqueta predecir.

Con esto aseguramos que las probabilidades no sean negativas y sumen uno.

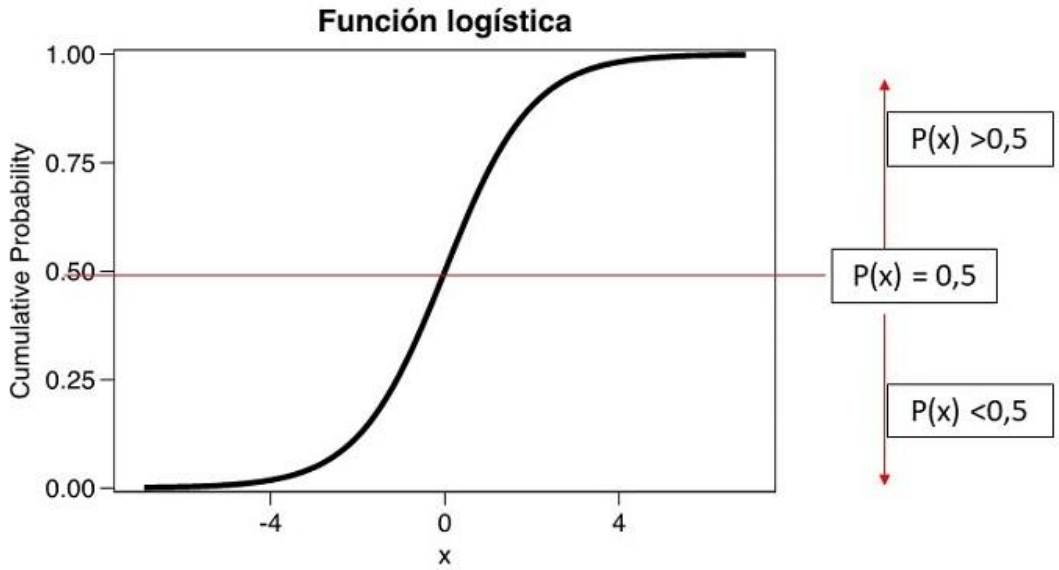


Figura 2*: Demostración gráfica de Regresión Logística.*

Función Sigmoide

La función sigmoide, tambien conocuna como funcion logistica, es una función matematica continua y no lineal que se encarga de tranformar cualquier número real en un valor comprendido entre 0 y 1.

Según James (2021) en An Introduction to Statistical, La función logistica es utilizada para modelar probabilidades, debido a que garantiza que las predicciones se encuentren dentro del rango de 1 y 0

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Donde:

e Significa la constante de euler

$$z \in R$$

6.5.3. **Odds-ratio**

El concepto general de odds según Monserrat G & María .(2020) “Los odds son la razón de momios u oportunidad de ocurrencia de un suceso entre la probabilidad de ocurra mismo que no ocurra” (Monserrat G & María A. 2020)

Así también, según Douglas C & Elizabeth A (2012), los odds pueden interpretar la estimación de una variación de una unidad en la variable predictora.

Se describen de la siguiente manera $odds = \frac{\boldsymbol{p}}{\boldsymbol{1-p}}$ esta medida compara la capacidad de ocurrencia relativa de un evento por ende se coloca como una base para interpretación de los modelos de regresión logística.

Los logs–odds se establecen como un algoritmo natural de la lógica entre la probabilidad de que un evento ocurra y la posibilidad de su no ocurrencia. Con esto se puede establecer una respuesta binaria a un conjunto de variables explicativas mediante el modelo.

Así también tenemos modelos de los log- odds.

$$Ln\left(\frac{P\left(Y = \frac{1}{X}\right)}{1 - P\left(Y = \frac{1}{X}\right)}\right)$$

El siguiente paso es describirlo en términos de odds:

$$\frac{P\left(Y=\frac{1}{X}\right)}{1-P\left(Y=\frac{1}{X}\right)}=\ \frac{e^{b}c\ +\Sigma_{i=1}^{n}(b_i x_i)}{1+e^{b}0+\Sigma_{i=1}^{n}(b_i x_i)}$$

Por consiguiente, se calculan los distintos valores de la probabilidad para las combinaciones posibles entre la variable Y en relación con la X independiente.

$$\frac{p\left(Y=\frac{1}{X}=1\right)}{1-P\left(Y=\frac{1}{X}=1\right)}=\frac{e^{b}0+b_1}{1+e^{b}0+b_1}$$

$$\frac{p\left(Y=\frac{1}{X}=0\right)}{1-P\left(Y=\frac{1}{X}=0\right)}=\frac{e^{b}0}{1+e^{b}0\{}$$

$$\frac{p\left(Y=\frac{0}{X}=1\right)}{1-P\left(Y=\frac{0}{X}=1\right)}=\frac{1}{1+e^{b}0^{+b_1}}$$

$$\frac{p\left(Y=\frac{0}{X}=0\right)}{1-P\left(Y=\frac{0}{X}=0\right)}=\frac{1}{1+e^{b}0}$$

Donde se entiende que OR calculan según indica (Douglas C. Montgomery & Elizabeth A. Peck, 2000) “se estiman como la razón entre odds, donde el resultado de la variable Y se presenta entre individuos que toma el valor de Y=1, así la variable independiente X puede representar el no”

Entonces X toma los siguientes valores: X= 1 y X=0.

$$\frac{\frac{p\left(Y=\frac{1}{x=1}\right)}{1-p\left(y=\frac{1}{x=1}\right)}}{\frac{p\left(y=\frac{1}{x=0}\right)}{1-p\left(Y=\frac{1}{x=0}\right)}} = e^{b}1$$

Así es que como ejemplo tenemos que la mortalidad del camarón, dentro de un ciclo productivo la tasa de supervivencia es del 80%, los odds de supervivencia se deberán obtener de la relación entre la supervivencia y porcentaje de mortalidad, dado como resultado que si se obtiene un valor mayor a 1 indica que este es más probable a sobrevivir.

6.5.4. **Función de Verosimilitud**

La función de verosimilitud es aquella que evalúa qué tan probables son los datos observados, dado un conjunto de parámetros del modelo. Se define como el producto de las funciones de probabilidad individuales para todas y cada una de las observaciones de la muestra. En el modelo, se asume que el término de error sigue una distribución normal estándar, es por esto que las probabilidades se expresan mediante la función de distribución acumulada de una normal estándar, denotada por Φ(.), donde (.) significa que la función puede recibir cualquier argumento.

La función de verosimilitud del modelo es:

$$L(\beta) = \prod_{i=1}^{n} [\Phi(x'_i\beta)]^{y_i}[1-\Phi(x'_i\beta)]^{1-y_i}$$

Donde:

$L(\beta)$: Representa la probabilidad conjunta de observar los datos, dada una combinación de parámetros

$\prod_{i=1}^{n}$.: Producto de los términos para todas las observaciones de la muestra.

..^{y_i}: Variable dependiente para la observación i, que toma valores 0 o 1.

x_i: Vector de variables explicativas para la observación i

β : Vector de parámetros del modelo que se desea estimar.

x'_i: Producto escalar entre los regresores y los parámetros: representa el índice lineal de la observación i.

$\Phi(x'_i\beta$): Función de distribución acumulada de un normal estándar evaluada en $xi'\beta$, que da la probabilidad de que yi=1 dado xi

$[\Phi(x'_i\beta)]^{y_i}$: Toma el valor $[\Phi(x'_i\beta)]$·sí yi=1, y toma 1 si y1=0

$[1-\Phi(x'_i\beta)]^{1-y_i}$: Toma el valor$[1-\Phi(x'_i\beta)]$ si yi=1, y toma 1 si yi=0

De esta manera, la función se encarga de seleccionar la probabilidad correspondiente para cada observación según su valor observado de yi. Esta función de verosimilitud es globalmente cóncava en el caso del modelo, lo cual asegura que las condiciones de segundo orden para un máximo se cumplan y, por tanto, exista un único óptimo.

6.5.5. **Máxima verosimilitud (MLE)**

La máxima verosimilitud es un procedimiento de estimación, el cual consiste en encontrar los valores de los parámetros β que maximizan la función de verosimilitud, en otras palabras, los parámetros que hacen más probable la observación de los datos reales bajo el modelo especificado.

El procedimiento se implementa numéricamente (por ejemplo, con software estadístico) para encontrar los coeficientes estimados.

Para facilitar los cálculos, en la práctica se trabaja con la log-verosimilitud, que transforma el producto en una suma:

Donde:

$\log \Phi(x'_i\beta)$:Logaritmo de la probabilidad de éxito. Solo se suma si yi=1.

$\log(1 - \Phi(x'_i\beta))$:Logaritmo de la probabilidad de fracaso. Solo se suma si yi=0.

Los estimadores que se obtienen con la máxima verosimilitud cuentan con buenas propiedades estadísticas, siempre que haya condiciones regulares, como que sean muestras grandes. Algunas de estas propiedades son:

Insesgadez: El valor esperado del estimador coincide con el valor real del parámetro en promedio.

Consistencia: El estimador converge al valor verdadero del parámetro (Wasserman, 2004)

6.5.6. **Pseudo R2**

La regresión logística al ser un modelo no lineal no podemos usar como método de ajuste el r2 tradicional entonces en su lugar, se recurre a medidas alternativas denominadas pseudo R2, (Hemmert, Laura M. Schons, Jan Wieseke, & Heiko Schimmelpfennig, 2016)

6.5.7. **McFadden**

Uno de los más utlizados dentro de la regresión probit y logit, lo usamos cuando comparamos que tanto el modelo estimado trabajado mejora en comparación a un modelo sin sus variables explicativas. (Hosmer, D. W & Lemeshow, S., 2000)

$$R^2 = 1 - \frac{\ell(\beta)}{\ell(0)}$$

- $\ell(\beta)$: log-verosimilitud del modelo estimado.
- $\ell(0)$: log-verosimilitud del modelo con solo una constante (modelo nulo)

6.5.8. **Cox & Snell**

Tiene un propósito parecido al R2 tradicional, este no se puede usar para comparar modelos como el McFadden, se usa cuando se desea una medida que sea basada en la relación entre verosimilitudes (Hosmer & Stanley, 2000)

$$R^2_{Cox\ \&\ Snell} = 1 - \left(\frac{L_0}{L_1}\right)^{\frac{2}{n}}$$

- L_0: verosimilitud del modelo nulo.
- L_1: verosimilitud del modelo estimado.
- n : número de observaciones.

6.5.9. **Nagelkerke**

Cuando queremos un pseudo R2 escalado para que el valor máximo sea 1, lo que lo haría ser una correlación de Cox & Snell (Nagelkerke, 1991)

$$R^2_{NagelKerke} = \frac{R^2_{Cox\ \&\ Snell}}{1 - L_0^{\frac{2}{n}}}$$

- L_0: verosimilitud del modelo nulo.
- n : numero de observaciones

6.6. **Clasificación del modelo**

6.6.1. **Interpretación: El Odds Ratio (OR)**

Dado que la Regresión Logística es lineal en la escala del Log-Odds pero no en la probabilidad, la interpretación de los coeficientes de Beta no puede hacerse directamente. Por ello, la medida estándar para cuantificar el efecto de un predictor es el Odds Ratio (OR).

El Odds Ratio para un predictor x se calcula exponenciando su coeficiente estimado $\hat{\beta}_i$

$$OR = e^{\widehat{\beta}_i}$$

El OR es una medida de asociación que indica el cambio multiplicativo en las chances (Odds) de que el evento de interés ocurra (ej. un crédito sea pagado) por cada aumento de una unidad en la variable predictora x_i, manteniendo todas las demás variables constantes (ceteris paribus).

La interpretación se centra en tres escenarios clave: (Kuhn & Johnson, Feature Engineering and Selection: A Practical Approach for Predictive Models, 2019)

- Si OR > 1: El predictor x_i está asociado positivamente con el evento. Por cada aumento de una unidad en x_i, las chances del evento se multiplican por el valor del OR.

- Si OR < 1: El predictor $x_i$está asociado negativamente con el evento. Por cada aumento de una unidad en x_i, las chances se reducen o se dividen por 1/ OR

- Si OR = 1: La variable x_i no tiene asociación con la ocurrencia del evento.

"El Odds Ratio es la interpretación principal para la Regresión Logística, ya que, a diferencia de la Regresión Lineal, el efecto de una variable en la probabilidad no es constante, sino que depende del valor actual de esa probabilidad. El OR proporciona una medida constante de asociación en la escala logarítmica linealizada" (Chatterjee & Hadi, 2015)

6.6.2. **Mecanismo de Clasificación por Probabilidad**

La Regresión Logística por probabilidad hace la clasificación mediante un proceso de umbralización (thresholding) que traduce la probabilidad continua de la función Sigmoide $\pi(x) \in [0,1]$a una clase discreta $\hat{Y} \in \{0,1\}$

6.6.3. **Umbral de Decision.-**

La función Sigmoide produce una estimación de probabilidad, no la clase final. Por lo tanto, se aplica un Umbral de Decisión $\mathcal{T}$ para asignar la observación a una de las dos categorías de la variable de respuesta (Hastie, Tibshirani, & Friedman, The Elements of Statistical Learning (2nd ed.), 2009)

La Regla de Decisión se formaliza algebraicamente de la siguiente manera:

$$Y = \begin{cases} 1 \; si \; \pi(x) \geq \mathcal{T} \\ 0 \; si \; \pi(x) < \mathcal{T} \end{cases}$$

6.6.4. **Dummies**

Una variable dummy es un tipo de variable binaria que representa la pertenencia a una categoría especifica de una variable cualitativa, permitiendo su inclusión en un modelo de regresión. Su interpretación siempre se hace en comparación con una categoría de referencia (te Grotenhuis & Pelzer, 2015)

Representada de la siguiente manera:

$$D = \begin{cases} 1 \\ 0 \end{cases}$$

1= si pertenece al grupo

0= si no pertenece al grupo

6.6.5. **Escalado de variables**

El escalado de variables es una técnica de preprocesamiento que consiste en transformar de las variables numéricas para que presenten rangos comparables. Esta transformación es necesario debido a que las variables pueden tener magnitudes muy diferentes, lo que puede influir negativamente en el proceso de estimación y en la interpretación de los coeficientes del modelo. (Kuhn & Johnson, Applied Predictive Modeling, 2013)

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}}$$

Donde:

- $x =$ valor original
- $x_{min =}$ valor mínimo de la variable
- x_{max} =valor máximo de la variable
- $x' =$ valor escalado

6.6.6. **Overfitting**

El overfitting o conocido al español como “sobreajuste” ocurre cuando un modelo incorpora uno o mas componentes que presentan un desempeño aparentemente significativo en el conjunto de datos de entrenamiento pero no mantendrían ese mismo nivel de desempeño en otros conjuntos de datos provenientes de la misma población de datos. En consecuencia, el modelo se adapta excesivamente a las particulidades del conjunto de entrenamiento y pierde

capacidad de generalización (James, Witten, Hastie, & Tibshirani, An Introduction to Statistical Learning, 2021)

Se expresa de la siguiente manera:

$$Pr(max(x1, \ldots, xn) \geq k) = 1 - (1 - p)n$$

$X1, \ldots, Xn$ = puntajes de los componentes evaluados.

k = valor critico (umbral).

$p = Pr(Xi \geq k)p$ = probabilidad de que un solo componente supere el umbral.

n = numero de componentes evaluados.

6.7. **Pruebas de validación**

6.7.1. **Matriz de Confusión**

A la hora de evaluar la eficacia de nuestro modelo es necesario y fundamental la utilización de las respectivas métricas para poder cuantificar su rendimiento a través de la matriz de confusión.

Antes de comenzar hay que sentar las bases de los significados de las clasificaciones positivas y negativas:

True Positives (Verdadero Positivo): Observaciones positivas que el modelo clasificó como positivas.

True Negatives (Verdadero Negativo): Observaciones negativas que el modelo clasificó como negativas.

False Positive (Falso Positivo): Observaciones negativas que el modelo clasificó erróneamente como positivas.

False Negatives (Falso Negativo): Observaciones positivas que el modelo clasificó erróneamente como negativas.

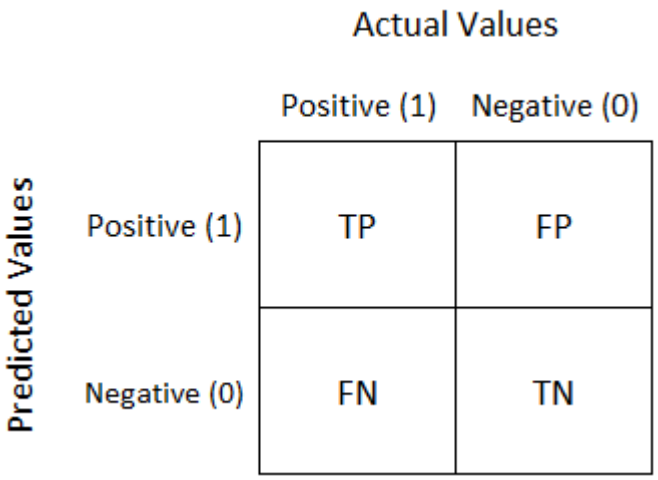


Figura 3: *Demostración gráfica de una Matriz de Confusión.*

6.7.2. **Exactitud (Accuracy)**

El Accuracy nos sirve como métrica de evaluación para visualizar la proporción de clasificaciones correctas sobre el total de observaciones, mide que tanto el modelo clasifica de manera correcta el outcome. Es decir, mide que tanto evalúa de manera correcta el modelo.

El Accuracy se mide en una escala de 0 a 1, o de manera porcentual.

$$\boldsymbol{Accuracy = \frac{Correct\ predictions}{All\ predictions}}$$

Se calcula de la división donde en el numerador se encuentran el total de clasificaciones correctas, ya sea positiva o negativa, mientras que en el denominador se encuentran todas las clasificaciones realizadas.

6.7.3. **Precision**

La precisión mide en una escala del 0 a 1, o porcentualmente que tanto fueron realizadas las clasificaciones positivas de manera correcta, al medir la proporción de cuantos casos clasificados como positivos son realmente positivos.

Mientras más alto sea el valor de precisión podemos interpretar que el modelo comete menos falsos positivos, es decir; clasificar erróneamente un negativo como positivo.

$$\boldsymbol{Precision} = \frac{\boldsymbol{True\ Positives}}{\boldsymbol{True\ Positives} + \boldsymbol{False\ Positives}}$$

Se calcula de la división donde en el numerador se encuentran el total de clasificaciones positivas verdaderas, mientras que en el denominador se encuentran todas las clasificaciones positivas realizadas (falsas y verdaderas).

6.7.4. **Recall**

El Recall nos sirve como métrica de medición para visualizar que tanto el modelo clasifica de manera correcta las observaciones positivas reales. Se podría interpretar como la capacidad del modelo para detectar cualquier tipo de observaciones positivas, minimizando de esta manera los falsos negativos.

$$\boldsymbol{Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives}}$$

El Recall se mide en una escala de 1 a 0, o de manera porcentual mediante la división en la cual el total de clasificaciones positivas verdaderas, la suma de las clasificaciones positivas verdaderas y las clasificaciones negativas falsas.

6.7.5. **F1 Score**

Mientras que las anteriores métricas de evaluación se hacen directamente sobre las clasificaciones realizadas, el f1 score no es más que la media armónica entre precisión y recall, es decir mientras más altos sean los valores de dichas métricas, mayor será el resultado el F1 Score, si uno de ambos valores es bajo, el valor en sí mismo del F1 Score se verá afectado drásticamente.

El F1 Score se mide en una escala de 1 a 0, o de manera porcentual y es el resultado de la media armónica entre el Recall y la Precision, consecuentemente presenta una mayor utilidad a la hora de interpretar el balance que tiene el modelo en cuanto a su capacidad para poder detectar positivos y que efectivamente dichos hallazgos sean correctos.

$$\boldsymbol{F1 = 2 * \frac{Precision * Recall}{Precision + Recall}}$$

6.7.6. **ROC**

El grafico roc o también conocido como un gráfico de características operativas de receptor, según tom fawcett, se emplean para organizar los clasificadores, visualizar, ordenar y seleccionar el mejor rendimiento del modelo desarrollado. (2006)

Según Hosmer, Lemeshow Y Sturdivant (2013), roc es una herramieta que evalua el desempeño de un modelo de clasificación binaria, representa la relación entre los verdaderos positivos y falos positivos para los distintos puntos de corte.

Este de construye de la siguiente manera:

Sensibilidad(verdaderos positivos) :

$$TPR = \frac{VP}{VP + FN}$$

Falsos positivos:

$$FPR = \frac{FP}{FP + VN}$$

Donde:

Vp son los verdaderos positivos

Fp son los falsos positivos.

Vn son los verdaderos negativos

Fn son falsos negativos

6.7.7. **AUC**

Auc por el contrario ayuda a cuantificar el desempeño global del modelo de clasificación binaria, midiendo su capacidad para distinguir entre las observaciones negativas y positivas. Según Fawcett (2006), el área bajo la curva roc es la medida que cuantifica la capacidad del modelo de clasificación para interpretar entre las clases positivas y negativas, con la probabilidad de que el modelo asigne un puntuación mayor a una observación positiva a una negativa sin intervención del umbral de decisión.

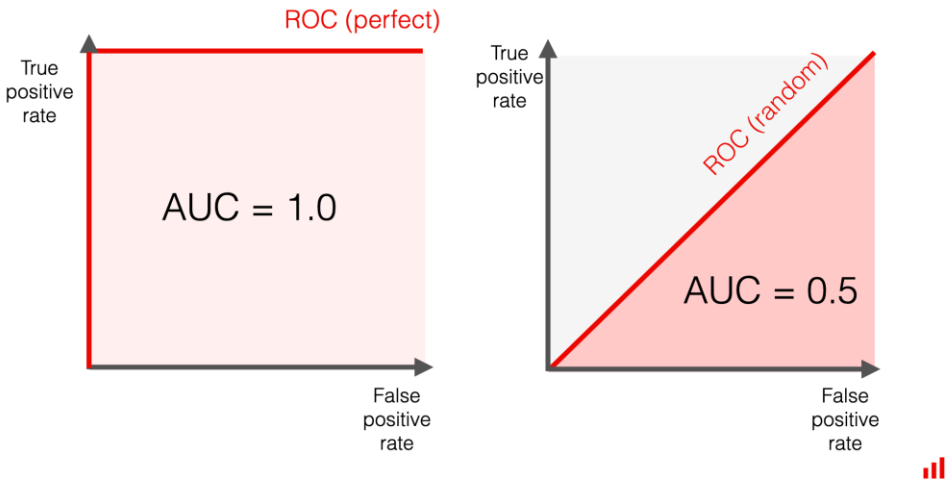


Figura 4: *Comparación entre Curvas ROC & AUC.*

6.8. **Indicadores de Desempeño**

Para optimizar la producción y garantizar la sostenibilidad del cultivo de camarón, es necesario medir indicadores técnicos que permitan evaluar el crecimiento, la eficiencia del alimento, el manejo de densidades y la planificación de la cosecha. Los indicadores principales son: la Ganancia Diaria de Peso (ADG), la Conversión Alimenticia (FCR), la Biomasa y pronóstico de cosecha, generando ventajas competitivas en un mercado que se desarrolla a grandes pasos.

6.8.1. **Ganancia Diaria de Peso (ADG)**

La ganancia diaria de peso (ADG) refleja el incremento promedio de crecimiento, constituyendo un indicador esencial para medir el incremento de la biomasa y por ende el rendimiento de la siembra. Según Boyd y Tucker (2012), el crecimiento eficiente depende directamente de factores ambientales, nutricionales y genéticos.

$$ADG = (Peso\ final - Peso\ inicial) / Número\ de\ días$$

6.8.2. **Conversión Alimenticia (FCR)**

La Conversión Alimenticia (FCR) nos ayuda a determinar la relación entre la cantidad de alimento suministrado y la biomasa obtenida como resultado del crecimiento. Un FCR bajo indica mayor eficiencia productiva y menor impacto ambiental. (Tacon & Metian, 2015)

$$FCR = Alimento\ suministrado\ (kg) / Biomasa\ producida\ (kg)$$

6.8.3. **Biomasa**

La biomasa es el peso total del camarón existente en el sistema de cultivo Este valor sirve para planificar la alimentación, calcular capacidades de carga y prevenir riesgos por sobrepoblación. De acuerdo con FAO (2023), la estimación precisa de biomasa es esencial para la sostenibilidad ambiental del cultivo.

Biomasa = Número estimado de camarones × Peso promedio individual

6.8.4. **Proyección de Cosecha**

La proyección de cosecha consiste en estimar la producción final en un período de tiempo específico. Este indicador integra la información de ADG, biomasa y mortalidad, lo que permite al productor coordinar procesos logísticos y comerciales con mayor precisión (Boyd & Tucker, 2012).

7. MARCO CONCEPTUAL

7.4. **Ciencia de Datos**

Ciencia de datos es el conjunto de dos conceptos que unidos conforman una nueva directriz en el ámbito empresarial, laboral y en general el aprendizaje. Si bien es cierto que ciencia según la Real Academia Española (RAE,2025), La ciencia es un conjunto de conocimientos obtenidos mediante la observación y el razonamiento, sistemático, estructurado y que deduce principios y leyes generales con capacidad predictiva como comprobables experimentalmente.

Así también podemos definir el concepto de datos la Real Academia Española (RAE,2025), Los Datos son la información sobre un tema específico o características que permiten conocer con exactitud información sobre el comportamiento histórico así mismo sirve para deducir las consecuencias de un hecho.

Es así que la ciencia de datos es un conjunto de conceptos y datos empleados como herramientas e históricos que combinados en diferentes campos ayudan a obtener información acerca de comportamientos que se tienen dentro de un campo especifico, sin embargo, se debe tener en cuenta que existen datos que sobre ajustan ciertas medidas es ahí donde interviene la estadística según la Real Academia Española (Real Academia Española, 2025) La estadística es un estudio científico que tiene por objeto la recolección de datos y análisis numéricos relacionados a determinados fenómenos, así como también las con la eclosiones a partir de ellas, basadas en cálculos de probabilidades.

Actualmente las empresas dedicadas a brindar servicios y bienes a sus clientes, buscan maneras de entender los datos de una u otra manera, los pequeños empresarios pueden llegar a medir el comportamiento de sus clientes por la interacción que mantienen, sin embargo, una empresa desarrollada, no puede mantenerse al tanto de las observaciones de sus clientes como también del mismo funcionamiento interno, es ahí donde surge la necesidad de obtener y entender los datos.

7.5. **Machine Learning**

El machine learning o también aprendizaje automático pertenece a una de las ramas de la inteligencia artificial, se enfoca en el desarrollo de modelos y algoritmos que aprenden de un conjunto de datos. En comparación con los sistemas tradicionales basados en reglas rígidas, el ML permite a las máquinas aprender automáticamente a partir de datos. Esta capacidad lo convierte en una herramienta útil para la automatización de procesos.

El objetivo principal del ML es generar modelos que puedan distinguir patrones complejos dentro de un gran volumen de datos. Para lograrlo, estos modelos usan datos históricos conocidos como datos de entrenamiento, lo que permite a los algoritmos conocer el comportamiento de los datos. De esta forma pueden generar predicciones y/o clasificar información nueva de forma precisa.

El ML se divide principalmente en tres tipos: aprendizaje supervisado, no supervisado y de refuerzo. En el contexto de esta investigación se hace énfasis en el aprendizaje supervisado, ya que el modelo de Regresión Logística va a ser enfocada a esta categoría. Este tipo de aprendizaje usa datos etiquetados y estos se encargan de entrenar al algoritmo, permitiendo la clasificación de datos y predicción de resultados.

Dentro del ámbito empresarial, el ML ha tomado gran relevancia debido a su capacidad para optimizar la toma de decisiones en base a un conjunto de datos. Al implementar las técnicas que contiene un aprendizaje automático las empresas pueden optimizar procesos, realizar estrategias empresariales, etc.

7.6. **Clasificación en Machine Learning**

Clasificar, es poder ubicar un objeto en una etiqueta correspondiente, en Machine Learning, el concepto es el mismo. De acuerdo con Velasco (2024) “Clasificar es la acción de ubicar, segregar o de separar los elementos en etiquetas o en grupos correspondientes basándose en las características del objeto”.

Retomando, el anterior comentario cuando trabajamos con algoritmos de clasificación en Machine Learning, es necesario entender que debemos manejar los datos que son etiquetados: a menudo se tratan de categorizar por ejemplo “Fraude” o “No Fraude”. Debemos comprender que los algoritmos de clasificación predicen, pero clases o probabilidades, para eso existen las regresiones. Estos algoritmos, tienden a centrarse en la probabilidad de que K sea parte de C clase existente. Clasificar no es algo fácil, de hecho, los humanos solemos hacerlo diariamente y casi siempre solemos ignorar ciertas características para poder facilitar la acción.

7.7. **Aprendizaje supervisado**

En el campo del aprendizaje automático o machine learning, el aprendizaje supervisado es una técnica fundamental para enseñar a las computadoras. Implica en entrenar modelos utilizando datos cuyas respuestas ya se conocen. En otras palabras, para cada dato de entrada también sabemos cuál debería ser la salida correcta de aquí el nombre “aprendizaje

supervisado” porque le indicamos al modelo cuales son las respuestas correctas mientras aprende.

Por ello, en el proceso de entrenamiento al modelo se le presenta numerosos ejemplos de entradas (como imágenes, textos o números) junto con sus respuestas correctas de este modo el modelo aprende a relacionar las entradas con sus salidas.

Para saber que tan bien lo está haciendo el modelo intenta hacer predicciones, y se le corrige cuando se equivoca. Para medir su rendimiento se utiliza una función de pérdida o error que calcula la diferencia entre lo que el modelo ha predicho y lo que debería haber predicho. Cabe mencionar que entre menos sea esa diferencia mejor está funcionando el modelo.

Una vez entrenado correctamente, el modelo ya puede hacer predicciones con nuevos datos (que nunca ha visto antes) incluso si no conoce la respuesta gracias a lo que ha aprendido con los ejemplos anteriores. Esto es posible porque ha aprendido patrones a partir de los datos anteriores y es capaz de generalizar ese conocimiento a situaciones nuevas.

Por esa razón el aprendizaje supervisado resulta particularmente beneficioso en problemas relacionados con la clasificación y la regresión. En la clasificación, el propósito es otorgar una clasificación o etiqueta a cada entrada, en cambio, en la regresión, el propósito es anticipar un valor numérico.

Dentro del aprendizaje supervisado, los algoritmos de clasificación se dividen en dos categorías: Clasificadores generativos y discriminatorios.

7.8. **Variables**

Para poder entender que es un aprendizaje automatizado, se define lo siguiente: los tipos de variables, que son las continuas y categóricas, ambas explican o describen como es la variable, sin embargo, no definen que dato se deberá emplear en el modelo a desarrollar.

El tipo de variable continua se denomina según Devore, J" una variable continua puede tomar un número infinito de valores dentro de un rango, se mide en una escala numérica donde los intervalos tienen significado real" (2015). Es decir que hace referencia a factores reales que pueden o alteran los datos a cierta magnitud dependiendo de lo que se estudie.

La variable categórica según Agresti. A "es aquella que organiza y clasifica a los datos u objetos en grupos mutuamente excluyentes, no hacen referencia a magnitudes o datos numéricos" estos pueden ser nominales (no tienen orden) u ordinales (con una jerarquía)

Así también debemos identificar el rol de nuestras variables en el modelo, para esto existen dos tipos de conjuntos que son las dependientes e independientes. Los dependientes (y) son aquellos objetivos que se quieren predecir o variables a pronosticar también se los conoce como target u outputs. La variable independiente(x) es usada para poder obtener los pronósticos, que pueden ser las variables que se encuentran en el conjunto de las variables y.

7.9. **Modelos de regresión**

Un modelo de regresión permite analizar la relación entre una variable dependiente respecto a otras variables en conjunto independientes. Su objetivo principal es entender como varía la variable de interés (Y) cuando se modifican una o varias variables explicativas (X). (Pelaez, 2006)

La regresión lineal es un modelo matemático que describe la relación entre una variable dependiente y una independiente. Este tipo de modelo estadístico es útil para hacer proyecciones sobre el futuro y ha sido ampliamente aplicado tanto en el ámbito científico como en el empresarial. Recientemente, también ha cobrado relevancia en el campo del aprendizaje automático. (Gelman & Hill, 2007)

Además de uso predictivo, la regresión lineal resulta eficaz para representar el comportamiento de ciertos sistemas. Cuando se desea modelar una variable numérica a partir de un número limitado de factores explicativos y se busca que el modelo sea fácil de interpretar, la regresión lineal suele ser una opción adecuada para ese propósito. (Fahrmeir, Kneib, Lang, & Marx, 2013)

7.10. **Pendiente**

En los modelos de regresión, la pendiente o en el coeficiente de regresión es un valor numérico que cuantifica la relación entre una variable independiente y la variable dependiente. Específicamente, indica Moreno Ruiz (2019) cuánto se espera que juegue la variable dependiente ante un cambio de una unidad en la variable independiente, manteniendo siempre constantes las demás variables del modelo.

Cada variable independiente en el modelo de regresión múltiple tiene su propia pendiente, lo que permite interpretar el efecto individual de cada predictor. Un claro ejemplo es si se estudian las ventas dependiendo en función de la publicidad y el precio del producto, la pendiente asociada a la variable "publicidad" reflejará el cambio esperado en las ventas por cada unidad adicional invertida en publicidad, suponiendo que el precio se mantiene constante.

El signo de la pendiente también proporciona información más que necesaria: una pendiente positiva indica una relación directa, lo que sería que a un valor más alto de la variable independiente, la dependiente siempre será aún más alta. Según Rodríguez Pérez (2022) una pendiente negativa señala una relación inversa siendo lo mismo que la pendiente positiva, pero desde un lado inverso.

De acuerdo con Díaz (2020), una correcta interpretación de las pendientes permite tomar decisiones estratégicas basadas en evidencia cuantitativa, ya que revela qué factores tienen un impacto significativo en la variable dependiente. Asimismo, la magnitud de la pendiente ofrece una idea de la sensibilidad del modelo ante cambios en la variable explicativa.

En resumen, la pendiente es un componente vital para poder entender y tener claro cómo las variables independientes influyen individualmente sobre la variable de interés en un entorno multivariable.

7.11. **Intercepto**

El intercepto, también conocido como término constante o término independiente en un modelo de regresión, Torres Gómez (2022) dice que representa el valor estimado de la variable dependiente cuando todas las variables independientes tienen un valor igual a cero.

En otras palabras, el intercepto es el punto donde el plano de regresión corta el eje vertical en un espacio multidimensional. Su interpretación práctica depende de hacia donde vaya nuestro problema. Por ejemplo, si se predicen las ventas en función del gasto en marketing y el precio, el intercepto representaría las ventas base esperadas cuando no se ha invertido lo poco o suficiente en marketing y el precio es cero. (González, 2018) expresa que, aunque esta situación no sea realista, el intercepto es necesario para crear el modelo matemático.

En términos algebraicos, el intercepto es β0 en la ecuación general de regresión múltiple. Como indica Díaz (2020), el intercepto no siempre tiene una interpretación práctica, pero su inclusión es necesaria para prevenir sesgos en los coeficientes estimados y para asegurar que el modelo posea el mínimo error cuadrático.

En conclusión, el intercepto es un componente estructural del modelo de regresión que permite establecer el punto de partida de la predicción y contribuye a la precisión del ajuste general del modelo.

Función Sigmoide

La función sigmoide es una de las funciones matematicas con las cuales de puede transformar cualquier valor real en un número comprendido entre 0 y 1. Esto es muy importante dentro de la regresión logistica ya que permite realzar una combinación lineal de variables explicativas en una probabilidad.

7.12. **Preprocesamiento de datos**

El preprocesamiento de datos se define como el conjunto de técnicas aplicadas a los datos brutos con el objetivo de prepararlos para el análisis estadístico y el modelo predictivo. Esta etapa permite mejorar la calidad de los datos, reducir errores y asegurar que los algoritmos utilizados funcionen de manera adecuada, ya que muchos modelos son sensibles a la forma y escala de las variables de entrada. (Hastie, Tibshirani, & Friedman, The elements of statistical learning, 2009)

7.13. **Variables numéricas**

Las variables numéricas son aquellas que representan cantidades medibles y pueden expresarse mediante valores numéricos. Estas variables permiten la aplicación directa de operaciones matemáticas y estadísticas, lo que las convierte en un componente esencial para la estimación de modelos estadísticos y de aprendizaje automático. Pueden clasificarse en variables discretas y continuas, dependiendo de los valores que pueden tomar. (James et al., 2021)

7.14. **Normalización**

La normalización es un método de escalado que transforma los valores de una variable a un rango especifico, generalmente entre 0 y 1. Este procedimiento es útil cuando los datos no siguen una distribución normal y cuando se desea mantener la relación proporcional entre los valores originales, evitando que las diferencias de escala afecten el desempeño del modelo. (James et al.,2021)

7.15. **Estandarización**

La estandarización es una técnica de escalado que consiste en centrar las variables en una media igual a cero y una desviación estándar igual a uno. Esta transformación facilita la comparación entre variables y mejora la estabilidad numérica de los algoritmos de optimización, siendo ampliamente utilizada en modelos estadísticos y métodos basados en máxima verosimilitud. (James, Witten, Hastie, & Tibshirani, An introduction to statistical learning: With applications in R, 2021)

7.16. **Odds Proporcionales**

Los odds proporcionales según indica Agresti (2010), es un modelo que se enfoca en utilizar en modelar la probabilidad acumulada de una variable dependiente con sus respectivas categorías ordenadas, bajo la premisa de que los efectos de estas variables sean constantes.

7.17. **Entrenamiento**

El entrenamiento de datos es el proceso mediante el cual un modelo estadístico o de aprendizaje automático ajusta sus parámetros internos utilizando información observada con el objetivo de aprender los patrones y relaciones existentes entre las variables. Durante esta fase, el modelo minimiza un error o función de perdida para mejorar su capacidad de realizar predicciones precisas (Hastie, Tibshirani & Friedman, 2009).

7.18. **Prueba de datos**

Prueba de datos es el proceso mediante se evalúa el desempeño de un modelo previamente entrenado, utilizando información que no intervino en el ajuste de sus parámetros. Su finalidad es medir la capacidad del modelo para generalizar y realizar predicciones correctas en datos nuevos, estimando su rendimiento real (Hastie, Tibshirani & Friedman, 2009).

Tipo de alimentacion

Si bien es cierto que dentro del mundo camaronero existen diferentes formas de alimentar a los camarones, dentro de la empresa estudiada se tienen presentes dos tipos de alimentación: por hidrófono o alimentación automática. A su vez, tienen una forma establecida de alimentación, la cual consiste en ciertos lapsos de tiempo programados para poder dispersar el alimento del animal.

Ambas cumplen la misma función; sin embargo, son completamente diferentes.

La alimentación automática por hidrófono se encarga de dispersar la comida automáticamente, siempre y cuando el hidrófono capte que el camarón está rebuscando comida.
Por el contrario, el Timing es más común y simple en el proceso: se establecen horarios en los cuales se debe brindar alimento al camarón, independientemente de si se percibe que el animal necesita alimentarse.

8. MARCO LEGAL

El presente trabajo, hace uso de información brindada por parte de una corporación centralizada en el sector camaronero, por los cuales se rige el proceso de dicha investigación bajo los lineamientos de La superintendencia de protección de datos. Aunque este proyecto de investigación no contempla el tratamiento de datos personales, puesto que trabaja con información productiva y operativa, sí emplea datos privados proporcionados por dicha empresa, por lo que se considera pertinente establecer el siguiente marco legal y ético para el tratamiento de dichos datos.

En este caso rigiéndonos bajo la Ley Orgánica de Protección de Datos Personales (2021) la cual constituye la referencia normativa nacional, especialmente en lo establecido en su Artículo 2 el cual excluye de su ámbito de aplicación a los datos anonimizados o aquellos que no permitan identificar a una persona natural. No obstante, los principios establecidos en el Artículo 10 de dicha Ley se adoptan como marco orientador en el manejo de datos dentro del desarrollo de esta investigación, debido a su aporte a la gobernanza, seguridad y gestión responsable de la información.

a) **Juridicidad**: La información proporcionada por la empresa será tratada dentro de un marco legal, respetando todas las normas vigentes, de manera que su uso sea legitimo y acorde con los fines investigativos del proyecto (Ley Orgánica de Protección de Datos Personales, 2021, art. 10).

b) **Lealtad**: Los datos se manejarán con honestidad y respeto hacia la empresa que los proporciona, evitando cualquier practica que puede causar perjuicio, malentendidos o un uso distinto al comunicado originalmente (Ley Orgánica de Protección de Datos Personales, 2021, art. 10).

c) **Transparencia**: La empresa colaboradora tendrá conocimiento claro del propósito del tratamiento de los datos, procesos involucrados y alcance de su uso durante el desarrollo del modelo, garantizando claridad en cada fase del proyecto (Ley Orgánica de Protección de Datos Personales, 2021, art. 10).

d) **Finalidad**: Los datos serán empleados únicamente con el propósito de diseñar, entrenar y validar el modelo, orientado a optimizar procesos productivos en el cultivo de camarón, sin destinarlos a actividades ajenas a la investigación (Ley Orgánica de Protección de Datos Personales, 2021, art. 10).

e) **Pertinencia y minimización**: Solo se solicitará la información estrictamente necesaria para el funcionamiento del modelo, evitando recolectar datos que no aporten valor al análisis o a la predicción dentro del estudio (Ley Orgánica de Protección de Daros Personales, 2021, art.10).

f) **Proporcionalidad**: La cantidad y tipo de datos se ajustara a lo que realmente requiere el proyecto, evitando solicitar o procesar información que supere las necesidades reales del desarrollo (Ley Orgánica de Protección de Datos Personales, 2021, art.10).

g) **Confidencialidad**: La información compartida será manejada de forma reservada y protegida contra accesos o divulgaciones no autorizadas, reconociendo importancia (Ley Orgánica de Protección de Datos Personales, 2021, art. 10).

h) **Calidad y exactitud**: Se emplearán datos verificados y depurados, entendiendo que la precisión de la información es esencial para obtener resultados confiables y minimizar el margen de error del modelo (Ley Orgánica de Protección de Datos Personales, 2021, art. 10).

j) **Seguridad de los datos**: Se establecerán medidas tecnológicas y administrativas que limiten los riesgos de pérdida, alteración o acceso indebido a la información, especialmente tratándose de herramientas basadas en plataformas digitales (Ley Orgánica de Protección de Datos Personales, 2021, art. 10).

k) **Responsabilidad proactiva**: Se documentarán los procedimientos asociados al uso de datos para garantizar trazabilidad, control y la capacidad de demostrar buenas prácticas en su tratamiento (Ley Orgánica de Protección de Datos Personales, 2021, art. 10).

l) **Aplicación favorable al titular**: Cualquier situación no prevista en este proceso será resuelta priorizando el beneficio y la protección de la empresa que suministra los datos, manteniendo un enfoque preventivo respecto del manejo de la información (Ley Orgánica de Protección de Datos Personales, 2021, art. 10).

9. METODOLOGIA

9.1. Tipo de investigación

En el presente capítulo se describen los métodos empleados para el diseño del modelo que nos permita determinar la tasa de supervivencia del camarón a partir de datos de producción, de la Empresa XYZ. El estudio se desarrolla bajo el principio de confidencialidad y mediante la aplicación de técnicas de ciencia de datos, se busca identificar los factores que influyen en la supervivencia del camarón, con el fin de optimizar los procesos productivos y apoyar la toma de decisiones estratégicas dentro de la empresa.

La presente investigación es de tipo cuantitativa, ya que se basa en el análisis de datos numéricos provenientes de los registros productivos de la Empresa XYZ. Además, tiene un enfoque descriptivo, explicativo, debido a que busca identificar y explicar los factores que influyen en la tasa de supervivencia del camarón. Para el análisis de clasificación se emplea un modelo de aprendizaje supervisado, como lo es la regresión logística, el cual permite estimar la probabilidad de supervivencia en función de diversas variables productivas.

9.2. Datos

9.2.1. Organización y desglose de datos.

Al inicio del proyecto presente, se contó con una base de datos de aproximadamente 430 cosechas distintas divididas en alrededor de 6000 semanas las cuales representan entradas distintas de datos, recalcando dichos datos forman parte de información confidencial de producción brindada por la empresa XYZ, los datos se encuentran almacenados en Excel, y son importados al entorno de Python para su procesamiento y análisis.

Posterior a varios procesos de limpieza y depuración de los datos terminamos reduciendo dicha base de 6000 semanas a un total de 109 cosechas, divididas en 1400 semanas, a continuación, el desglose de los datos y las variables utilizadas.

Variable	Definición conceptual	Definición operacional	Dimensiones	Indicadores	Instrumentos	Tipo de variable
Piscina	Unidad física de cultivo donde se desarrolla el ciclo productivo del camarón	Código o número asignado a cada piscina	Identificación	Número de piscina	Registro productivo de la finca	Categórica nominal
Zona	Área geográfica dentro de la finca camaronera	Zona asignada según ubicación interna	Ubicación	Zona productiva	Registro de producción	Categórica nominal
Finca	Unidad productiva donde se realiza el cultivo	Nombre o código de la finca	Ubicación	Finca productiva	Registro administrativo	Categórica nominal
HA	Superficie de cultivo	Hectáreas de cada piscina	Área	Número de hectáreas	Registro técnico	Numérica continua
Laboratorio	Centro de origen de las postlarvas	Laboratorio proveedor	Origen	Nombre del laboratorio	Registro de compra	Categórica nominal
Fecha de siembra	Momento inicial del ciclo productivo	Fecha exacta de siembra	Tiempo	Día, mes y año	Registro productivo	Numérica continua
Cod Gen	Código genético del camarón	Identificador del lote genético	Genética	Código genético	Registro del laboratorio	Categórica nominal

Densidad	Número de organismos sembrados por unidad de área	Animales por hectárea	Población	Animales/HA	Registro de siembra	Numérica continua
Peso inicial gramos	Peso promedio al inicio del cultivo	Peso promedio en gramos	Biomasa	Gramos	Muestreo biométrico	Numérica continua
Días de precría	Tiempo previo al engorde	Número de días	Tiempo	Días	Registro técnico	Numérica continua
Peso semana n	Incremento de peso del camarón durante la semana	Peso promedio en gramos en la semana n	Crecimiento	Gramos/semana	Muestreo biométrico semanal	Numérica continua
Supervivencia semana n	Proporción de camarones vivos en la semana	Porcentaje de supervivencia semanal	Supervivencia	% de organismos vivos	Registro productivo	Numérica continua
Tipo de alimento	Tipo de dieta suministrada al camarón	Clasificación del alimento	Alimentación	Tipo de alimento	Registro de alimentación	Categórica nominal
HP/HA	Capacidad de aireación por área	Caballos de fuerza por hectárea	Aireación	HP por HA	Registro técnico	Numérica continua
Animales totales	Número total de organismos al final del ciclo	Conteo final	Producción	Número de camarones	Registro productivo	Numérica continua
Peso final gramos	Peso promedio final del camarón	Peso promedio en gramos	Biomasa	Gramos	Muestreo biométrico	Numérica continua
Días totales	Duración total del cultivo	Número de días	Tiempo	Días	Registro productivo	Numérica continua

Biomasa	Peso total producido	Biomasa total en kilogramos	Producción	Kg totales	Cálculo productivo	Numérica continua
Kg * HA totales	Producción por área	Kilogramos por hectárea	Rendimiento	Kg/HA	Cálculo productivo	Numérica continua
Promedio crecimiento semanal	Incremento promedio de peso	Promedio de crecimiento en gramos por semana	Crecimiento	g/semana	Cálculo estadístico	Numérica continua

9.3. Preparación de datos.

9.3.1. Umbral de supervivencia para la clasificación.

En este estudio, la supervivencia del camarón se capturó inicialmente como una variable cuantitativa continua, en porcentaje, que representa la cantidad de animales vivos al final del ciclo productivo. Pero como lo que se pretende analizar son los factores que aumentan o disminuyen la probabilidad de tener un buen desempeño productivo, se consideró metodológicamente pertinente convertir esta variable en una variable dicotómica.

Esta adaptación permite moldear la supervivencia en términos probabilísticos, en la línea de un modelo de regresión logística, donde la variable dependiente debe ser un suceso dicotómico (ocurrencia o no de un evento).

Antes de establecer el criterio de clasificación, se hizo un análisis estadístico descriptivo de la tasa de supervivencia, calculando medidas de tendencia central y dispersión como la media, mediana y desviación estándar. Los resultados mostraron una alta variabilidad en los niveles de supervivencia entre ciclos productivos (desviación estándar alta) y la presencia de valores extremos asociados a eventos productivos negativos.

En este caso, el uso de la media como punto de corte clasificatorio era estadísticamente inapropiado, por ser muy susceptible a valores extremos y distribuciones sesgadas. En su lugar, se tomó como referencia la mediana de la tasa de supervivencia, ya que ésta es estadísticamente robusta y refleja mejor el comportamiento usual del sistema productivo.

La mediana separa la muestra en dos grupos iguales, clasificando en forma balanceada las observaciones de alta y baja supervivencia, lo que estabiliza los coeficientes estimados en el modelo logístico y mejora la capacidad discriminatoria del modelo.

De esta manera, se definió la variable binaria de supervivencia de la siguiente forma:

$$Y = \begin{cases} 1 & \text{si la tasa de supervivencia es mayor o igual a la mediana} \\ 0 & \text{si la tasa de supervivencia es inferior a la mediana} \end{cases}$$

Esta transformación asegura que la variable dependiente sea un criterio objetivo y medible en términos de datos, evitando la imposición de umbrales subjetivos y mejorando la robustez estadística del análisis. Además, la variable resultante puede expresar los resultados del modelo en términos de probabilidad de éxito productivo, lo que facilita su uso en la toma de decisiones en la camaronicultura.

9.3.2. Calculo promedio crecimiento semanal

Con el objetivo de estimar el crecimiento del camarón de manera precisa y metodológicamente sólida, se analizó el incremento de peso semanal expresado en gramos. Este enfoque permite capturar la dinámica real del crecimiento biológico a lo largo del ciclo productivo, superando las limitaciones de análisis basados únicamente en el peso final.

Paso 1: Cálculo del crecimiento semanal en gramos

El crecimiento semanal se definió como la diferencia entre el peso promedio del camarón en semanas consecutivas, de acuerdo con la siguiente expresión:

$$G_i = P_i - P_{i-1}$$

donde: G_i representa el crecimiento semanal en gramos en la semana (i), P_i corresponde al peso promedio en gramos en la semana (i), P_{i-1} corresponde al peso promedio en gramos de la semana anterior.

Este cálculo se realizó para cada una de las semanas del ciclo productivo, generando una serie de valores de crecimiento semanal.

Paso 2: Cálculo de la media muestral del crecimiento semanal

Una vez obtenidos los valores de crecimiento semanal, se calculó la media muestral como medida inicial de tendencia central:

$$\bar{G} = \frac{1}{n}\sum_{i=1}^{n} G_i$$

Donde: $\bar{G}$ es la media del crecimiento semanal, n es el número total de semanas analizadas.

No obstante, debido a la alta variabilidad observada entre semanas y al tamaño reducido de la muestra (menos de 30 observaciones), la media no fue utilizada como estimación final del crecimiento promedio.

Paso 3: Cálculo de la desviación estándar muestral

Con el fin de cuantificar la dispersión del crecimiento semanal alrededor de la media, se calculó la desviación estándar muestral:

$$s = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(G_i - \bar{G})^2}$$

donde: s representa la desviación estándar del crecimiento semanal.

Este estadístico refleja la variabilidad del crecimiento entre las distintas semanas del ciclo productivo.

Paso 4: Cálculo del error estándar de la media

Posteriormente, se estimó el error estándar de la media, el cual mide la precisión de la media muestral como estimador del parámetro poblacional.

$$SE = \frac{s}{\sqrt{n}}$$

Este valor incorpora tanto la variabilidad muestral como el tamaño de la muestra.

Paso 5: Definición del nivel de significancia y grados de libertad

Dado el carácter inferencial del análisis, se estableció un nivel de significancia del 5%:

$$\alpha = 0.05$$

Asimismo, se definieron los grados de libertad de la distribución t de Student como:

$$gl = n - 1$$

Paso 6: Obtención del valor crítico de la distribución t de Student

Considerando que el tamaño de la muestra es menor a 30 y que la desviación poblacional es desconocida, se utilizó la distribución t de Student. El valor crítico se obtuvo para un intervalo bilateral (dos colas), de la siguiente forma:

$$t_{\alpha/2,n-1}$$

Este valor se extrajo de la tabla de la distribución t correspondiente al nivel de significancia y a los grados de libertad definidos.

Paso 7: Cálculo del margen de error

El margen de error del estimador se calculó como:

$$ME = t_{\alpha/2,n-1} \times SE$$

Este término representa la amplitud de la incertidumbre asociada a la estimación de la media del crecimiento semanal.

Paso 8: Construcción del intervalo de confianza del crecimiento semanal

A partir de los valores anteriores, se construyó el intervalo de confianza bilateral para el crecimiento semanal promedio:

$$IC = [\bar{G} - ME,\ \bar{G} + ME]$$

Este intervalo contiene, con un 95% de confianza, el verdadero crecimiento semanal promedio de la población.

Paso 9: Selección del límite inferior como estimación robusta del crecimiento semanal

En lugar de utilizar la media muestral como valor representativo del crecimiento semanal, se adoptó un enfoque estadístico robusto y conservador, seleccionando el límite inferior del intervalo de confianza:

$$G = \bar{G} - ME$$

Este valor representa el crecimiento mínimo esperado del camarón bajo condiciones normales de producción, incorporando explícitamente la variabilidad observada y reduciendo el riesgo de sobreestimación.

Emplear el límite inferior del intervalo de confianza, en lugar el promedio aritmético, permite obtener una estimación del crecimiento semanal mas conservadora y cercana a la realidad. Esta elección resulta pertinente en sistemas productivos con elevaba variabilidad biológica y muestras de tamaño reducido, como ocurre en el cultivo de camarón.

De igual manera, este criterio metodológico aporta mayor solidez a los análisis posteriores, al minimizar la incorporación de estimaciones excesivamente optimistas en indicadores productivos derivados, como la biomasa total, el rendimiento por hectárea y los estudios de supervivencia.

9.4. Python

9.4.1. Empleo de Python

Python es un lenguaje de programación de alto nivel muy utilizado en ciencia de datos, análisis estadístico y aprendizaje automático debido a su sintaxis legible, versatilidad y la disponibilidad de bibliotecas especializadas. En esta investigación, Python es la herramienta para manipular y analizar los datos productivos confidenciales que la Empresa XYZ nos proporcionó. Estos datos incluyen información sobre el cultivo de camarón, tales como ganancia diaria de peso (ADG), conversión alimenticia (FCR), biomasa, densidad de siembra y mortalidad, datos para calcular la tasa de supervivencia.

Con bibliotecas como Pandas, NumPy, Python puede importar datos desde archivos Excel o CSV a estructuras de datos para realizar análisis estadísticos. En esta etapa se limpian

y preparan los datos, manejando valores faltantes, identificando valores atípicos y convirtiendo variables categóricas en numéricas. Estas etapas garantizan la homogeneización de la información y la confiabilidad de los resultados (McKinney, 2010)

Además, Python permite la estandarización de las variables independientes mediante escalamiento, lo cual es útil cuando se están analizando variables productivas que se miden en unidades distintas (gramos, kilogramos, porcentajes, etc.). Este proceso estabiliza numéricamente el modelo de regresión logística y proporciona una mejor estimación de sus coeficientes. Además, el lenguaje ofrece herramientas para generar gráficas estadísticas que permitan visualizar los resultados e exportar la información procesada en formatos legibles para revisarlos y analizarlos internamente en la Empresa XYZ (Hunter, 2007)

9.4.2. Métodos

El estudio de la tasa de supervivencia del camarón se hace con métodos de ciencia de datos en Python. Primero, los datos se limpian y preparan, eliminando valores atípicos, imputando datos faltantes y eligiendo variables apropiadas. Luego, los datos se dividen en conjuntos de entrenamiento y prueba para desarrollar y evaluar el modelo. El método estadístico utilizado es la regresión logística, que modela la probabilidad de supervivencia del camarón en función de varias variables explicativas.

9.5. Limpieza en Python

9.5.1. Detección de valores faltantes

df.isnull().sum()

Como primer paso del preprocesamiento de los datos, se identificó que el conjunto de datos contenía datos faltantes. La identificación de datos faltantes es un paso esencial en

cualquier análisis estadístico, ya que los datos faltantes pueden introducir sesgos en la estimación de parámetros y afectar la fiabilidad de los modelos predictivos.

Reconocer estos valores es importante porque la eliminación simple de las filas con datos perdidos puede disminuir el tamaño muestral efectivo y, por lo tanto, la potencia estadística del modelo, en particular en estudios con tamaños de muestra moderados.

9.5.2. Detección de valores atípicos (outliers)

```
#Deteccion de Outlayers o valores averrantes
#Creamos la funcion
#Primero defino que variables voy a usar
numericas = ["SUPER FINAL %"]
#le digo que los outliers, lo valide entre llaves {}, todo lo que esta en llaves significa que está fuera de lo normal.
outliers = {}
#ahora los cuartiles:
for col in numericas:
  Q1 = df[col].quantile(0.25)
  Q3 = df[col].quantile(0.75)
  IQR = Q3 - Q1
  lower_bound = Q1 - 1.5 * IQR
  upper_bound = Q3 + 1.5 * IQR
  outliers[col] = df[(df[col] < lower_bound) | (df[col] > upper_bound)]
  print('Cantidad de outliers detectados por variable')
  print(outliers)
```

Una vez identificados los valores perdidos, se identificaron los valores atípicos o aberrantes. En los datos productivos acuícolas, los valores atípicos pueden surgir por errores

de medición, errores en el registro de la información o eventos extremos de producción, como enfermedades o problemas operacionales.

La identificación de estos valores se realizó utilizando métodos de estadística descriptiva y herramientas de visualización, específicamente diagramas de caja (boxplots). Este método se basa en el rango intercuartílico (IQR), que viene dado por::

$$IQR = Q_3 - Q_1$$

donde Q_1y Q_3 son el primer y tercer cuartil de la distribución, respectivamente.

Un valor es atípico si se ajusta a los criterios definidos a partir de este rango intercuartílico.

$$x < Q_1 - 1.5 \cdot IQR \text{o} x > Q_3 + 1.5 \cdot IQR$$

Este criterio se ajusta a un procedimiento no paramétrico, ya que no exige normalidad en los datos, lo cual es coherente con la naturaleza biológica y productiva de las variables analizadas.

9.5.3. Visualización de outliers mediante boxplots

La visualización de valores atípicos se realizó mediante boxplots, los cuales permiten representar gráficamente la mediana, los cuartiles y los posibles valores extremos de cada variable, no dependiendo de supuestos de normalidad, siendo un enfoque robusto frente a distribuciones asimétricas, y siendo especialmente útil para identificar rápidamente dispersiones excesivas.

Esta visualización sirvió como criterio diagnóstico previo a la aplicación de técnicas de corrección de datos.

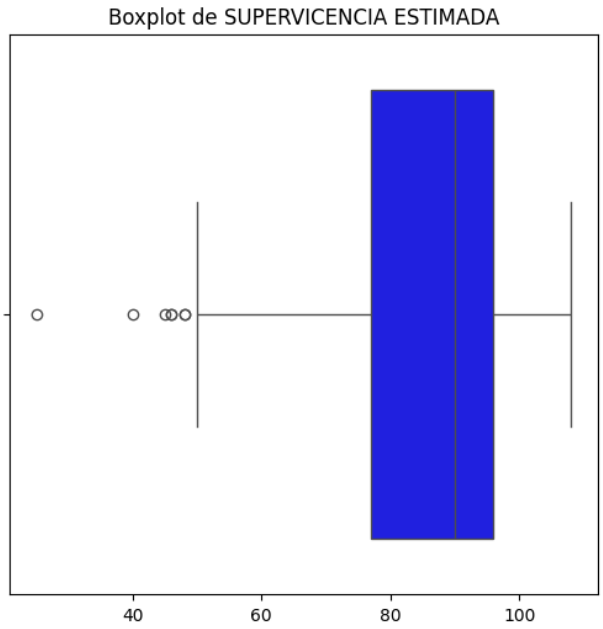


Figura 5 *Boxplot Supervivencia*

9.5.4. Reemplazo de valores atípicos mediante la mediana.

df['SUPER FINAL %'] = df['SUPER FINAL %'].fillna(round(df['SUPER FINAL %'].median(),1))

Para variables numéricas se optó por reemplazar los valores atípicos identificados por la mediana de cada variable. Esta decisión responde a un enfoque de estadística robusta, evitando el uso de medidas clásicas sensibles a valores extremos, como la media aritmética.

La mediana, definida como el percentil 50 de la distribución, se caracteriza por su resistencia a valores aberrantes y se expresa formalmente como:

$$\tilde{x} = \begin{cases} x_{\left(\frac{n+1}{2}\right)} & \text{si } n \text{ es impar} \\ \dfrac{x_{\left(\frac{n}{2}\right)} + x_{\left(\frac{n}{2}+1\right)}}{2} & \text{si } n \text{ es par} \end{cases}$$

Este procedimiento reduce la influencia de observaciones extremas sin eliminar información relevante del conjunto de datos, preservando el tamaño muestral y mejorando la estabilidad de los modelos posteriores.

9.5.5. Corrección de errores de tipeo.

#Analizamos la variable Cod Genetico

df['COD GEN'].unique()

#Corregimos error de tipeo

df['COD GEN'] = df['COD GEN'].str.strip().str.capitalize()

Se identificaron inconsistencias en los datos atribuibles a errores de digitación o tipeo, particularmente en variables categóricas como lo son los **Codigos geneticos**. Estos errores generan categorías artificiales que fragmentan la información y distorsionan el análisis.

La corrección de estos errores permite garantizar la coherencia semántica de las variables categóricas y evita la creación de niveles irreales que podrían afectar tanto el análisis descriptivo como el ajuste del modelo de regresión logística.

Posterior a dichos procesos de limpieza de datos procedimos a descargar la nueva base de datos, ahora limpia para recurrir a procesos descriptivos para apreciar mejor como se encuentran organizados nuestros datos.

9.6. Estadistica descriptiva.

9.6.1. Variables categóricas

freq_abs = df['FINCA'].value_counts(),

El primer paso es calcular la frecuencia absoluta Cuenta cuántas observaciones pertenecen a cada categoría.

$$f_i = \sum_{j=1}^{n} I(x_j = c_i)$$

donde: $c_i$es una categoría específica,

$I(\cdot)$es la función indicadora.

Posteriormente calcular las Frecuencia relativa,

*freq_rel = df['FINCA'].value_counts(normalize=True) * 100*

Con la función counts, contamos cuántas veces aparece cada valor único (frecuencia absoluta). normalize=True: Este es el truco clave. En lugar de darte el conteo (1, 2, 10...), divide cada conteo por el total de filas. El resultado es un número entre 0 y 1 y al multiplicarlo * 100: Convierte esa proporción en un porcentaje (de 0 a 100%). La frecuencia relativa te dice qué porcentaje del total representa cada grupo

Calculamos así mismo las frecuencias acumuladas de ambas, relativa y absoluta, finalmente creamos las tablas unificadas, haciendo uso de la función round para para trabajar con decimales.

```
aFINCA = pd.DataFrame({'Categoría': freq_abs.index,
                       'Frecuencia Absoluta': freq_abs.values,
                       'Frecuencia Relativa (%)': freq_rel.round(2).values,
                       'Frecuencia Acumulada': freq_acum.values,
                       'Frecuencia Relativa Acumulada (%)': freq_rel_acum.round(2).values})
```

Creamos gráficos para poder visualizar mejor nuestra tabla.

Repetimos dicho proceso para el resto de las variables categóricas, para poder representar de una manera la distribución de los datos;

```
#Variable Alimentacion
#Paso 1. Calcular la frecuencia absoluta
freq_absALI = df['TIPO DE ALIMENTO'].value_counts()
#Paso 2. Calcular la frecuencia relativa
freq_relALI = df['TIPO DE ALIMENTO'].value_counts(normalize=True)*100
#Paso 3. Calcular la frecuencia acumulada
freq_acumALI = freq_absALI.cumsum()
#Paso 4. Calcular la frecuencia relativa acumulada
freq_rel_acumALI = freq_relALI.cumsum()
# PASO 5. CREAMOS LA TABLA UNIFICADA / ROUND() ES PARA QUE PONGA DECIMALES PARA HACERLO MAS REALISTA
tablaALI = pd.DataFrame({'Categoría': freq_absALI.index,
                         'Frecuencia Absoluta': freq_absALI.values,
                         'Frecuencia Relativa (%)': freq_relALI.round(2).values,
                         'Frecuencia Acumulada': freq_acumALI.values,
                         'Frecuencia Relativa Acumulada (%)': freq_rel_acumALI.round(2).values})
#Mostramos tabla
display(tablaALI)
```

9.6.2. Variables numéricas NO agrupadas

Son datos cuantitativos continuos NO agrupados, es decir; cada observación se analiza de forma individual

Primero, calcular la frecuencia absoluta, manteniendo el origen (index), calculamos la frecuencia relativa, y la acumulada de ambas, creamos una tabla donde se puede apreciar las frecuencias y utilizando librerías para graficar, expresamos los datos de dichas tablas de una manera mejor traducida y clara para la futura interpretación.

```
freq_absDT = DIAS.value_counts().sort_index()
freq_relDT = (freq_absDT / len(DIAS) * 100). round(2)
freq_acumDT = freq_absDT.cumsum()
freq_rel_acumDT = freq_relDT.cumsum().round(2)
```

9.6.3. Variables numéricas agrupadas

1. Calcular el numero de clases con la regla de Sturges, porque no tenemos N definido

```
n= len(SupervivenciaF)
k= int(np.ceil(1 + 3.322 * np.log10(n)))
```

Es la forma estándar de decidir cuántas "clases" debe tener tu histograma para que no se vea ni muy saturado ni muy vacío. Al usar np.ceil, estás redondeando hacia arriba al entero más cercano, lo cual es correcto para asegurar que cubres todos los datos.

```
2. Crear intervalos con 2 decimales
min_val = SupervivenciaF.min()
max_val = SupervivenciaF.max()
bins = np.linspace(min_val, max_val, k+1)
```

bins = np.round(bins, 2)

3. Agrupampos las clases

clases = pd.cut(SupervivenciaF, bins=bins, right=False)

freq_absSF = clases.value_counts().sort_index()

*freq_relSF = (freq_absSF / len(SupervivenciaF) * 100). round(2)*

freq_acumSF = freq_absSF.cumsum()

freq_rel_acumSF = freq_relSF.cumsum().round(2)

Matemáticamente, el procedimiento inicia con:

Rango de la variable

$$R = x_{\max} - x_{\min}$$

donde: x_{max} *es el valor máximo observado,*

x_{min} *es el valor mínimo observado.*

Número de clases

El número de clases (k) se determina siguiendo criterios estadísticos clásicos como la regla de Sturges:

$$k = 1 + 3.322 \log_{10}(n)$$

donde:

$n$es el número total de observaciones.

Este criterio busca un equilibrio entre:

Excesiva agregación (pérdida de información),

Excesiva desagregación (ruido estadístico).

En términos metodológicos, el código implementa este principio al definir un número razonable de intervalos que permita capturar la forma de la distribución sin sobreajustarla.

Amplitud de clase

Una vez definido el número de clases, se calcula la amplitud de cada intervalo:

$$A = \frac{R}{k}$$

Esta amplitud define el ancho de cada clase y garantiza que los intervalos cubran todo el dominio de la variable analizada.

Asignación de observaciones a clases (frecuencias)

Frecuencia absoluta

Cada observación $x_i$es asignada a un intervalo específico. El número de observaciones dentro de cada clase constituye la frecuencia absoluta (f_i).

Desde el punto de vista del código, este paso se implementa mediante funciones de agrupación (cut, groupby o equivalentes), que asignan automáticamente cada valor a su intervalo correspondiente.

Matemáticamente:

$$f_i = \sum I(x_j \in \text{Clase } i)$$

donde $I(\cdot)$es una función indicadora que vale 1 si la condición se cumple y 0 en caso contrario.

4. Cálculo de medidas descriptivas a partir de datos agrupados

Una vez construida la tabla de frecuencias, las medidas descriptivas se calculan utilizando los puntos medios de cada clase.

Marca de clase

La marca de clase (m_i) se define como:

$$m_i = \frac{L_i + U_i}{2}$$

donde:

$L_i$es el límite inferior del intervalo,

$U_i$es el límite superior del intervalo.

Replicamos el mismo proceso para obtener de igual manera los descriptivos y posteriormente realizar el grafico de distintas variables numérica de frecuencia agrupadas como lo son, Biomasa, Peso Inicial Gramos, Peso Final Gramos, Densidad, Animales Totales, Dias De Precia, Kg*Ha Totales Y Promedio De Crecimiento Semanal.

9.7. Regresión logística

La regresión logística es un modelo estadístico perteneciente a los métodos de aprendizaje supervisado, ampliamente utilizado para la clasificación binaria y la estimación de probabilidades. En el contexto de la presente investigación, se emplea para estimar la probabilidad de supervivencia del camarón durante el ciclo productivo, considerando como

variable dependiente un resultado dicotómico que representa la supervivencia o no supervivencia del lote productivo.

Estadísticamente, la variable dependiente (Y) se modela como una variable aleatoria de Bernoulli, la cual se define como:

$$Y_i \sim Bernoulli(p_i)$$

donde $p_i = P(Y_i = 1)$ es la probabilidad de supervivencia del camarón en la observación (i).

La regresión logística no supone una relación lineal entre las variables explicativas y la variable dependiente, sino entre las variables explicativas y el logaritmo de la razón de probabilidades (logit).

Matemáticamente, el modelo se expresa como:

$$\log\left(\frac{p_i}{1-p_i}\right) = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \cdots + \beta_n x_{ni}$$

donde:

• β_0 es el intercepto del modelo,

• β_j representa el efecto de la variable explicativa x_j sobre el logit de la probabilidad de supervivencia.

Para obtener valores de probabilidad comprendidos entre 0 y 1, se aplica la función logística:

$$pi = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \cdots + \beta_n x_{ni})}}$$

Las variables independientes del modelo fueron indicadores productivos y de manejo, como el promedio semanal de crecimiento (ADG), la conversión alimenticia (FCR), la biomasa total, la densidad de siembra y el consumo de alimento. La incorporación de estas variables permite medir cómo los factores productivos afectan la probabilidad de supervivencia del camarón.

Los coeficientes estimados se interpretan en términos de odds ratios, definidos como e^{β_j}, que representan el cambio proporcional en las probabilidades de supervivencia por cada unidad adicional en la variable explicativa asociada. Por ejemplo, un coeficiente negativo de conversión alimenticia (FCR) señala que mientras más alimento se necesite para producir biomasa, menos posibilidades de supervivencia tendrá el camarón.

La estimación de los parámetros del modelo se lleva a cabo por máxima verosimilitud, que consiste en maximizar la probabilidad de observar los datos de la muestra dados los parámetros del modelo. La función de verosimilitud es:

$$L(\beta) = \prod_{i=1}^{n} P_i^{y_i}(1 - P_i)^{1-y_i}$$

La validación del modelo se realiza mediante la separación del conjunto de datos en muestras de entrenamiento y prueba, lo que permite evaluar su capacidad predictiva sobre observaciones que no fueron consideradas durante el proceso de ajuste. El desempeño del modelo se analiza a través de distintas métricas, entre ellas la matriz de confusión, la exactitud, la sensibilidad y la curva ROC junto con su respectivo valor AUC. Estas herramientas permiten cuantificar (Fawcett, 2006; Hosmer, Lemeshow & Sturdivant, 2013; Pedregosa et al., 2011).

9.8. Librerías Implementadas en Regresión Logística

9.8.1. Pandas

La librería Pandas se emplea para la carga, organización y manipulación de datos estructurados. En el desarrollo del modelo de regresión logística, facilita la importación de información desde archivos externo y su almacenamiento en estructuras tipo *DataFrame,* lo que simplifica la selección de variables independientes y dependientes. Asimismo, permite realizar procesos de limpieza, transformación y ordenamiento de los datos así como la elaboración de tablas finales con los resultados del modelo, las cuales pueden ser exportadas por su posterior (McKinney, 2010).

9.8.2. **Numpy**

Numpy es la base para la computación numérica y el manejo de arreglos multidimensionales. En la regresión logística apoya en los cálculos matemáticos necesarios para el procesamiento de datos, en especial para el escalamiento de variables y el cálculo de probabilidades. Su aplicación garantizó alto rendimiento computacional y uso adecuado de los métodos estadísticos (Harris et al., 2020).

9.8.3. **Matplotlib**

La librería Matplotlib sirve para graficar los resultados del modelo de regresión logística. En particular, hace posible la representación gráfica de la curva ROC, una medida para evaluar la capacidad discriminante del modelo, en términos de la tasa de verdaderos positivos y la tasa de falsos positivos. Esta visualización apoya la evaluación cuantitativa del modelo (Hunter, 2007)..

9.8.4. **Seaborn**

Seaborn es una biblioteca de visualización estadística de alto nivel construida sobre Matplotlib y que produce gráficos más atractivos. Su aplicación analítica permite hacer más comprensible la presentación de resultados, identificar patrones y comprender mejor las métricas estadísticas. Aunque no está involucrada en la modificación del modelo, refuerza la comunicación visual de los resultados (Waskom, 2021)

9.8.5. **Scikit-learn**

Scikit-learn representa el componente central del proceso de modelado y evaluación. Esta librería contiene las funciones para separar los datos en conjuntos de entrenamiento y prueba, para el modelo de regresión logística y para evaluarlo. Además, es capaz de entrenar el modelo con máxima verosimilitud y hacer predicciones de clase y probabilidad, asegurando un análisis eficiente y reproducible. (Pedregosa et al., 2011).

9.8.6. **Standardscaler (Scikit-learn)**

El StandardScaler de Scikit-learn se utiliza para estandarizar las variables independientes, escalándolas para que tengan media cero y desviación estándar unitaria; esto es importante en la regresión logística para mejorar la estabilidad numérica del modelo y evitar que variables en diferentes escalas dominen la estimación de los coeficientes (Pedregosa et al., 2011).

9.8.7. **Métricas de evaluación (Scikit-learn)**

Scikit-learn proporciona diversas métricas para evaluar el desempeño del modelo de regresión logística, entre las que se incluyen la matriz de confusión, el reporte de clasificación

y la curva ROC con su respectivo valor AUC. Estas métricas permiten analizar la precisión, sensibilidad y capacidad discriminante del modelo, ofreciendo una evaluación integral de su eficacia como clasificador binario (Fawcett, 2006; Pedregosa et al., 2011).

9.9. Implementacion del modelo en python

9.9.1. Codificar variables categóricas

df_encoded = pd.get_dummies(

Al ser un modelo de regresión logistcia lo primero que hicimos fue asegurarnos de rabajar con vairbales numéricas, en este caso hicimos uso de la función dummies para poder codificar y registrasr observacoines categóricas de manera numérica para poder mantener la información que nos brindan dichas variables y respetar las necesidades del modelo.

Este comando implementa **One-Hot Encoding**, que transforma una variable categórica con $k$categorías en $k$variables binarias:

$$X_{ij} = \begin{cases} 1 & \text{si la observación pertenece a la categoría } j \\ 0 & \text{en caso contrario} \end{cases}$$

Tras la codificación, el modelo se expresa como:

$$z = \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_A D_A + \beta_B D_B + \beta_C D_C$$

donde $D_j$son variables indicadoras (0/1).

Cada coeficiente mide el **efecto marginal de pertenecer a esa categoría** sobre el log-odds de supervivencia.

drop_first=True,

Si se incluyen todas las categorías, ocurre que:

$$D_1 + D_2 + \cdots + D_k = 1$$

Esto genera **colinealidad perfecta**, haciendo imposible invertir la matriz de información durante la estimación por máxima verosimilitud.

Con drop_first=True, se elimina una categoría, que actúa como **grupo base**.

$$z = \beta_0 + \beta_1 D_1 + \beta_2 D_2 + \cdots + \beta_{k-1} D_{k-1}$$

La categoría omitida queda incorporada en el intercepto. Cada variable dummy contribuye a la verosimilitud como:

Si $D_{ij} = 1$:

→ La categoría modifica el log-odds.

Si $D_{ij} = 0$:

→ No tiene efecto.

$$p_i = \frac{1}{1 + e^{-(\beta_0 + \sum \beta_j D_{ij})}}$$

Si $D_{ij} = 1$, la categoría desplaza la probabilidad; si $D_{ij} = 0$, no tiene efecto.

9.9.2. Definición de variables dependiente e independientes

y = df_encoded["SOBREVIVE FINAL (POR ENCIMA DE LA MEDIANA DE 70,08%)"]

X = df_encoded.drop(columns=["SOBREVIVE FINAL (POR ENCIMA DE LA MEDIANA DE 70,08%)"])

Aquí se construye el espacio de variables explicativas Xy la variable respuesta binaria Y.

Matemáticamente, cada fila del dataset se representa como:

$$(X_i, Y_i) = ((x_{i1}, x_{i2}, \dots, x_{ip}), y_i)$$

donde: $y_i \in \{0,1\}$,

$X_i \in \mathbb{R}^p$.

Este paso define formalmente el problema de aprendizaje supervisado.

9.9.3. Separación en datos de entrenamiento y prueba

from sklearn.model_selection import train_test_split X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, random_state=42)

Se divide el conjunto total de observaciones Den dos subconjuntos disjuntos:

$$D_{\text{train}} \cup D_{\text{test}} = D \text{y} D_{\text{train}} \cap D_{\text{test}} = \emptyset$$

Concretamente:

- 70% de los datos → entrenamiento
- 30% de los datos → prueba

Este paso implementa validación fuera de muestra, necesaria para estimar el error de generalización:

$$\mathbb{E}_{(X,Y)\sim\mathcal{D}}[\text{Error del modelo}]$$

Sin este paso, el error estaría sesgado a la baja.

9.9.4. Entrenamiento del modelo: estimación de parámetros

model = LogisticRegression(max_iter=1000, random_state=42, class_weight='balanced')

Durante el entrenamiento, se ajusta el modelo logístico definido como:

$$P(Y = 1 \mid X) = \frac{1}{1 + e^{-z}}, z = \sum_{j=1}^{p} \beta_j X_j$$

donde: Xes el vector de variables explicativas,

- βes el vector de parámetros desconocidos.

9.9.5. Estimación por máxima verosimilitud

Los parámetros βse estiman maximizando la función de verosimilitud sobre el conjunto de entrenamiento:

$$L(\beta) = \prod_{i \in D_{\text{train}}} p_i^{y_i} (1 - p_i)^{1 - y_i}$$

Equivalentemente, se maximiza la log-verosimilitud:

$$\ell(\beta) = \sum_{i \in D_{\text{train}}} [y_i \log(p_i) + (1 - y_i) \log(1 - p_i)]$$

Este proceso ajusta el modelo exclusivamente a los datos de entrenamiento, sin acceso a la información del conjunto de prueba.

9.9.6. Optimización numérica durante el entrenamiento

Dado que no existe una solución analítica cerrada para maximizar $\ell(\beta)$, se emplean algoritmos iterativos de optimización, tales como:

- Gradiente descendente,
- Newton–Raphson,
- Métodos cuasi-Newton.

En términos matemáticos, el algoritmo busca valores $\hat{\beta}$tales que:

$$\nabla \ell(\hat{\beta}) = 0$$

Este proceso converge hacia un máximo local (y bajo condiciones regulares, global) de la función de verosimilitud.

9.9.7. Clasificación del modelo en el conjunto de prueba

Una vez estimados los parámetros, el modelo se aplica al conjunto de prueba para obtener probabilidades predichas:

$$\hat{p}_i = P(Y_i = 1 \mid X_i), i \in D_{\text{test}}$$

Este paso es estrictamente predictivo y no influye en la estimación de β.

El escalado es crítico en modelos que dependen de: Distancias euclidianas (KNN), Magnitudes relativas (SVM con kernel RBF), Gradientes sensibles a escala (redes neuronales profundas). La regresión logística no calcula distancias entre observaciones, sino:

$$z = \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p$$

Cada variable contribuye linealmente al logit, al no escalar, los coeficientes $\beta_j$se interpretan directamente en las unidades originales:

$$\beta_j \Rightarrow \text{cambio en el log-odds por unidad real de } X_j$$

Ejemplo conceptual:

- 1 gramo adicional,
- 1 unidad de densidad,
- 1 día adicional.

Si se escalan los datos:

$$X_j^* = \frac{X_j - \mu_j}{\sigma_j}$$

entonces:

$$\beta_j^* \Rightarrow \text{cambio por desviación estándar}$$

9.9.8. Sin escalado.

El escalado es crítico en modelos que dependen de: Distancias euclidianas (KNN), Magnitudes relativas (SVM con kernel RBF), Gradientes sensibles a escala (redes neuronales profundas). La regresión logística no calcula distancias entre observaciones, sino:

$$z = \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p$$

Cada variable contribuye linealmente al logit, al no escalar, los coeficientes $\beta_j$se interpretan directamente en las unidades originales:

$\beta_j \Rightarrow$ cambio en el log-odds por unidad real de X_j

Ejemplo conceptual:

- 1 gramo adicional,
- 1 unidad de densidad,
- 1 día adicional.

Si se escalan los datos:

$$X_j^* = \frac{X_j - \mu_j}{\sigma_j}$$

entonces:

$\beta_j^* \Rightarrow$ cambio por desviación estándar

y_pred = model.predict(X_test)

y_proba = model.predict_proba(X_test)[:, 1]

Esta instrucción asigna una clase binaria final a cada observación del conjunto de prueba.

El modelo calcula primero:

$$p_i = P(Y = 1 \mid X_i) = \frac{1}{1 + e^{-z_i}}$$

donde:

$$z_i = \beta_0 + \sum_{j=1}^{p} \beta_j X_{ij}$$

Luego aplica una regla de decisión: y_pred contiene solo etiquetas (0 = no supervivencia, 1 = supervivencia).

$$\hat{Y}_i = \begin{cases} 1 & \text{si } p_i \geq 0.5 \\ 0 & \text{si } p_i < 0.5 \end{cases}$$

9.9.9. Interpretación de coeficientes en la Regresión Logística

```
coef_df = pd.DataFrame({
    'Variable': X_train.columns,
    'Coeficiente (β)': model.coef_[0]
}).sort_values(by='Coeficiente (β)', key=abs, ascending=False)
```

La regresión logística estima un vector de parámetros:

$$\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_p)$$

donde cada coeficiente corresponde a una variable explicativa del modelo.

El modelo completo es:

$$\log\left(\frac{P(Y = 1 \mid X)}{1 - P(Y = 1 \mid X)}\right) = \beta_0 + \sum_{j=1}^{p} \beta_j X_j$$

Esto se conoce como la función logit.

Construcción del DataFrame de coeficientes

pd.DataFrame({ 'Variable': X_train.columns, 'Coeficiente (β)': model.coef_[0] })

Construimos el vector de parámetros estimados:

- Cada fila → una variable X_j

- Cada coeficiente → efecto marginal sobre el log-odds

Ordenamiento por magnitud absoluta

.sort_values(by='Coeficiente (β)', key=abs, ascending=False)

Ordenar por $|\beta_j|$ permite identificar:

$$\text{Variables con mayor impacto en } \log\left(\frac{P}{1-P}\right)$$

De esta manera medimos la influencia relativa dentro del modelo.

Ecuación logística estimada

print("logit(P) = β₀ + β₁·X₁ + β₂·X₂ + ...")

print(f"Intercepto (β₀): {model.intercept_[0]:.4f}")

Interpretación matemática del intercepto (β_0)

$$\beta_0 = \log\left(\frac{P(Y=1 \mid X=0)}{1-P(Y=1 \mid X=0)}\right)$$

Es el log-odds base de supervivencia cuando todas las variables explicativas toman valor cero.

No es una probabilidad directa, sino un punto de referencia matemático.

Interpretación estadística de cada coeficiente β_j

Para una variable X_j:

$$\beta_j = \frac{\partial}{\partial X_j} \log\left(\frac{P}{1-P}\right)$$

Un incremento unitario en $X_j$modifica el log-odds de supervivencia en $\beta_j$unidades, manteniendo constantes las demás variables.

Odds Ratio (OR)

$$OR_j = e^{\beta_j}$$

Interpretación:

- $OR > 1$→ aumenta la probabilidad de supervivencia
- $OR < 1$→ reduce la probabilidad de supervivencia
- $OR = 1$→ no hay efecto

9.9.10. Calculadora

A continuación, realizamos una calculadora que nos va a permitir Clasificar si la supervivencia será $\geq 70.08\%$ o $< 70.08\%$

En vez de predecir directamente el porcentaje, el modelo calcula:

$$P(Y = 1 \mid X)$$

Donde:

- $Y = 1$: supervivencia alta ($\geq$70.08%)

• $Y = 0$: supervivencia baja (<70.08%)

• X: conjunto de variables de producción

El modelo usa como variables independientes:

$$X = \{Genética, Alimento, Laboratorio, Densidad, HP/HA, Días\}$$

Es decir:

$$X = (x_1, x_2, x_3, x_4, x_5, x_6)$$

Transformación matemática de variables categóricas

Las variables categóricas no pueden usarse directamente en un modelo matemático.

Por eso se aplica One-Hot Encoding:

Ejemplo: Si existen 3 genéticas:

$$\text{GEN} = \{A, B, C\}$$

Se transforma en:

$$GEN_A = 1, GEN_B = 0, GEN_C = 0$$

Esto convierte las variables en vectores binarios, es decir; si una variable categórica tiene $k$categorías: $X_{cat} \rightarrow (x_1, x_2, \ldots, x_k)$

Paso 1: Selección de variables

Se extraen opciones del dataset: gen_opts = sorted(df["COD GEN"].unique())

Esto crea los valores posibles para el usuario.

Paso 2: Interfaz de entrada (widgets)

Se crean controles para que el usuario ingrese datos, ejemplo: dens_w = widgets.FloatSlider(...)

Esto permite seleccionar:

$$x_4 = \text{densidad}$$

Paso 3: Construcción del vector de entrada

Se crea un dataframe con los valores elegidos: $X_{input} = [x_1, x_2, x_3, x_4, x_5, x_6]$

Paso 4: Codificación categórica

Se convierte: $X_{input} \rightarrow X_{input\_codificado}$ Con: pd.get_dummies()

Paso 5: Alineación con el modelo entrenado

El modelo fue entrenado con un conjunto fijo de variables: $X_{train} = [x_1, x_2, \ldots, x_n]$

Entonces se hace: input_enc.reindex(columns=X_train.columns, fill_value=0) lo cual garantiza: $X_{input} \equiv X_{train}$

Paso 6: Predicción probabilística

El modelo calcula: prob = model.predict_proba(input_enc)[0,1]

Es decir: $prob = P(Y = 1 \mid X_{input})$

Paso 7: Clasificación final

Se aplica:

$$\text{Clase} = \begin{cases} Alta, & \text{si } prob \geq 0.5 \\ Baja, & \text{si } prob < 0.5 \end{cases}$$

Interpretación productiva del resultado

El modelo no predice el porcentaje exacto, sino la probabilidad de estar en el grupo de supervivencia alta. Ejemplo: Si: $prob = 0.82$, significa: Hay un 82% de probabilidad de que el ciclo tenga una supervivencia ≥70.08%.

9.9.11. Mejores combinaciones

Finalmente decidimos crear un modelo que nos permite hallar las mejores combinaciones posibles dentro de las distintas variables, realizando una simulación de escenarios.

El principal objetivo es Encontrar las combinaciones de variables productivas Xque maximizan la probabilidad de supervivencia alta.

$$\max_{X} \; P(Y = 1 \mid X)$$

Paso 1: Definir el umbral de clasificación

THRESHOLD = globals().get("best_threshold", 0.5)

Expresado:

$$\text{Clase} = \begin{cases} 1, & \text{si } P(Y = 1 \mid X) \geq \text{threshold} \\ 0, & \text{si } P(Y = 1 \mid X) < \text{threshold} \end{cases}$$

Donde:

- Si existe un umbral óptimo → se usa.
- Si no existe → se usa 0.5.

Paso 2: Combinaciones reales de variables categóricas

cat_combos = df[cat_cols].drop_duplicates()

Se generan combinaciones reales:

$$X_{cat} = \{Genética, Alimento, Laboratorio\}$$

Paso 3: Generación de rangos numéricos "inteligentes"

Se generan valores dentro de rangos estadísticos.

Densidad, *np.linspace(q25, q75, 3)*

$$DENSIDAD \in [Q_{25}, Q_{75}]$$

Se generan 3 valores equidistantes:

$$d_1, d_2, d_3$$

Esto representa:

- Baja densidad típica
- Densidad media
- Alta densidad típica

Días

dias_vals = [median], Se usa el valor central, la mediana.

Peso final

peso_vals = np.linspace(q50, q90, 3)

$$PESO \in [Q_{50}, Q_{90}]$$

Esto representa:

- Peso medio
- Peso alto
- Peso muy alto

4. Construcción matemática de los escenarios

El código hace un producto cartesiano: por cada categoría, por cada densidad, por cada hp*ha, por días, por peso.

Es decir: Escenarios=C_cat×D×H×T×P

Donde:

C_cat: combinaciones categóricas

D: valores de densidad

H: valores de HP/HA

T: días

P: peso final

Si hay:

10 combinaciones categóricas

3 densidades

3 HP/HA

1 valor de días

3 pesos

Entonces: 10×3×3×1×3=270 escenarios

5. Transformación de los escenarios

Igual que en la calculadora:

$$X_{escenario} \rightarrow X_{codificado}$$

Con *pd.get_dummies()*

Y luego: $X_{escenario} \equiv X_{train}$ Para que el modelo pueda evaluar.

6. Predicción probabilística

proba = model.predict_proba(scenarios_enc)[:,1]

El modelo predice:

$$P_i = P(Y = 1 \mid X_i)$$

Para cada escenario:

$$i = 1,2,3,\ldots,n$$

7. Clasificación de cada escenario

CLASE = (proba >= THRESHOLD)

$$CLASE_i = \begin{cases} 1, & P_i \geq threshold \\ 0, & P_i < threshold \end{cases}$$

8. Optimización por ranking

El algoritmo ordena: *sort_values("PROBABILIDAD", ascending=False)*

Ordenando P_i de mayor a menor y posterior seleccionar los top 20

9. Interpretación productiva

De ésta manera respondemos qué combinaciones de genética, alimento, densidad y manejo tienen mayor probabilidad de supervivencia alta. No predice un ciclo, explora cientos de ciclos posibles y selecciona los mejores.

9.10. Evaluación del Modelo

9.10.1. Accuracy

model.score(X_test, y_test)

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Mide el porcentaje total de aciertos

9.10.2. Precision

$$\text{Precision} = \frac{TP}{TP + FP}$$

Responde a:

"De todos los casos que el modelo predijo como supervivencia, ¿cuántos fueron correctos?"

9.10.3. Recall

$$\text{Recall} = \frac{TP}{TP + FN}$$

Responde a:

"De todos los casos reales de supervivencia, ¿cuántos logró detectar el modelo?"

F1-score

$$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

9.10.4. Matriz de confusión

Donde:

- **TN (True Negative)**: correctamente clasificado como no superviviente
- **TP (True Positive)**: correctamente clasificado como superviviente
- **FP (False Positive)**: clasificado como superviviente cuando no lo es
- **FN (False Negative)**: clasificado como no superviviente cuando sí lo es

En nuestro estudio:

- 0 = supervivencia < 70.08%
- 1 = supervivencia ≥ 70.08%

confusion_matrix(y_test, y_pred)

Esta función calcula:

$$CM_{ij} = \sum_{k=1}^{n} I(y_k = i \wedge \hat{y}_k = j)$$

donde:

- $I(\cdot)$es la función indicadora
- $i, j \in \{0,1\}$

Cuenta frecuencias conjuntas, no probabilidades.

Visualización con heatmap

sns.heatmap(..., annot=True, fmt='d', cmap='Blues')

- annot=True → muestra los valores absolutos (frecuencias)
- fmt='d' → datos discretos (conteos)
- cmap='Blues' → intensidad del color proporcional a la frecuencia

Etiquetas de clases

xticklabels=["<70.08%", "≥70.08%"]

yticklabels=["<70.08%", "≥70.08%"]

La binarización de la supervivencia se basó en la mediana (70.08%), una medida frente a alta dispersión.

Esto permite:

- Interpretar errores en términos productivos
- Evaluar riesgos reales de mala clasificación

I**nterpretación matemática de cada celda**

Si la matriz es:

$$\begin{bmatrix} TN & FP \\ FN & TP \end{bmatrix}$$

Entonces:

- **Exactitud global**:

$$\frac{TP + TN}{N}$$

- **Error tipo I (FP)**:

$$P(\hat{Y} = 1 \mid Y = 0)$$

- **Error tipo II (FN)**:

$$P(\hat{Y} = 0 \mid Y = 1)$$

En acuicultura, el error tipo II suele ser más crítico, porque implica no detectar condiciones favorables reales de supervivencia.

9.10.5. Curva ROC

Código que se está explicando

fpr, tpr, _ = roc_curve(y_test, y_proba)

Esta función calcula la Curva ROC (Receiver Operating Characteristic) evaluando el modelo para todos los posibles umbrales de decisión t.

Para cada umbral $t \in [0,1]$:

$$\hat{Y}_t = \begin{cases} 1 & \text{si } P(Y = 1 \mid X) \geq t \\ 0 & \text{si } P(Y = 1 \mid X) < t \end{cases}$$

Visualización de la curva ROC

plt.plot(fpr, tpr, color='darkorange', lw=2, label=f'ROC (AUC = {roc_auc_score(y_test, y_proba):.3f})')

Interpretación matemática

La curva ROC es una función paramétrica:

$$ROC(t) = (FPR(t), TPR(t))$$

No depende del umbral fijo (como 0.5), sino del comportamiento global del clasificador.

Línea de referencia aleatoria

plt.plot([0,1], [0,1], linestyle='--')

Esta línea representa un clasificador sin capacidad discriminante:

$$TPR = FPR \Rightarrow AUC = 0.5$$

Cualquier modelo útil debe ubicarse por encima de esta diagonal.

AUC: Área bajo la curva

roc_auc_score(y_test, y_proba)

$$AUC = \int_0^1 TPR(FPR)\, d(FPR)$$

En términos probabilísticos:

$$AUC = P(P(Y = 1 \mid X^+) > P(Y = 1 \mid X^-))$$

Es probabilidad que el modelo asigne una mayor probabilidad de supervivencia a un caso verdaderamente superviviente que a uno no superviviente.

10. RESULTADOS

Resultados Descriptivos

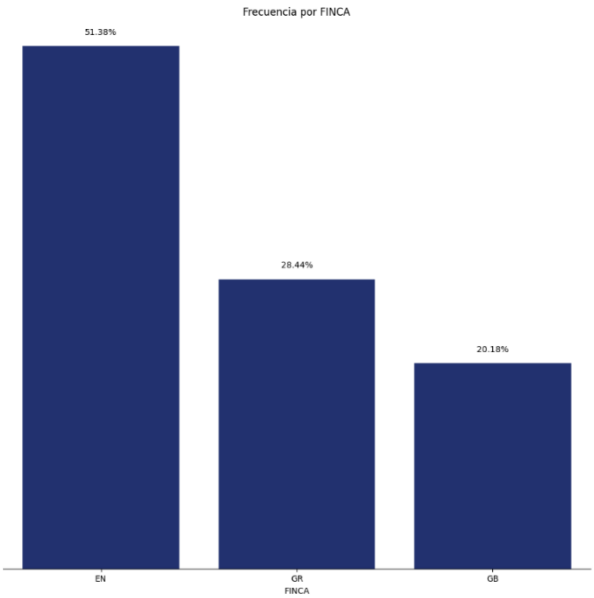


Figura 6 *Distribución de datos*

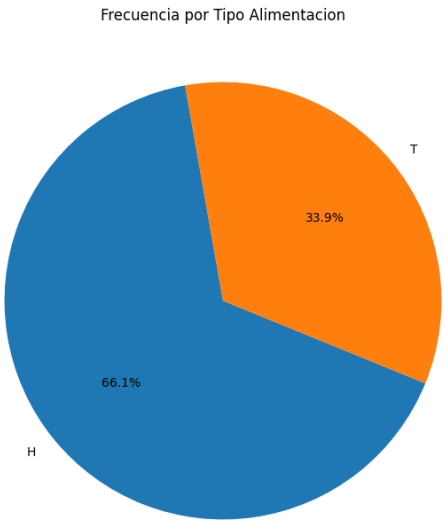


Figura 7 *Distribución de datos "Tipo de alimentación"*

En el análisis descriptivo que más apreciado cómo concentramos una mayor cantidad de datos provenientes de la finca EN con un 51% representando más de la mitad de los datos de

la base de datos total, De igual manera 2/3 partes de los datos totales son con un tipo de alimentación mediante hidrófonos representando el 66%.

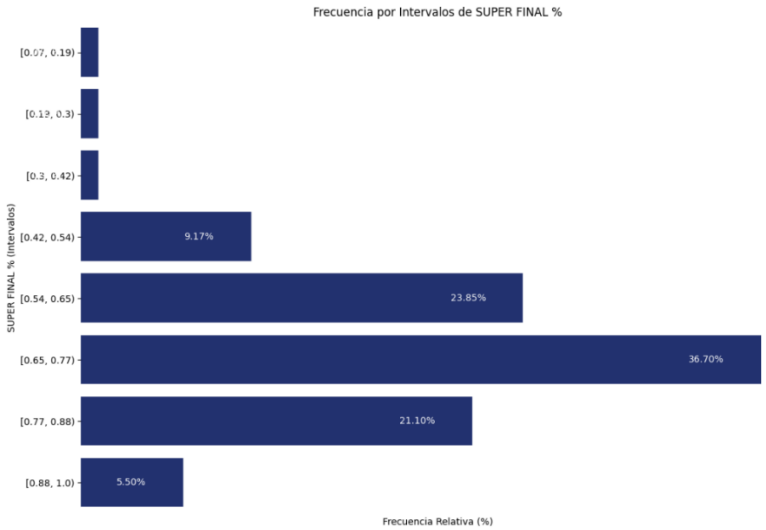


Figura 8 *Frecuencia de intervalos de supervivencia del camarón*

Podemos apreciar como en el rango de 65 a 77% de supervivencia final se encuentra concentrada la mayor cantidad de los datos alrededor de 36.7%, seguido por el rango de 54 a 65% de supervivencia final

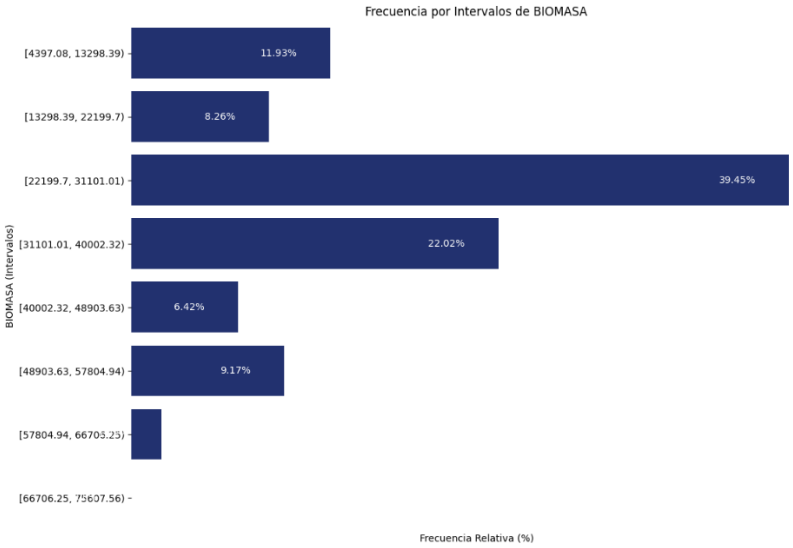


Figura 9 *Frecuencia de intervalos de Biomasa*

Así mismo un 39.45% de los datos se encuentran en un rango de biomasa de 22199.7 a 31101, seguido por un 22% que se encuentran en un rango de 31101 a 40002.

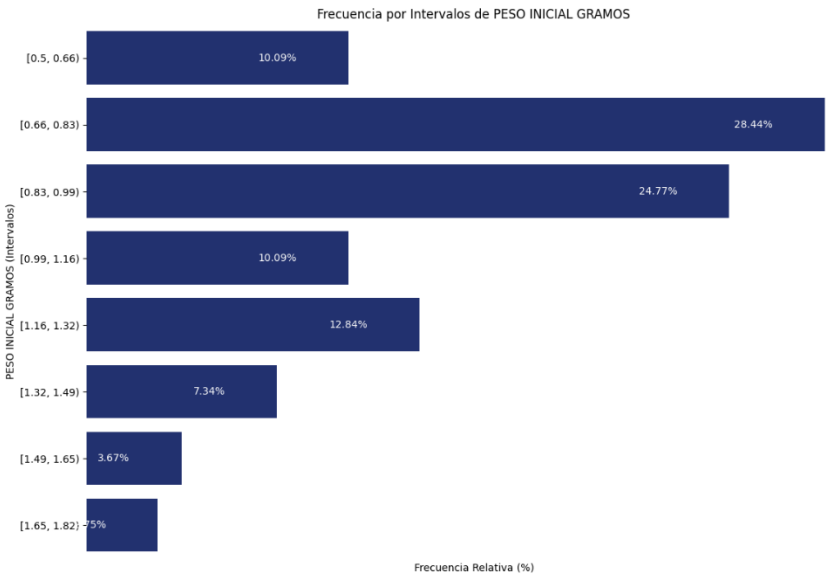


Figura 10 *Frecuencia de intervalos de Peso Inicial en gramos*

La mayoría de las cosechas presentan un peso promedio inicial por camarón de alrededor de 0.66 g a 0.83 g, 0.83 g a cero 99 g entre estos dos rangos se concentra más del 50% de las cosechas

Lo cual es bastante interesante cuando lo comparamos con los pesos finales en gramos en promedio por camarón cómo es muy poco probable que se vayan a los extremos en este caso que pasen de los 32 g o que sean por debajo de los 20 g y como la gran cantidad de datos de las cosechas se encuentran concentrados en alrededor de 24 a 30 g representando alrededor del 60% de las cosechas

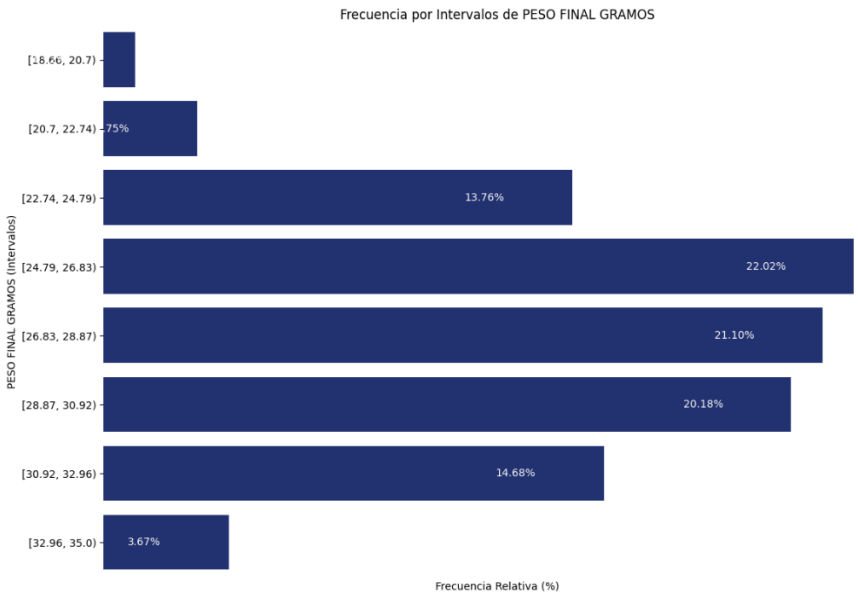


Figura 11 *Frecuencia de Peso Final en gramos*

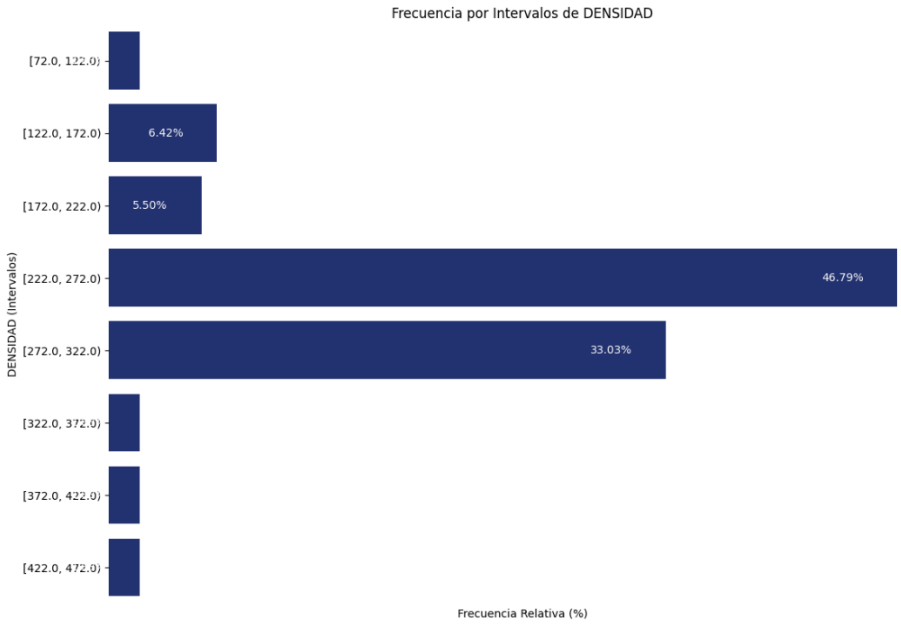


Figura 12 *Intervalos de densidad de siembra*

En cuestiones de densidad es evidente como salvo situaciones atípicas la densidad de los animales por metros cuadrados alrededor de los 222 a los 270 representando casi el 50% de todas las cosechas seguido de 270 a 320 representando un restante 30% de cosechas entre estos dos rangos representan el 80% de las cosechas como nos señalan evidentemente una tendencia en la densidad de dichos procesos.

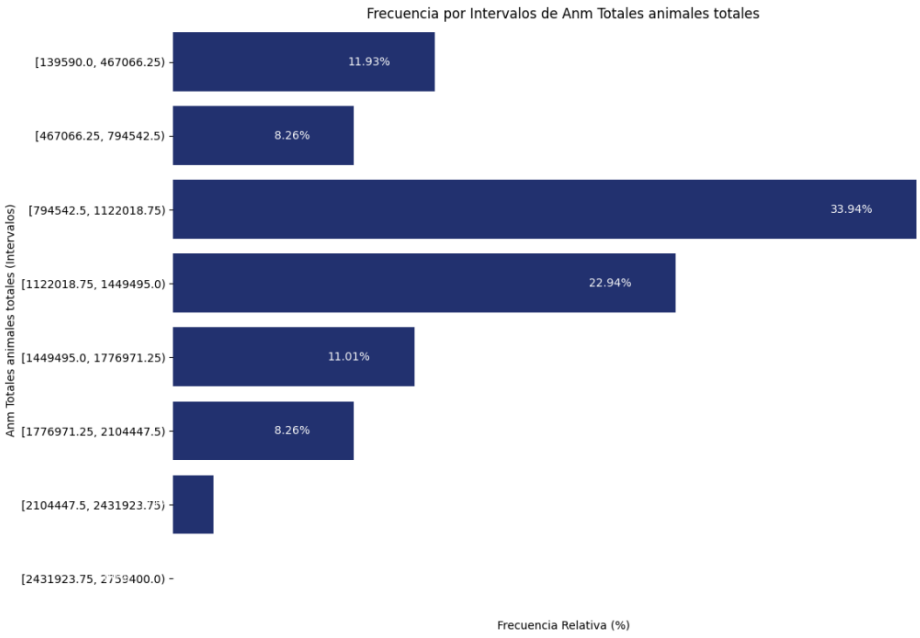


Figura 13 *Frecuencia de intervalos Animales Sembrados*

Es bastante interesante analizar como en distintas variables se puede apreciar como la mayoría de los datos se encuentran concentrados en un rango medio siendo muy pocos los casos donde suelen irse a los extremos del esto lo podemos apreciar en la frecuencia de animales totales como la gran mayoría de los datos se encuentran ubicados Entre los 7000000 a los 14000000 de animales cosechados

de igual manera en los días de procrea que los 27.52% de procesos tuvieron alrededor de 30 a 32 días de pre cría seguido por un 27 a 30 días con 18% y un 24 a 27 con 17%

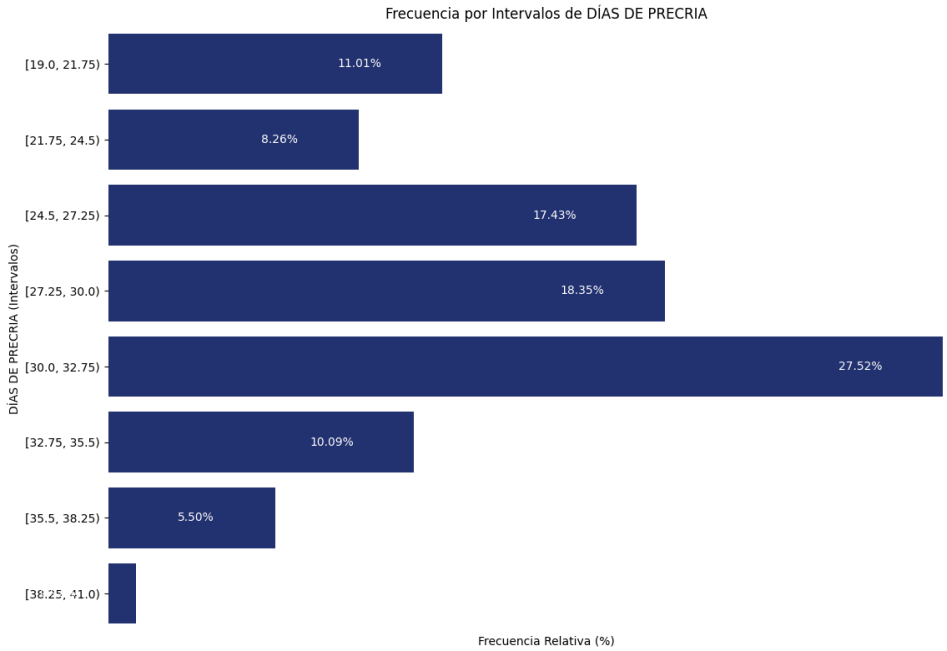


Figura 14 *Intervalos de días en pre-cría*

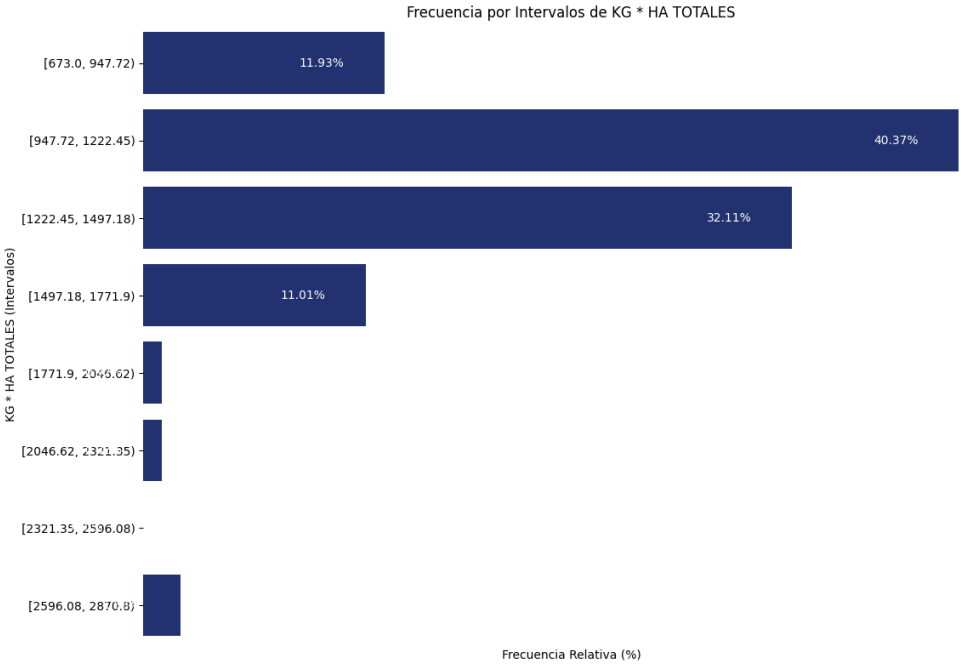


Figura 15 *Intervalos de siembra Kg/Ha*

En cuanto a la alimentación se puede apreciar cómo la norma suele ser un promedio de 947 a 1200 kilogramos de alimentos por hectárea por cosecha, representando un 40%, siendo

perseguido por un rango de 1222 a 1500, lo cual nos señala un promedio de cuánto alimento se suele consumir por proceso.

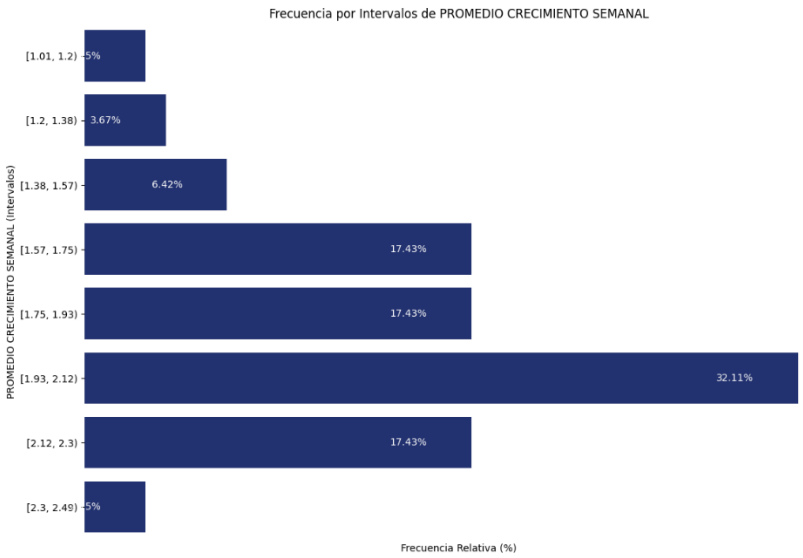


Figura 16 *Frecuencia promedio de crecimiento semanal en gramos*

Directamente relacionada podemos ver como se da en el peso inicial en promedio y cómo se concentra la mayoría de los datos en el peso final en promedio tiene sentido cuando vemos como el promedio de crecimiento semanal tiene un evidente rango donde se concentra la mayoría de datos siendo en este caso el alrededor de los 2 g de crecimiento semanal, Representando como que crezcan menos de 1.2 g o más de 2.3 g es tan poco frecuente como se puede apreciar con tan solo 5% en ambas categorías.

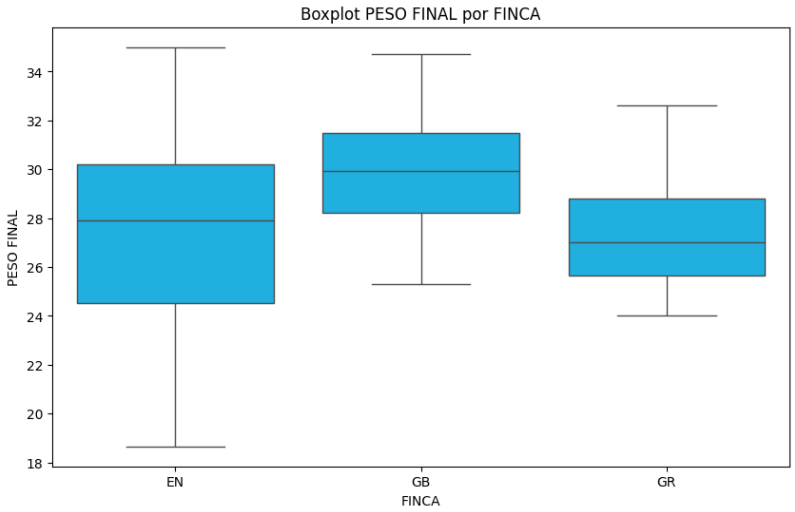


Figura 17 *BoxPlot peso Final del camarón*

Se puede apreciar como los procesos que provienen de la finca GB tienen una tendencia a presentar un mayor peso final en promedio por camarón se concentran por encima de 30 g, lo cual nos puede indicar como a su vez como lo vimos anteriormente la finca n es el que nos entra la mayor cantidad de datos sin embargo la finca que nos entrega a los camarones más grandes es la GB y la que nos entrega unos camarones un poco menor en tamaño es la GR, Interesantísimo cómo ver cómo el tope de peso de la finca con mayor concentración de datos es el 30 y como una finca con una menor concentración de cosechas como la GB sobrepasa con casi un 50% de sus procesos ese límite de 30 g que representa en la EN.

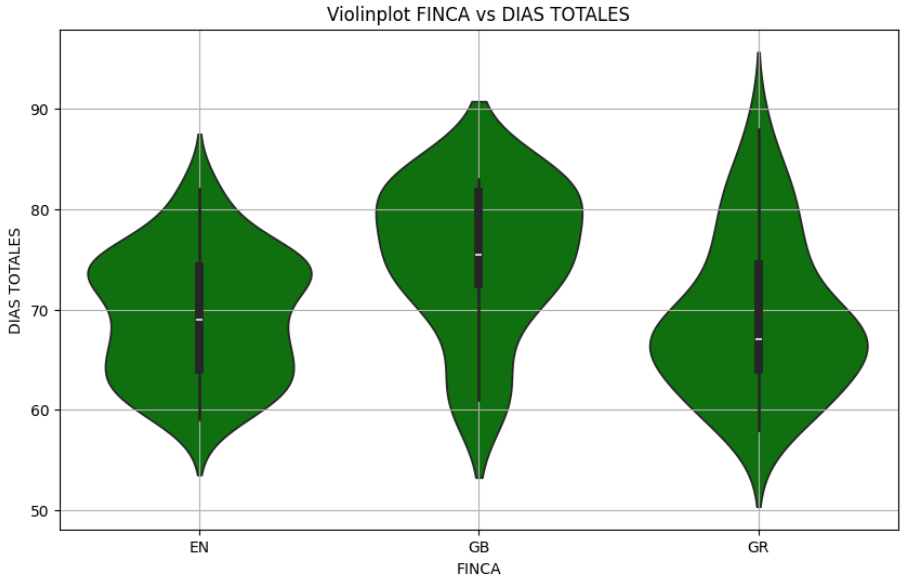


Figura 18 *ViolinPlot promedio de días de cosecha promedio*

Al contrastar con la información anterior no nos sorprende como de igual manera la finca que nos representa unos camarones de menor tamaño es la que a su vez concentra la mayor cantidad de sus procesos en un menor número de días totales y la finca que nos representa un mayor tamaño es la que se concentra en una mayor cantidad de sus procesos con un número de días totales por encima de alrededor de los 75 días y como la finca más grande se encuentran de una manera más variada la concentración de sus procesos en relación a sus días totales siendo la finca GB la que suele sobrepasar los 75 días de manera común y como casi la finca GR no tiende a pasar de los 70 días.

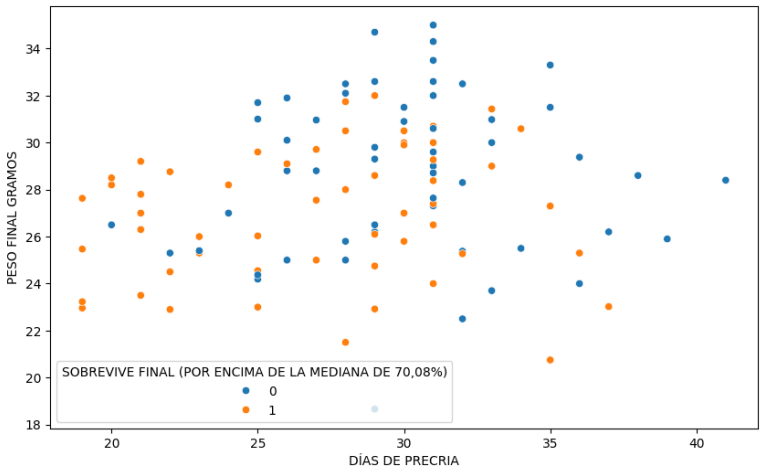


Figura 19 *Dispersión del peso en relación a los días en pre-cría*

Analizar este gráfico es interesantísimo puesto que nos demuestra cómo en distintas ocasiones procesos que cuentan con menor números de días de pre cría tienen una tendencia a que el promedio de peso final de sus camarones sea menor, más sin embargo, también presentan una tendencia a que estos procesos sobrepasen nuestro umbral de supervivencia preestablecido del 70% y cómo procesos que presentan una mayor cantidad de días de pre cría tienden a presentar un mayor peso final de sus animales pero también se concentran una mayor negatividad a la hora de sobrepasar el 70% de supervivencia contrastando completamente, Una percepción erróneamente establecida que los camarones con mayores números de pero hay cría si bien sí tienden a pesar más no es una característica confiable para creer que van a presentar una mayor tasa de supervivencia.

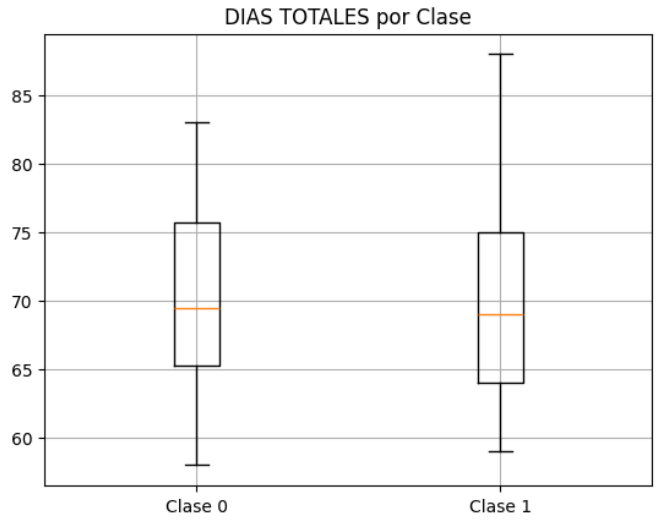


Figura 20 *Distribución de la variable 'Días Totales' según la variable dependiente (Supervivencia)*

Aquí se puede apreciar como en procesos que qué cuentan con un una menor cantidad de días totales tienen una ligera ventaja a la hora de sobrepasar los túmbele supervivencia del 70%, la media de los datos se encuentra en el error de los 70 días de igual manera sin embargo se puede apreciar cómo en la clase 1 los cual sí sobrepasan nuestro umbral tienden un poco a contar con una mayor cantidad de procesos por debajo de los 65 días mientras que en la clase dos no hay ningún proceso que baje de ese límite y sí que sobrepasen de los 75 días.

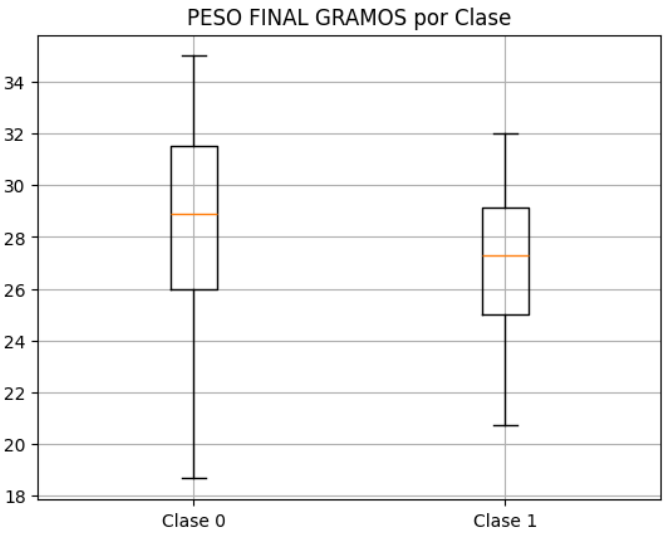


Figura 21 *Distribución de la variable 'Peso Final" según la variable dependiente (Supervivencia)*

Se evidencia nuevamente como el peso final de un camarón que en promedio es mayor también aumenta la probabilidad que dichos procesos de cosecha no superen nuestro umbral de 70% supervivencia, y es claro cómo se demuestra que si una cosecha tiene un promedio de peso final de 30 g no va a superar el 70% de supervivencia total.

Resultados modelo

Principales coeficientes (valores originales):

Variable	Coeficiente (β)
LABORATORIO_SANLAB FSA S.A.S.	1.011485
COD GEN_Q-HLNE1.235(100%)	-0.888153
COD GEN_MB7-H215OMG2.2G(100%)	0.677883
COD GEN_LNCEMOG4(100%)	-0.645727
LABORATORIO_SANLAB BSO S.A.	-0.628382
COD GEN_HP10BV17G2(100%)	0.628283
LABORATORIO_SHRIMP WORLD ALEXMAR S.A.S.	-0.619984
COD GEN_Genomar(M-04-02)(100%)	-0.619984
COD GEN_MB3-OMPI8C36G2.2G(100%)	0.575090
COD GEN_NC41AG(100%)	0.572341
COD GEN_PB-LNCEMOG4(100%)	-0.550631
LABORATORIO_Acuatecsa	-0.550195
COD GEN_C-OMPI9H227G5(100%)	-0.497057
COD GEN_O-OMTP41G2(100%)	-0.490037
COD GEN_PRMCont29RG(100%)	-0.489931

Figura 22 *Variables con mayor impacto en la probabilidad de supervivencia.*

La interpretación matemática correcta es usando: Odds Ratio $= e^{\beta}$. Esto indica cuánto cambian las probabilidades relativas.

LABORATORIO_SANLAB FSA S.A.S. $\beta = 1.011$

Odds ratio: $e^{1.011} \approx 2.75$. Usar este laboratorio multiplica por 2.75 veces las probabilidades de supervivencia alta respecto al laboratorio base.

COD GEN_Q-HLNE1.235(100%) $\beta = -0.888$

Odds ratio: $e^{-0.888} \approx 0.41$. Esta genética reduce las probabilidades de supervivencia alta a 41% del valor base, es decir es una genética asociada a peor supervivencia.

COD GEN_MB7-H2150MG2.2G(100%) $\beta = 0.678$

Odds ratio: $e^{0.678} \approx 1.97$ Esta genética casi **duplica la probabilidad** de supervivencia alta.

LABORATORIO_BSO S.A. $\beta = -0.628$

Odds ratio: $e^{-0.628} \approx 0.53$ Reduce las probabilidades de supervivencia alta a un 53% del valor base.

Factores que aumentan supervivencia:

- LABORATORIO_SANLAB FSA
- GEN_MB7-H2150MG2.2G
- GEN_HPJ10BV17G2
- GEN_OMPIBC36G2

Factores que reducen supervivencia:

- GEN_Q-HLNE1.235
- LABORATORIO_BSO
- LABORATORIO_SHRIMP WORLD

Como la genética y el laboratorio son factores que representan un mayor peso en el porcentaje de clasificación en la supervivencia final de las cosechas, sin embargo, el tipo de alimentación no representa una mayor en relación con dichos resultados.

⏳ TOP 20 MEJORES COMBINACIONES ESTIMADAS

	COD GEN	TIPO DE ALIMENTO	LABORATORIO	DENSIDAD	HP/HA	DIAS TOTALES	PESO FINAL GRAMOS	PROBABILIDAD	CLASE
0	G-OMPI7G2BS(100%)	H	SANLAB FSA S.A.S.	250.0	5.43	69	28.2	0.832531	1
1	G-OMPI7G2BS(100%)	H	SANLAB FSA S.A.S.	265.5	5.43	69	28.2	0.831475	1
2	G-OMPI7G2BS(100%)	H	SANLAB FSA S.A.S.	281.0	5.43	69	28.2	0.830414	1
3	G-OMPI7G2BS(100%)	H	SANLAB FSA S.A.S.	250.0	6.95	69	28.2	0.824481	1
4	G-OMPI7G2BS(100%)	H	SANLAB FSA S.A.S.	265.5	6.95	69	28.2	0.823385	1
5	T-PRMConT26Bio7.36(100%)	H	SANLAB FSA S.A.S.	250.0	5.43	69	28.2	0.823295	1
6	G-OMPI7G2BS(100%)	H	SANLAB FSA S.A.S.	281.0	6.95	69	28.2	0.822284	1
7	T-PRMConT26Bio7.36(100%)	H	SANLAB FSA S.A.S.	265.5	5.43	69	28.2	0.822194	1
8	T-PRMConT26Bio7.36(100%)	H	SANLAB FSA S.A.S.	281.0	5.43	69	28.2	0.821087	1
9	P-OMTP41G2(100%)	H	SANLAB FSA S.A.S.	250.0	5.43	69	28.2	0.820201	1
10	P-OMTP41G2(100%)	H	SANLAB FSA S.A.S.	265.5	5.43	69	28.2	0.819084	1
11	P-OMTP41G2(100%)	H	SANLAB FSA S.A.S.	281.0	5.43	69	28.2	0.817962	1
12	HLNE1.235(100%)	H	SANLAB FSA S.A.S.	250.0	5.43	69	28.2	0.816658	1
13	G-OMPI7G2BS(100%)	H	SANLAB FSA S.A.S.	250.0	8.47	69	28.2	0.816129	1
14	HLNE1.235(100%)	H	SANLAB FSA S.A.S.	265.5	5.43	69	28.2	0.815524	1
15	G-OMPI7G2BS(100%)	H	SANLAB FSA S.A.S.	265.5	8.47	69	28.2	0.814993	1
16	T-PRMConT26Bio7.36(100%)	H	SANLAB FSA S.A.S.	250.0	6.95	69	28.2	0.814900	1
17	HLNE1.235(100%)	H	SANLAB FSA S.A.S.	281.0	5.43	69	28.2	0.814385	1
18	G-OMPI7G2BS(100%)	H	SANLAB FSA S.A.S.	281.0	8.47	69	28.2	0.813851	1
19	T-PRMConT26Bio7.36(100%)	H	SANLAB FSA S.A.S.	265.5	6.95	69	28.2	0.813758	1

Figura 23 *Variables con mayor impacto en la probabilidad de supervivencia*

Genéticas más frecuentes

Las genéticas que más se repiten en el Top 20 son:

1. **G-OMPI7G2BS(100%)**
2. **T-PRMConT26Bio7.36(100%)**
3. **P-OMTP41G2(100%)**
4. **HLNE1.235(100%)**

Estas genéticas están asociadas con mayor supervivencia según el modelo.

Laboratorio dominante

En las 20 mejores combinaciones domina el LABORATORIO SANLAB FSA S.A.S. Es decir, según los datos históricos, este laboratorio presenta mejores resultados de supervivencia, esto coincide con los coeficientes positivos que viste antes.

Tipo de alimento

En todos los escenarios TIPO DE ALIMENTO HIDROFONO. Es decir podemos apreciar como a pesar de no representar un mayor peso a la hora de la clasificación, se evidencia como este alimento está asociado con mejores probabilidades de supervivencia en los datos.

Densidad

El modelo indica que las densidades **moderadas** son más favorables, no aparecen densidades extremas como apreciamos en los descriptivos.

$$250 \leq \text{DENSIDAD} \leq 281 \text{ PL/m}^2$$

HP/HA óptimo

En el caso de los aireadores podemos ver cómo el número de oradores óptimo para aumentar la supervivencia es alrededor de 5 a 8 aireador es decir que producciones intermedias representan una mayor estabilidad para la supervivencia

$$5.43 \leq HP/HA \leq 8.47$$

Días de cultivo

Nuevamente se puede apreciar como evidentemente procesos qué cuentan con una menor cantidad de 70 a 75 días, siendo más en concreto de 69 días son los procesos que presentan una mayor probabilidad a la hora de pasar el umbral de supervivencia de 70%.

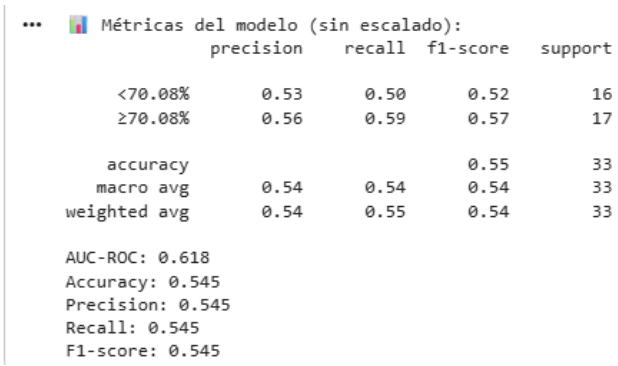
$$DIAS = 69$$

Peso final óptimo

Otra clara señal al igual que los días totales, aquí evidenciamos cómo el peso óptimo está por debajo de los 30 g siendo en concreto 28.2 g la característica que presenta una mayor probabilidad de supervivencia.

$$PESO = 28.2 \text{ g}$$

Resultados Evaluación



📊 Métricas del modelo (sin escalado):

	precision	recall	f1-score	support
<70.08%	0.53	0.50	0.52	16
≥70.08%	0.56	0.59	0.57	17
accuracy			0.55	33
macro avg	0.54	0.54	0.54	33
weighted avg	0.54	0.55	0.54	33

AUC-ROC: 0.618
Accuracy: 0.545
Precision: 0.545
Recall: 0.545
F1-score: 0.545

Figura 24 *Métricas de confiabilidad del modelo.*

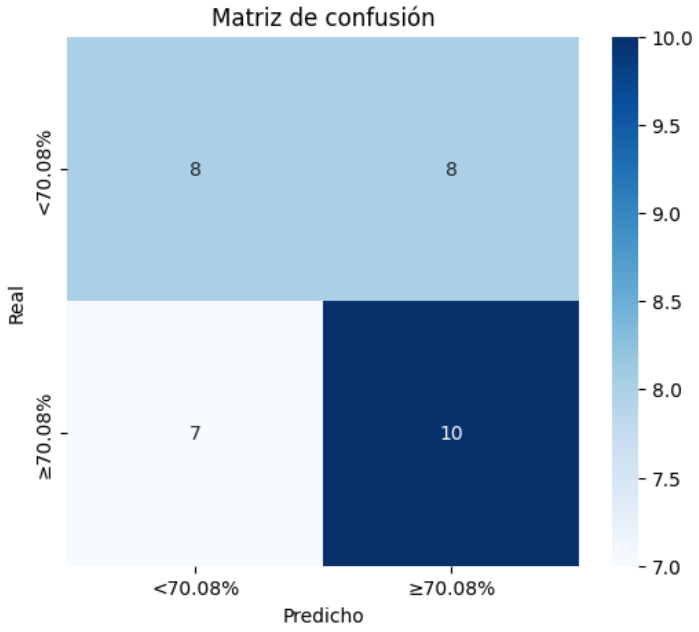


Figura 25 *Matriz de confusión*

Matriz de confusión

- Verdaderos negativos (VN): 8 - predice bajo y es bajo
- Falsos positivos (FP): 8 -predice alto pero es bajo
- Falsos negativos (FN): 7 - predice bajo pero es alto
- Verdaderos positivos (VP): 10 - predice alto y es alto

Cosechas con supervivencia baja (<70.08%) Total: $8 + 8 = 16$. El modelo acertó 8 y falló 8, dejándonos con una precisión en ciclos malos $= 50\%$

Cosechas con supervivencia alta (≥70.08%) Total: $7 + 10 = 17$. El modelo acertó 10 y falló 7, dejándonos con una recall en ciclos buenos $= 60\%$

Accuracy.- El modelo acierta el 54.5% de los casos.

Precision.- Cuando el modelo predice supervivencia alta, acierta el 54.5% de las veces.

Recall.- El modelo detecta el 54.5% de los ciclos buenos.

F1-score

$$F1 = 0.545$$

El F1 es el balance entre precisión y recall, por lo que el desempeño general del modelo es moderado.

Interpretación por clase

Clase de supervivencia alta

- Precision: 56%
- Recall: 59%

El modelo funciona mejor detectando ciclos buenos que malos.

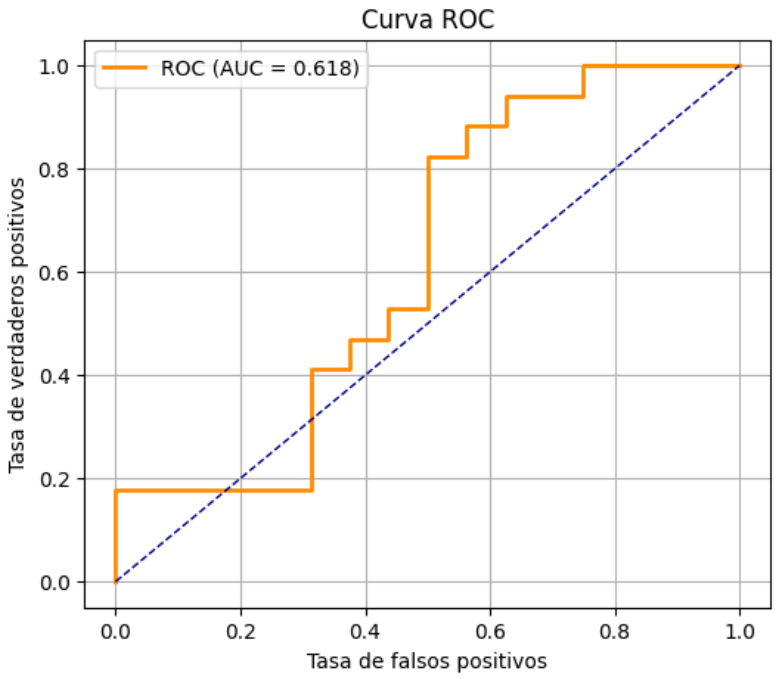


Figura 26 *Curva ROC.*

Curva ROC

Valor AUC: $AUC = 0.618$

El AUC mide la capacidad del modelo para separar ciclos buenos de malos, un AUC de 0.61 en la teoría podría presentarse como moderada-baja en la práctica representa un resultado aceptable.

El AUC nos representa una acierto en el 61.8% de los casos, a la hora de identificar correctamente cuál es mejor en caso de tomar cosechas buenas y malas al azar

Análisis

El modelo detecta patrones reales en los datos, su capacidad predictiva aún no es la mejor, sin embargo estas medidas de evaluación pueden subir con la correcta retroalimentación y remuestreo, puesto a que identifica más de la mitad de los ciclos buenos.

El modelo logístico se desempeña de manera aceptable, con una exactitud de 54.5% y AUC de 0.618. Estos resultados indican que el modelo acierta más que el azar, pero no lo suficiente como para fiarse de él para decidir. La matriz de confusión indica que el modelo clasifica correctamente el 59% de las altas supervivencias, pero con errores altos en ambas clases. Como simulador, el modelo puede explotarse para estudiar casos, pero no para la toma de decisiones automatizada sin supervisión técnica.

11. DISCUSIÓN

El proyecto se centra en la empleabilidad de la regresión logística, para identificar mediante la genética del camarón, tipo de alimentación y laboratorio, la supervivencia del camarón en el proceso de producción de tal manera que permite identificar que gen es el Optimo para introducir a producción este análisis se puede contrastar contra el proyecto de "Productive lifespan and stayability in Argentine Creole cattle in the Salado Basin"(Maria Tpopayan & Luciana Erneta, 2025) el cual identifica que raza de bovinos serían los más indicados para la producción de terneros para carne de la región de Cuenca de Salado usando Regresión Logística.

En dicho articulo se recopilaron datos, productivos y genealógicos provenientes del establecimiento Cruz de Guerra, situado en la Cuenca Del Salado (Buenos Aires, Argentina) Se centraron en la Raza Bovina "Criolla Argentica" (bca). Se trataron datos históricos de 1989 hasta el 2023. Para ello se consideraron a las hembras que iniciaron y finalización su vida productiva. Se evaluaron a 196 vacas con información disponibles desde el nacimiento, primer parto, destete y ultimo parto del ciclo productivo.

Al iniciar, realizaron una limpieza de datos, eliminando registros inconclusos e incongruentes, principalmente de las hembras que aún no terminaban su ciclo de prodcutivo. Las principales variables fueron la longitud de vida productiva, numero de crías totales, edad del primer parto y su capacidad reproductiva, esta variable se clasifico como binaria para poder establecer un mínimo de 7 crías por madre.

Dada la complejidad de las variables relacionadas y la naturaleza dispersa de la longevidad productiva de los bovinos, establecieron evaluar diferentes enfoques estadísticos con la finalidad de establecer un modelo adecuado para explicar el comportamiento reproductivo de las hembras y su permanencia en el rodeo.

En primera instancia, realizaron un análisis exploratorio integral, evaluando la relación entre el LVP Y NCT, porque estas variables presentaban una correlación extremadamente alta. Se definió que la NCT podría funcionar para establecer el indicador de la vida productiva del bovino debido a su fenómeno productivo. Así mismo se emplearon modelos lineales mixtos para evaluar los dos tipos de grupos EPP24 VS EPP36, con esto pueden controlar las varianzas y comparar medias.

Adicional a ello debido a que la permanencia del bovino en el ciclo productivo es binario, debido a que de esta manera pueden definir si son de una alta producción o baja, así mismo mide el éxito reproductivo estimando el odds ratio correspondiente. El modelo es el siguiente:

$$\text{Log}\left(\frac{p}{1-p}\right) = \beta_0 - \beta_1 EPP$$

Esta fórmula permite realizar una estimación de la probabilidad de éxito reproductivo en función del primer parto (epp), ya que transforma dicha probabilidad mediante la función logit, la cual asegura que las predicciones permanezcan dentro del intervalo de 1 y 0. Así mismo

mide la manera en que el epp influye en la eficiencia productiva, de tal manera que se puede optimizar recursos para obtener crías de calidad.

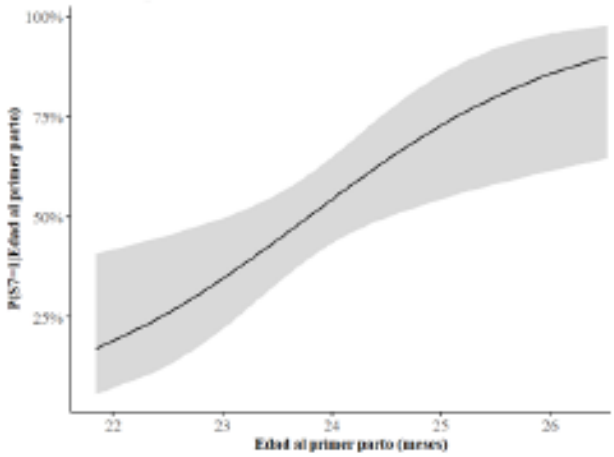


Figura 27 *Modelo Regresión Discusión*

El siguiente grafico nos permite evaluar que el modelo de regresión logística en este caso que mientras más aumenta la probabilidad de tener 7 crías, aumenta a su vez la edad del primer parto dando como resultado que mediante el análisis se puede establecer un plan de optimización de planificación genética.

Por otro lado, el objetivo de este estudio es identificar correctamente la relación de las variables de tipo de alimentación, código genético, laboratorio en relación con la supervivencia del camarón. Con enfoque en seleccionar que gen sería el mejor a producir dentro de la producción de la empresa XYZ y conocer que laboratorios brindan la mejor estructura genética. En contraste, el objetivo de la investigación contraria es Analizar la longitud de vida productiva de las Vacas y la permanecía de los vientres de la raza Bovina Criolla Argentina. Estos proyectos se centran en realizar a viabilidad de producción de genes dependiendo de diferentes aspectos, sin embargo, se tienen premisas similares las cuales conllevan a similitudes

estructurales como lo son la longevidad de la producción, así también ambos rescatan el valor de emplear el modelo de regresión Logística.

El estudio de longitud de vida productiva se centró en estudiar a las hembras con el fin de identificar su incidencia tanto en la producción de nuevas crías para carne, como su vida útil siendo vacas lecheras. En consecuencia, lograron identificar que el promedio de vida útil del Gen Criollo es en promedio de 93.85 meses, lo que les permitió evaluar la posibilidad de que el uso recurrente de esta línea genética podría llegar a incrementar la producción de carne en la región de estudio. Debido a que en este mercado mientras más longevas sean son más rentables debido a su rentabilidad por crianza. (Maria Tpopayan & Luciana Erneta, 2025)

En contraste la presente investigación, la cual busca identificar no solo si la producción del código genético sea más productiva, busca identificar a su vez la calidad que brinda cada uno de los laboratorios relacionados a la empresa. Teniendo en cuenta otros aspectos como el tipo de alimentador y días de producción. Con la finalidad de poder crear una base estructurada la cual servirá para poder crear políticas de desarrollo tanto con los laboratorios como en la producción.

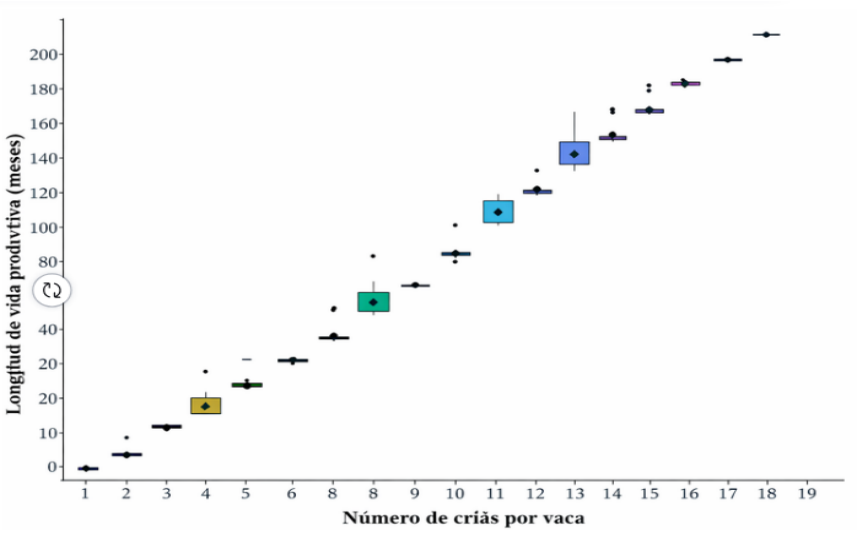


Figura 28 *Longitud de vida productiva*

12. CONCLUSIÓN

En este estudio se evidencia la importancia del análisis de datos y el uso de herramientas de Machine Learning en la acuicultura, específicamente en la supervivencia del camarón en función de la genética y la dieta. El empleo de estas metodologías proporciona información más exacta y verdadera; al conocer la información de datos productivos se puede determinar qué factores realmente influyen en la supervivencia y productividad del cultivo. De esta manera se pueden hacer mejores elecciones y eso es fundamental para hacer más eficientes y rentables a las empresas camaroneras.

En el transcurso de la práctica, se hizo una revisión bibliográfica del comportamiento productivo del camarón y algunas técnicas de Machine Learning que se podrían utilizar para este tipo de análisis. Esto permitió desarrollar una teoría que describa de manera sencilla cómo estas herramientas pueden ser aplicadas en el sector acuícola y por qué son tan viables para estudios como estos.

Luego, se entrenaron modelos de Machine Learning con datos históricos de producción de una empresa camaronera. Con este método se pudo aislar la información genética y la información de dieta y así poder encontrar diferencias en la supervivencia entre los grupos. Esta clasificación permitió conocer cómo afecta cada variable en los resultados productivos, pues no todas las combinaciones generan los mismos resultados.

Finalmente, los resultados se compararon con otros estudios de metodología similar, coincidiendo en que la genética y la nutrición son determinantes en la producción acuícola. Esto validó los hallazgos y demostró que las herramientas de análisis de datos se pueden usar en esta industria.

En conclusión, el análisis de supervivencia segmentado por genética y nutrición es una herramienta que puede contribuir a mejorar la producción de camarón. Una buena interpretación de los datos ayuda a las empresas a tomar mejores decisiones, a escoger mejores combinaciones productivas y a disminuir ciertos riesgos. Es de este modo como el uso del Machine Learning y el análisis estadístico contribuyen a alcanzar los objetivos de esta investigación y mejorar la competitividad del sector, pero aún puede mejorar en este tipo de estudios.

13. REFERENCIAS

Aqua Culture Asia Pacific. (2023). Marine Shrip in 2022: plauged by diseases. Aqua Culture Asia Pacific.

Balcázar, R. d. (2019). ANÁLISIS DE LOS FACTORES ESTRATÉGICOS QUE INCIDEN EN LA PRODUCCIÓN CAMARONERA ECUATORIANA. 29.

Boyd, C. E. (1990). Water Quality in Ponds for Aquaculture. Alabama Agricultural Experiment Station, Auburn University.

Boyd, C. E. (1990). Water Quality in Ponds for Aquaculture. Alabama Agricultural Experiment Station.

Carlos Joel Cabrera Peñaloza, Rosana de Jesús Eras Agila, & Margot Isabel Lalangui Balcazar. (2022). APLICACIÓN NIC 41 “ACTIVOS BIOLÓGICOS” EN LAS EMPRESAS CA-MARONERAS, PROVINCIA EL ORO –ECUADOR. Obtenido de https://aes.ucf.edu.cu/index.php/aes/article/view/511

Cayuela, L. (2015). Modelos lineales generalizados (GLM). Madrid: Universidad Rey Juan Carlos.

Chatterjee, S., & Hadi, A. (2015). Regression Analysis by Example. Wiley.

David G. Kleinbaum , & Mitchel Klein. (2012). Springer Nature Link. Obtenido de Springer Nature Link: https://link.springer.com/book/10.1007/978-1-4419-6646-9

David W. Hosmer, & Stanley Lemeshow. (2000). Applied Logistic Regression (2nd ed.). https://onlinelibrary.wiley.com/doi/book/10.1002/0471722146.

Douglas C. Montgomery, & Elizabeth A. Peck. (2000). Introduction to Linear Regression Analysis (5th ed.). Wiley.

Engle, C. R. (2010). Aquaculture Economics and Financing: Management and Analysis.

Fahrmeir, L., Kneib, T., Lang, S., & Marx, B. (2013). Regression: Models, Methods and Applications. Springer.

Fast, Andrew W., & Lester, Lawrence J. (1992). Marine Shrimp Culture: Principles and Practices. Elsevier Science Publishers.

Food and Agriculture Organization of the United Nations. (07 de 06 de 2024). Food and Agriculture Organization of the United Nations. Obtenido de FAO: https://www.fao.org/newsroom/detail/fao-report-global-fisheries-and-aquaculture-production-reaches-a-new-record-high/en

Gelman, A., & Hill, J. (2007). Data Analysis Using Regression and Multilevel/Hierarchical Models. Cambridge University Press.

Guerra, S. W. (20 de 10 de 2025). primicias.ec. Obtenido de primicias.ec: https://www.primicias.ec/revistagestion/analisis/comercio-exterior-ecuador-camaron-petroleo-exportaciones-107478/

Guillén, M., & Alonso, A. (2020). Modelos estadísticos para la toma de decisiones. Editorial Universitat de Barcelona.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). The elements of statistical learning. Springer.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). The Elements of Statistical Learning (2nd ed.). Springer.

Hemmert, G., Laura M. Schons, Jan Wieseke, & Heiko Schimmelpfennig. (2016). Log-likelihood-based pseudo R^2 in logistic regressioN. Obtenido de https://doi.org/10.1177/0049124115615340

Hosmer, D. W, & Lemeshow, S. (2000). Applied Logistic Regression. Wiley.

Hosmer, D. W., & S. L. (2000). Applied Logistic Regression. Wiley.

Hunter, J. (2007). Matplotlib: A 2D graphics environment. Computing in Science & Engineering, 90–95.

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2021). An Introduction to Statistical Learning. Springer.

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2021). An introduction to statistical learning: With applications in R. Springer.

Kuhn, M., & Johnson, K. (2013). Applied Predictive Modeling. Springer.

Kuhn, M., & Johnson, K. (2019). Feature Engineering and Selection: A Practical Approach for Predictive Models. CRC Press.

Lee, F. (S-F.). IBM. Obtenido de IBM: https://www.ibm.com/think/topics/logistic-regression

Lightner, D. (2011). Virus diseases of farmed shrimp in the Western Hemisphere (the Americas). Journal of Invertebrate Pathology, 110-130.

McCullagh, P., & Nelder, J. (1989). Generalized Linear Models. Chapman & Hall.

McKinney, W. (2010). Data structures for statistical computing in Python. Proceedings of the 9th Python in Science Conference (págs. 51–56). Austin, Texas: SciPy.

Nagelkerke, N. J. (1991). A note on a general definition of the coefficient of determination. Nagelkerke: Biometrika.

Piedrahita, Y. (25 de 06 de 2018). Global Seafood. Obtenido de Global Seafood: https://www.globalseafood.org/advocate/shrimp-farming-industry-in-ecuador-part-1/

Real Academia Española. (2025). Real Academia Española. Obtenido de Real Academia Española: https://dle.rae.es/estad%C3%ADstica

Ross, S. M. (2014). A First Course in Probability. Pearson. Obtenido de https://media.pearsoncmg.com/aw/wps-wraps/retired/

Servicios de consultoría y asesoramiento para el crecimiento | Fortune Business Insights. Obtenido de Insights, Fortune Business: https://www.fortunebusinessinsights.com/es/shrimp-market-106303

Tabares, S. (10 de 05 de 2023). Sustainable Shirmp Parnership. Obtenido de Sustainable Shirmp Parnership: https://sustainableshrimppartnership.org/es/el-nacimiento-casual-de-la-acuicultura-de-camaron-ecuatoriana/

Tacon, A. G. (1988). THE NUTRITION AND FEEDING OF FARMED FISH AND SHRIMP - A TRAINING MANUAL. BRASILIA.

Tacon, A. G. (2002). Thematic Review of Feeds and. Aquatic Farms Ltd.

Tacon, A., & Metian, M. (2015). Feed matters: Satisfying the feed demand of aquaculture. Reviews in Fisheries Science & Aquaculture, 1–10.

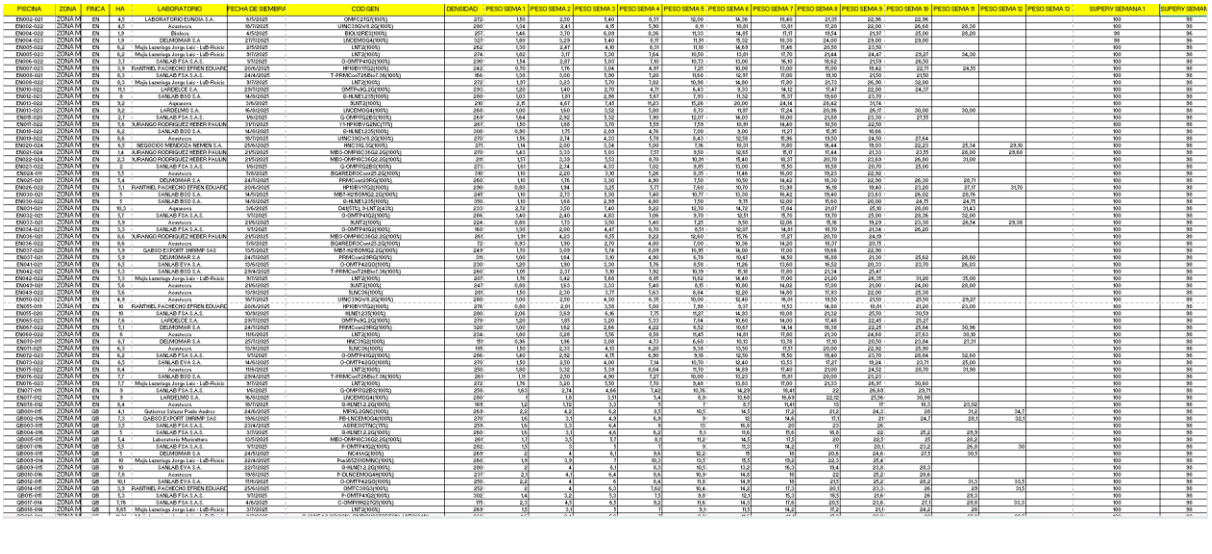
Te Grotenhuis, M., & Pelzer, B. (2015). Dummy variables and their interpretation in regression analysis. Sociological Methods & Research, 44 o 45.

Vistazo. (18 de 10 de 2022). Innovación, tecnología y dietas funcionales se destacan en Aqua Expo. Innovación, tecnología y dietas funcionales se destacan en Aqua Expo.

Wasserman, L. (2004). All of Statistics: A Concise Course in Statistical Inference. Springer.

14. ANEXOS

Anexo 1 Base de datos



Anexo 2 Script Limpieza en Python

```
#Importo las librerías necesarias para análisis decriptvio
import pandas as pd #Esta librería sirve para manipulación de datos
import numpy as np #Esta librería sirve para trabajar con valores nulos o perdidos
```

```
##CARGAR LOS DATOS A COLAB##
from google.colab import files
uploaded = files.upload()
```

```
Elegir archivos Sin archivos seleccionados Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.
Saving DATOS CAMARON.xlsx to DATOS CAMARON.xlsx
```

```
#LEER EL ARCHIVO DE DATOS
# Cargar datos
df = pd.read_excel("DATOS CAMARON.xlsx", header=1)
df.head()
```

	PISCINA	ZONA	FINCA	HA	LABORATORIO	FECHA DE SIEMBRA	COD GEN	DENSIDAD	PESO SEMA 1	PESO SEMA 2	...	semana 8	semana 9	semana 10	semana 11	semana 12	semana 13	semana 14	KG * HA TOTALES	BIOMASA	PROMEDIO CRECIMIENTO SEMANAL
0	EN071-025	ZONA M	EN	6.3	Acuatecsa	2025-09-13	5LNC36(100%)	195	1.50	2.33	...	2.95	1.61	1.61	NaN	NaN	NaN	NaN	1178.000000	32438.19600	1.617606
1	EN049-022	ZONA M	EN	5.6	Acuatecsa	2025-09-13	5LNC36(76%); 5LNC36(24%)	281	1.50	2.30	...	3.39	4.8	4.68	1.7	NaN	NaN	NaN	1596.666667	32749.13376	1.596705
2	EN033-021	ZONA M	EN	5.9	Acuatecsa	2025-06-21	9LNT2(100%)	224	0.80	1.73	...	2.37	2.43	3.03	3.2	NaN	NaN	NaN	1501.000000	36505.82520	1.966886
3	EN049-	ZONA	EN	5.6	Acuatecsa	2025-06-	9LNT2(100%)	247	0.80	1.63		5.70	5	5	NaN	NaN	NaN	NaN	1644.000000	26291.86560	2.125478

```
#OBSERVAMOS LA INOFRMACIÓN DEL DATASET
df.info()
```

Mostrar el resultado oculto

```
#BUSCAR LOS VALORES FALTANTES
df.isnull().sum()
```

Mostrar el resultado oculto

```
#EJECUTO DESCRIPTIVOS
df.describe()
```

	HA	FECHA DE SIEMBRA	DENSIDAD	PESO SEMA 1	PESO SEMA 2	I
count	109.000000	109	109.000000	109.000000	109.000000	1
mean	5.896881	2025-06-24 00:52:50.642201856	263.275229	1.636972	3.082477	
min	0.740000	2025-04-22 00:00:00	72.000000	0.700000	1.400000	
25%	4.500000	2025-05-24 00:00:00	250.000000	1.300000	2.400000	
50%	5.900000	2025-06-21 00:00:00	269.000000	1.600000	3.100000	
75%	7.600000	2025-07-22	294.000000	1.910000	3.600000	

```python
#Deteccion de Outlayers o valores averrantes
#Creamos la funcion
#Primero defino que variables voy a usar
numericas = ["SUPER FINAL %"]
#le digo que los outliers, lo valide entre llaves {}, todo lo que esta en llaves significa que está fuera de lo normal.
outliers = {}
#ahora los cuartiles:
for col in numericas:
  Q1 = df[col].quantile(0.25)
  Q3 = df[col].quantile(0.75)
  IQR = Q3 - Q1
  lower_bound = Q1 - 1.5 * IQR
  upper_bound = Q3 + 1.5 * IQR
  outliers[col] = df[(df[col] < lower_bound) | (df[col] > upper_bound)]
  print('Cantidad de outliers detectados por variable')
  print(outliers)
```

Mostrar el resultado oculto

```python
#Visualizacion de outliers con boxplot
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(15, 10))
for i, col in enumerate(numericas, 1):
    plt.subplot(2, 3, i+1)
    sns.boxplot(x=df[col], color = 'blue')
    plt.xlabel('')
    plt.title(f'Boxplot de {col}')
plt.tight_layout()
plt.show()
```

Mostrar el resultado oculto

```python
#Deteccion de Outlayers o valores averrantes
#Creamos la funcion
#Primero defino que variables voy a usar
numericas = ["SUPERVICENCIA ESTIMADA"]
#le digo que los outliers, lo valide entre llaves {}, todo lo que esta en llaves significa que está fuera de lo normal.
outliers = {}
#ahora los cuartiles:
for col in numericas:
  Q1 = df[col].quantile(0.25)
  Q3 = df[col].quantile(0.75)
  IQR = Q3 - Q1
  lower_bound = Q1 - 1.5 * IQR
  upper_bound = Q3 + 1.5 * IQR
  outliers[col] = df[(df[col] < lower_bound) | (df[col] > upper_bound)]
  print('Cantidad de outliers detectados por variable')
  print(outliers)
```

```python
#Visualizacion de outliers con boxplot
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(15, 10))
for i, col in enumerate(numericas, 1):
    plt.subplot(2, 3, i+1)
    sns.boxplot(x=df[col], color = 'blue')
    plt.xlabel('')
    plt.title(f'Boxplot de {col}')
plt.tight_layout()
plt.show()
```

Mostrar el resultado oculto

```python
#Variable super
import matplotlib.pyplot as plt
import seaborn as sns

#Paso 1. Asegurarse que no hay valores nulos,
#por si gubo fallos en el data clean
Supervivencia = df['SUPER FINAL %'].dropna()

#Paso 2. Calcular el numero de clases con la regla de Sturges, porque no tenemos N definido
n= len(Supervivencia)
k= int(np.ceil(1 + 3.322 * np.log10(n)))

#Paso 3. Crear intervalos con 2 decimales
min_val = Supervivencia.min()
max_val = Supervivencia.max()
bins = np.linspace(min_val, max_val, k+1)
bins = np.round(bins, 2)

#Paso 4. Agrupampos las clases
clases = pd.cut(Supervivencia, bins=bins, right=False)
freq_absR = clases.value_counts().sort_index()
freq_relR = (freq_absR / len(Supervivencia) * 100). round(2)
freq_acumR = freq_absR.cumsum()
freq_rel_acumR = freq_relR.cumsum().round(2)

#Paso 5. Creamos la Tabla
tablaR = pd.DataFrame({'Supervivencia': freq_absR.index,
                       'Frecuencia Absoluta': freq_absR.values,
                       'Frecuencia Relativa (%)': freq_relR.values,
                       'Frecuencia Acumulada': freq_rel_acumR.values})

#Mostramos tabla
display(tablaR)
```

```python
#REMPLAZAMOS CON LA MEDIANA LAS VARIABLES E.Y. Y P.S.
df['SUPER FINAL %'] = df['SUPER FINAL %'].fillna(round(df['SUPER FINAL %'].median(),1))
```

```python
#LIMPIEZA VARIABLE CATEGORICA
#Analizamos la variable Cod Genetico
df['COD GEN'].unique()
```

Mostrar el resultado oculto

Anexo 3 Script Descriptivo en Python

```python
#Importo las librerías necesarias para análisis decriptvio
import pandas as pd #Esta librería sirve para manipulación de datos
import numpy as np #Esta librería sirve para trabajar con valores nulos o perdidos
```

```python
#CARGAR LOS DATOS A COLAB#
from google.colab import files
uploaded = files.upload()
```

```
Elegir archivos Sin archivos seleccionados Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.
Saving DATOS CAMARON.xlsx to DATOS CAMARON.xlsx
```

```python
#LEER EL ARCHIVO DE DATOS
df = pd.read_excel("DATOS CAMARON.xlsx", header=1)
df.head()
```

Mostrar el resultado oculto

DESCRIPTIVO VARIABLE CATEGORICA

```python
#Variable department
#Paso 1. Calcular la frecuencia absoluta
freq_abs = df['FINCA'].value_counts()
```

```python
#Paso 2. Calcular la frecuencia relativa
freq_rel = df['FINCA'].value_counts(normalize=True) * 100
```

```python
#Paso 3. Calcular la frecuencia acumulada
freq_acum = freq_abs.cumsum()
```

```python
#Paso 4. Calcular la frecuencia relativa acumulada
freq_rel_acum = freq_rel.cumsum()
```

```python
# PASO 5. CREAMOS LA TABLA UNIFICADA / ROUND() ES PARA QUE PONGA DECIMALES PARA
# HACERLO MAS REALISTA
tablaFINCA = pd.DataFrame({'Categoría': freq_abs.index,
                           'Frecuencia Absoluta': freq_abs.values,
                           'Frecuencia Relativa (%)': freq_rel.round(2).values,
                           'Frecuencia Acumulada': freq_acum.values,
                           'Frecuencia Relativa Acumulada (%)': freq_rel_acum.round(2).values})
```

```
#Mostramos tabla
display(tablaFINCA)
```

Mostrar el resultado oculto

```
#CREAMOS EL GRAFICO
import matplotlib.pyplot as plt
import seaborn as sns

#Tomamos los datos de la tabla de datos
plt.figure(figsize=(10,10)) #Generamos y delimitamos el area para el grafico
ax = sns.barplot(x=tablaFINCA['Categoría'],
                 y=tablaFINCA['Frecuencia Relativa (%)'], color= '#15297c')

#Etiquetas encima de cada barra
for i, v in enumerate(tablaFINCA['Frecuencia Relativa (%)']):
    ax.text(i, v+1, f'{v:.2f}%',
            ha='center', va='bottom', fontsize=10)

#Personalizacion estetica
plt.title('Frecuencia por FINCA')
plt.xlabel('FINCA')
plt.ylabel('')
ax.set_yticks([])     #Eliminar el eje y
sns.despine(left=True) #Eliminar los bordes
plt.tight_layout()     #Ajustar el espaciado

plt.show() #Mostramos el grafico
```

Mostrar el resultado oculto

```
#Variable Alimentacion
#Paso 1. Calcular la frecuencia absoluta
freq_absALI = df['TIPO DE ALIMENTO'].value_counts()
#Paso 2. Calcular la frecuencia relativa
freq_relALI = df['TIPO DE ALIMENTO'].value_counts(normalize=True)*100
#Paso 3. Calcular la frecuencia acumulada
freq_acumALI = freq_absALI.cumsum()
#Paso 4. Calcular la frecuencia relativa acumulada
freq_rel_acumALI = freq_relALI.cumsum()
# PASO 5. CREAMOS LA TABLA UNIFICADA / ROUND() ES PARA QUE PONGA DECIMALES PARA
# HACERLO MAS REALISTA
tablaALI = pd.DataFrame({'Categoría': freq_absALI.index,
                         'Frecuencia Absoluta': freq_absALI.values,
                         'Frecuencia Relativa (%)': freq_relALI.round(2).values,
                         'Frecuencia Acumulada': freq_acumALI.values,
                         'Frecuencia Relativa Acumulada (%)': freq_rel_acumALI.round(2).values})
#Mostramos tabla
display(tablaALI)
```

```
#CREAMOS GRAFICO DE PASTEL

plt.figure(figsize=(8,8))
plt.pie(freq_relALI, labels=freq_absALI.index.astype(str),
        autopct='%1.1f%%', startangle=100)
plt.title('Frecuencia por Tipo Alimentacion')
plt.show()
```

Mostrar el resultado oculto

```
#EN CASO DE DUDA SOBRE EL TIPO DE DATOS USAR:
df.info()
```

Mostrar el resultado oculto

```
df['COD GEN'].unique()
```

Mostrar el resultado oculto

LA DE FRECUENCIAS DE DATOS NO AGRUPADOS

```
#Variable DIAS TOTALES
import matplotlib.pyplot as plt
import seaborn as sns

#Paso 1. Asegurarse que no hay valores nulos,
#por si gubo fallos en el data clean
DIAS = df['DIAS TOTALES'].dropna()

#Variable Salary #es diferente la sintaxis porque estamos contando valores,
#ya no estamos contando caracteristicas
#Paso 2. Calcular la frecuencia absoluta, manteniendo el origen (index)
freq_absDT = DIAS.value_counts().sort_index()
#Paso 3. Calcular la frecuencia relativa
freq_relDT = (freq_absDT / len(DIAS) * 100). round(2)
#Paso 4. Calcular la frecuencia acumulada
freq_acumDT = freq_absDT.cumsum()
#Paso 5. Calcular la frecuencia relativa acumulada
freq_rel_acumDT = freq_relDT.cumsum().round(2)

#Paso 6. CREAMOS LA TABLA UNIFICADA
#ROUND() ES PARA QUE PONGA DECIMALES
#PARA HACERLO MAS REALISTA
tablaDT = pd.DataFrame({'DIAS': freq_absDT.index,
                        'Frecuencia Absoluta': freq_absDT.values,
                        'Frecuencia Relativa (%)': freq_relDT.values,
                        'Frecuencia Acumulada': freq_acumDT.values,
                        'Frecuencia Relativa Acumulada (%)': freq_rel_acumDT.values})
#Mostramos tabla
display(tablaDT)
```

```
#CREAMOS EL GRAFICO DE BARRAS (OTRA OPCION)
import matplotlib.pyplot as plt
import seaborn as sns

#Tomamos los datos de la tabla de datos
plt.figure(figsize=(14,14)) #Generamos y delimitamos el area para el grafico
ax = sns.barplot(x=tablaDT['DIAS'],
                 y=tablaDT['Frecuencia Relativa (%)'], color= '#15297c')

#Etiquetas encima de cada barra
for i, v in enumerate(tablaDT['Frecuencia Relativa (%)']):
    ax.text(i, v+1, f'{v:.2f}%',
            ha='center', va='bottom', fontsize=10)
#Personalizacion estetica
plt.title('Frecuencia por DIAS')
plt.xlabel('DIAS')
plt.ylabel('')
ax.set_yticks([])     #Eliminar el eje y
sns.despine(left=True) #Eliminar los bordes
plt.tight_layout()     #Ajustar el espaciado

plt.show() #Mostramos el grafico
```

Mostrar el resultado oculto

ABLES POR AGRUPAMIENTO (DATOS AGRUPADOS)

```
#Variable Supervivencia final
import matplotlib.pyplot as plt
import seaborn as sns

#Paso 1. Asegurarse que no hay valores nulos,
#por si gubo fallos en el data clean
SupervivenciaF = df['SUPER FINAL %'].dropna()

#Paso 2. Calcular el numero de clases con la regla de Sturges, porque no tenemos N definido
n= len(SupervivenciaF)
k= int(np.ceil(1 + 3.322 * np.log10(n)))

#Paso 3. Crear intervalos con 2 decimales
min_val = SupervivenciaF.min()
max_val = SupervivenciaF.max()
bins = np.linspace(min_val, max_val, k+1)
bins = np.round(bins, 2)

#Paso 4. Agrupampos las clases
clases = pd.cut(SupervivenciaF, bins=bins, right=False)
freq_absSF = clases.value_counts().sort_index()
freq_relSF = (freq_absSF / len(SupervivenciaF) * 100). round(2)
freq_acumSF = freq_absSF.cumsum()
```

```python
#Paso 4. Agrupampos las clases
clases = pd.cut(SupervivenciaF, bins=bins, right=False)
freq_absSF = clases.value_counts().sort_index()
freq_relSF = (freq_absSF / len(SupervivenciaF) * 100). round(2)
freq_acumSF = freq_absSF.cumsum()
freq_rel_acumSF = freq_relSF.cumsum().round(2)

#Paso 5. Creamos la Tabla
tablaSF = pd.DataFrame({'SupervivenciaF': freq_absSF.index,
                        'Frecuencia Absoluta': freq_absSF.values,
                        'Frecuencia Relativa (%)': freq_relSF.values,
                        'Frecuencia Acumulada': freq_rel_acumSF.values})

#Mostramos tabla
display(tablaSF)
```

Mostrar el resultado oculto

```python
import matplotlib.pyplot as plt
import seaborn as sns

#CREAMOS EL GRAFICO DE BARRAS para tablaSF
plt.figure(figsize=(12, 8)) # Ajustar el tamaño del gráfico para una mejor visualización
ax = sns.barplot(y=tablaSF['SupervivenciaF'].astype(str), # Cambiado a eje Y para barras horizontales
                 x=tablaSF['Frecuencia Relativa (%)'], color='#15297c') # Cambiado a eje X

#Etiquetas dentro de cada barra
for i, v in enumerate(tablaSF['Frecuencia Relativa (%)']):
    ax.text(v - 2, i, f'{v:.2f}%', # Posición x ajustada para estar dentro de la barra
            color='white', va='center', ha='right', fontsize=10) # Color blanco y alineación a la derecha

#Personalizacion estetica
plt.title('Frecuencia por Intervalos de SUPER FINAL %')
plt.xlabel('Frecuencia Relativa (%)') # Etiqueta del eje X
plt.ylabel('SUPER FINAL % (Intervalos)') # Etiqueta del eje Y
ax.set_xticks([])     # Eliminar el eje X para que solo se vean los porcentajes dentro de las barras
sns.despine(bottom=True, left=True) # Eliminar los bordes de abajo y de la izquierda
plt.tight_layout() # Ajustar el espaciado

plt.show()
```

Frecuencia Agrupada para la variable BIOMASA

```
import matplotlib.pyplot as plt
import seaborn as sns

#Paso 1. Asegurarse que no hay valores nulos,
#por si gubo fallos en el data clean
Biomasa = df['BIOMASA'].dropna()

#Paso 2. Calcular el numero de clases con la regla de Sturges
n = len(Biomasa)
k = int(np.ceil(1 + 3.322 * np.log10(n)))

#Paso 3. Crear intervalos con 2 decimales
min_val = Biomasa.min()
max_val = Biomasa.max()
bins = np.linspace(min_val, max_val, k + 1)
bins = np.round(bins, 2)

#Paso 4. Agrupar las clases
clases_biomasa = pd.cut(Biomasa, bins=bins, right=False)
freq_absBIOMASA = clases_biomasa.value_counts().sort_index()
freq_relBIOMASA = (freq_absBIOMASA / len(Biomasa) * 100).round(2)
freq_acumBIOMASA = freq_absBIOMASA.cumsum()
freq_rel_acumBIOMASA = freq_relBIOMASA.cumsum().round(2)

#Paso 5. Crear la Tabla
tablaBIOMASA = pd.DataFrame({'BIOMASA (Intervalos)': freq_absBIOMASA.index,
                             'Frecuencia Absoluta': freq_absBIOMASA.values,
                             'Frecuencia Relativa (%)': freq_relBIOMASA.values,
                             'Frecuencia Acumulada': freq_rel_acumBIOMASA.values})

#Mostramos tabla
display(tablaBIOMASA)
```

Mostrar el resultado oculto

```
import matplotlib.pyplot as plt
import seaborn as sns

#CREAMOS EL GRAFICO DE BARRAS horizontal para tablaBIOMASA
plt.figure(figsize=(12, 8)) # Ajustar el tamaño del gráfico para una mejor visualización
ax = sns.barplot(y=tablaBIOMASA['BIOMASA (Intervalos)'].astype(str), # Eje Y para barras horizontales
                 x=tablaBIOMASA['Frecuencia Relativa (%)'], color='#15297c') # Eje X

#Etiquetas dentro de cada barra
for i, v in enumerate(tablaBIOMASA['Frecuencia Relativa (%)']):
    ax.text(v - 2, i, f'{v:.2f}%', # Posición x ajustada para estar dentro de la barra
            color='white', va='center', ha='right', fontsize=10) # Color blanco y alineación a la derecha
```

```python
import matplotlib.pyplot as plt
import seaborn as sns

#Paso 1. Asegurarse que no hay valores nulos
PESO_INICIAL_GRAMOS = df['PESO INICIAL GRAMOS'].dropna()

#Paso 2. Calcular el numero de clases con la regla de Sturges
n = len(PESO_INICIAL_GRAMOS)
k = int(np.ceil(1 + 3.322 * np.log10(n)))

#Paso 3. Crear intervalos con 2 decimales
min_val = PESO_INICIAL_GRAMOS.min()
max_val = PESO_INICIAL_GRAMOS.max()
bins = np.linspace(min_val, max_val, k + 1)
bins = np.round(bins, 2)

#Paso 4. Agrupar las clases
clases_peso_inicial = pd.cut(PESO_INICIAL_GRAMOS, bins=bins, right=False)
freq_absPIG = clases_peso_inicial.value_counts().sort_index()
freq_relPIG = (freq_absPIG / len(PESO_INICIAL_GRAMOS) * 100).round(2)
freq_acumPIG = freq_absPIG.cumsum()
freq_rel_acumPIG = freq_relPIG.cumsum().round(2)

#Paso 5 y 6. Crear la Tabla
tablaPESO_INICIAL = pd.DataFrame({'PESO INICIAL GRAMOS (Intervalos)': freq_absPIG.index,
                          'Frecuencia Absoluta': freq_absPIG.values,
                          'Frecuencia Relativa (%)': freq_relPIG.values,
                          'Frecuencia Acumulada': freq_acumPIG.values,
                          'Frecuencia Relativa Acumulada (%)': freq_rel_acumPIG.values})

#Paso 7. Mostramos tabla
display(tablaPESO_INICIAL)
```

Mostrar el resultado oculto

```python
import matplotlib.pyplot as plt
import seaborn as sns

#Paso 8. Generar un gráfico de barras horizontal para tablaPESO_INICIAL
plt.figure(figsize=(12, 8)) # Ajustar el tamaño del gráfico para una mejor visualización
ax = sns.barplot(y=tablaPESO_INICIAL['PESO INICIAL GRAMOS (Intervalos)'].astype(str), # Eje Y para barras horizontales
                 x=tablaPESO_INICIAL['Frecuencia Relativa (%)'], color='#15297c') # Eje X

#Paso 9 y 10. Etiquetas dentro de cada barra mostrando el porcentaje
for i, v in enumerate(tablaPESO_INICIAL['Frecuencia Relativa (%)']):
    ax.text(v - 2, i, f'{v:.2f}%', # Posición x ajustada para estar dentro de la barra
            color='white', va='center', ha='right', fontsize=10) # Color blanco y alineación a la derecha

#Paso 11. Personalizacion estetica
plt.title('Frecuencia por Intervalos de PESO INICIAL GRAMOS')
plt.xlabel('Frecuencia Relativa (%)') # Etiqueta del eje X
```

```
#Paso 11. Personalizacion estetica
plt.title('Frecuencia por Intervalos de PESO INICIAL GRAMOS')
plt.xlabel('Frecuencia Relativa (%)') # Etiqueta del eje X
plt.ylabel('PESO INICIAL GRAMOS (Intervalos)') # Etiqueta del eje Y

#Paso 12. Eliminar los ticks del eje X y los bordes
ax.set_xticks([])     # Eliminar el eje X para que solo se vean los porcentajes dentro de las barras
sns.despine(bottom=True, left=True) # Eliminar los bordes de abajo y de la izquierda

#Paso 13. Ajustar el diseño y mostrar el plot
plt.tight_layout() # Ajustar el espaciado

plt.show()
```

Mostrar el resultado oculto

```
import matplotlib.pyplot as plt
import seaborn as sns

#Paso 1. Asegurarse que no hay valores nulos
PESO_FINAL_GRAMOS = df['PESO FINAL GRAMOS'].dropna()

#Paso 2. Calcular el numero de clases con la regla de Sturges
n = len(PESO_FINAL_GRAMOS)
k = int(np.ceil(1 + 3.322 * np.log10(n)))

#Paso 3. Crear intervalos con 2 decimales
min_val = PESO_FINAL_GRAMOS.min()
max_val = PESO_FINAL_GRAMOS.max()
bins = np.linspace(min_val, max_val, k + 1)
bins = np.round(bins, 2)

#Paso 4. Agrupar las clases
clases_peso_final = pd.cut(PESO_FINAL_GRAMOS, bins=bins, right=False)
freq_absPFG = clases_peso_final.value_counts().sort_index()
freq_relPFG = (freq_absPFG / len(PESO_FINAL_GRAMOS) * 100).round(2)
freq_acumPFG = freq_absPFG.cumsum()
freq_rel_acumPFG = freq_relPFG.cumsum().round(2)

#Paso 5 y 6. Crear la Tabla
tablaPESO_FINAL = pd.DataFrame({'PESO FINAL GRAMOS (Intervalos)': freq_absPFG.index,
                          'Frecuencia Absoluta': freq_absPFG.values,
                          'Frecuencia Relativa (%)': freq_relPFG.values,
                          'Frecuencia Acumulada': freq_acumPFG.values,
                          'Frecuencia Relativa Acumulada (%)': freq_rel_acumPFG.values})

#Paso 7. Mostramos tabla
display(tablaPESO_FINAL)
```

```
import matplotlib.pyplot as plt
import seaborn as sns

# Generar un gráfico de barras horizontal para tablaPESO_FINAL
plt.figure(figsize=(12, 8)) # Ajustar el tamaño del gráfico para una mejor visualización
ax = sns.barplot(y=tablaPESO_FINAL['PESO FINAL GRAMOS (Intervalos)'].astype(str), # Eje Y para barras horizontales
                 x=tablaPESO_FINAL['Frecuencia Relativa (%)'], color='#15297c') # Eje X

# Etiquetas dentro de cada barra mostrando el porcentaje
for i, v in enumerate(tablaPESO_FINAL['Frecuencia Relativa (%)']):
    ax.text(v - 2, i, f'{v:.2f}%', # Posición x ajustada para estar dentro de la barra
            color='white', va='center', ha='right', fontsize=10) # Color blanco y alineación a la derecha

# Personalizacion estetica
plt.title('Frecuencia por Intervalos de PESO FINAL GRAMOS')
plt.xlabel('Frecuencia Relativa (%)') # Etiqueta del eje X
plt.ylabel('PESO FINAL GRAMOS (Intervalos)') # Etiqueta del eje Y

# Eliminar los ticks del eje X y los bordes
ax.set_xticks([])     # Eliminar el eje X para que solo se vean los porcentajes dentro de las barras
sns.despine(bottom=True, left=True) # Eliminar los bordes de abajo y de la izquierda

# Ajustar el diseño y mostrar el plot
plt.tight_layout() # Ajustar el espaciado

plt.show()
```

Mostrar el resultado oculto

```
import matplotlib.pyplot as plt
import seaborn as sns

#Paso 1. Asegurarse que no hay valores nulos
DENSIDAD = df['DENSIDAD'].dropna()

#Paso 2. Calcular el numero de clases con la regla de Sturges
n = len(DENSIDAD)
k = int(np.ceil(1 + 3.322 * np.log10(n)))

#Paso 3. Crear intervalos con 2 decimales
min_val = DENSIDAD.min()
max_val = DENSIDAD.max()
bins = np.linspace(min_val, max_val, k + 1)
bins = np.round(bins, 2)

#Paso 4. Agrupar las clases
clases_densidad = pd.cut(DENSIDAD, bins=bins, right=False)
freq_absD = clases_densidad.value_counts().sort_index()
freq_relD = (freq_absD / len(DENSIDAD) * 100).round(2)
freq_acumD = freq_absD.cumsum()
freq_rel_acumD = freq_relD.cumsum().round(2)
```

```
#Paso 4. Agrupar las clases
clases_densidad = pd.cut(DENSIDAD, bins=bins, right=False)
freq_absD = clases_densidad.value_counts().sort_index()
freq_relD = (freq_absD / len(DENSIDAD) * 100).round(2)
freq_acumD = freq_absD.cumsum()
freq_rel_acumD = freq_relD.cumsum().round(2)

#Paso 5 y 6. Crear la Tabla
tablaDENSIDAD = pd.DataFrame({'DENSIDAD (Intervalos)': freq_absD.index,
                          'Frecuencia Absoluta': freq_absD.values,
                          'Frecuencia Relativa (%)': freq_relD.values,
                          'Frecuencia Acumulada': freq_acumD.values,
                          'Frecuencia Relativa Acumulada (%)': freq_rel_acumD.values})

#Paso 7. Mostramos tabla
display(tablaDENSIDAD)
```

Mostrar el resultado oculto

```
import matplotlib.pyplot as plt
import seaborn as sns

# Generar un gráfico de barras horizontal para tablaDENSIDAD
plt.figure(figsize=(12, 8)) # Ajustar el tamaño del gráfico para una mejor visualización
ax = sns.barplot(y=tablaDENSIDAD['DENSIDAD (Intervalos)'].astype(str), # Eje Y para barras horizontales
                 x=tablaDENSIDAD['Frecuencia Relativa (%)'], color='#15297c') # Eje X

# Etiquetas dentro de cada barra mostrando el porcentaje
for i, v in enumerate(tablaDENSIDAD['Frecuencia Relativa (%)']):
    ax.text(v - 2, i, f'{v:.2f}%', # Posición x ajustada para estar dentro de la barra
            color='white', va='center', ha='right', fontsize=10) # Color blanco y alineación a la derecha

# Personalizacion estetica
plt.title('Frecuencia por Intervalos de DENSIDAD')
plt.xlabel('Frecuencia Relativa (%)') # Etiqueta del eje X
plt.ylabel('DENSIDAD (Intervalos)') # Etiqueta del eje Y

# Eliminar los ticks del eje X y los bordes
ax.set_xticks([])     # Eliminar el eje X para que solo se vean los porcentajes dentro de las barras
sns.despine(bottom=True, left=True) # Eliminar los bordes de abajo y de la izquierda

# Ajustar el diseño y mostrar el plot
plt.tight_layout() # Ajustar el espaciado

plt.show()
```

```python
import matplotlib.pyplot as plt
import seaborn as sns

#Paso 1. Asegurarse que no hay valores nulos
ANM_TOTALES = df['Anm Totales  animales totales'].dropna()

#Paso 2. Calcular el numero de clases con la regla de Sturges
n = len(ANM_TOTALES)
k = int(np.ceil(1 + 3.322 * np.log10(n)))

#Paso 3. Crear intervalos con 2 decimales
min_val = ANM_TOTALES.min()
max_val = ANM_TOTALES.max()
bins = np.linspace(min_val, max_val, k + 1)
bins = np.round(bins, 2)

#Paso 4. Agrupar las clases
clases_anm_totales = pd.cut(ANM_TOTALES, bins=bins, right=False)
freq_absAT = clases_anm_totales.value_counts().sort_index()
freq_relAT = (freq_absAT / len(ANM_TOTALES) * 100).round(2)
freq_acumAT = freq_absAT.cumsum()
freq_rel_acumAT = freq_relAT.cumsum().round(2)

#Paso 5 y 6. Crear la Tabla
tablaANM_TOTALES = pd.DataFrame({'Anm Totales (Intervalos)': freq_absAT.index,
                         'Frecuencia Absoluta': freq_absAT.values,
                         'Frecuencia Relativa (%)': freq_relAT.values,
                         'Frecuencia Acumulada': freq_acumAT.values,
                         'Frecuencia Relativa Acumulada (%)': freq_rel_acumAT.values})

#Paso 7. Mostramos tabla
display(tablaANM_TOTALES)
```

Mostrar el resultado oculto

```python
import matplotlib.pyplot as plt
import seaborn as sns

# Generar un gráfico de barras horizontal para tablaANM_TOTALES
plt.figure(figsize=(12, 8)) # Ajustar el tamaño del gráfico para una mejor visualización
ax = sns.barplot(y=tablaANM_TOTALES['Anm Totales (Intervalos)'].astype(str), # Eje Y para barras horizontales
                 x=tablaANM_TOTALES['Frecuencia Relativa (%)'], color='#15297c') # Eje X

# Etiquetas dentro de cada barra mostrando el porcentaje
for i, v in enumerate(tablaANM_TOTALES['Frecuencia Relativa (%)']):
    ax.text(v - 2, i, f'{v:.2f}%', # Posición x ajustada para estar dentro de la barra
            color='white', va='center', ha='right', fontsize=10) # Color blanco y alineación a la derecha

# Personalizacion estetica
plt.title('Frecuencia por Intervalos de Anm Totales animales totales')
```

```python
import matplotlib.pyplot as plt
import seaborn as sns

#Paso 1. Asegurarse que no hay valores nulos
ANM_TOTALES = df['Anm Totales  animales totales'].dropna()

#Paso 2. Calcular el numero de clases con la regla de Sturges
n = len(ANM_TOTALES)
k = int(np.ceil(1 + 3.322 * np.log10(n)))

#Paso 3. Crear intervalos con 2 decimales
min_val = ANM_TOTALES.min()
max_val = ANM_TOTALES.max()
bins = np.linspace(min_val, max_val, k + 1)
bins = np.round(bins, 2)

#Paso 4. Agrupar las clases
clases_anm_totales = pd.cut(ANM_TOTALES, bins=bins, right=False)
freq_absAT = clases_anm_totales.value_counts().sort_index()
freq_relAT = (freq_absAT / len(ANM_TOTALES) * 100).round(2)
freq_acumAT = freq_absAT.cumsum()
freq_rel_acumAT = freq_relAT.cumsum().round(2)

#Paso 5 y 6. Crear la Tabla
tablaANM_TOTALES = pd.DataFrame({'Anm Totales (Intervalos)': freq_absAT.index,
                         'Frecuencia Absoluta': freq_absAT.values,
                         'Frecuencia Relativa (%)': freq_relAT.values,
                         'Frecuencia Acumulada': freq_acumAT.values,
                         'Frecuencia Relativa Acumulada (%)': freq_rel_acumAT.values})

#Paso 7. Mostramos tabla
display(tablaANM_TOTALES)
```

Mostrar el resultado oculto

```python
import matplotlib.pyplot as plt
import seaborn as sns

# Generar un gráfico de barras horizontal para tablaANM_TOTALES
plt.figure(figsize=(12, 8)) # Ajustar el tamaño del gráfico para una mejor visualización
ax = sns.barplot(y=tablaANM_TOTALES['Anm Totales (Intervalos)'].astype(str), # Eje Y para barras horizontales
                 x=tablaANM_TOTALES['Frecuencia Relativa (%)'], color='#15297c') # Eje X

# Etiquetas dentro de cada barra mostrando el porcentaje
for i, v in enumerate(tablaANM_TOTALES['Frecuencia Relativa (%)']):
    ax.text(v - 2, i, f'{v:.2f}%', # Posición x ajustada para estar dentro de la barra
            color='white', va='center', ha='right', fontsize=10) # Color blanco y alineación a la derecha
```

```
# Etiquetas dentro de cada barra mostrando el porcentaje
for i, v in enumerate(tablaANM_TOTALES['Frecuencia Relativa (%)']):
    ax.text(v - 2, i, f'{v:.2f}%', # Posición x ajustada para estar dentro de la barra
            color='white', va='center', ha='right', fontsize=10) # Color blanco y alineación a la derecha

# Personalizacion estetica
plt.title('Frecuencia por Intervalos de Anm Totales animales totales')
plt.xlabel('Frecuencia Relativa (%)') # Etiqueta del eje X
plt.ylabel('Anm Totales animales totales (Intervalos)') # Etiqueta del eje Y

# Eliminar los ticks del eje X y los bordes
ax.set_xticks([])     # Eliminar el eje X para que solo se vean los porcentajes dentro de las barras
sns.despine(bottom=True, left=True) # Eliminar los bordes de abajo y de la izquierda

# Ajustar el diseño y mostrar el plot
plt.tight_layout() # Ajustar el espaciado

plt.show()
```

Mostrar el resultado oculto

```
import matplotlib.pyplot as plt
import seaborn as sns

#Paso 1. Asegurarse que no hay valores nulos
DIAS_PRECRIA = df['DÍAS DE PRECRIA'].dropna()

#Paso 2. Calcular el numero de clases con la regla de Sturges
n = len(DIAS_PRECRIA)
k = int(np.ceil(1 + 3.322 * np.log10(n)))

#Paso 3. Crear intervalos con 2 decimales
min_val = DIAS_PRECRIA.min()
max_val = DIAS_PRECRIA.max()
bins = np.linspace(min_val, max_val, k + 1)
bins = np.round(bins, 2)

#Paso 4. Agrupar las clases
clases_dias_precria = pd.cut(DIAS_PRECRIA, bins=bins, right=False)
freq_absDDP = clases_dias_precria.value_counts().sort_index()
freq_relDDP = (freq_absDDP / len(DIAS_PRECRIA) * 100).round(2)
freq_acumDDP = freq_absDDP.cumsum()
freq_rel_acumDDP = freq_relDDP.cumsum().round(2)

#Paso 5 y 6. Crear la Tabla
tablaDIAS_PRECRIA = pd.DataFrame({'DÍAS DE PRECRIA (Intervalos)': freq_absDDP.index,
                          'Frecuencia Absoluta': freq_absDDP.values,
                          'Frecuencia Relativa (%)': freq_relDDP.values,
                          'Frecuencia Acumulada': freq_acumDDP.values,
                          'Frecuencia Relativa Acumulada (%)': freq_rel_acumDDP.values})
```

```
import matplotlib.pyplot as plt
import seaborn as sns

# Generar un gráfico de barras horizontal para tablaDIAS_PRECRIA
plt.figure(figsize=(12, 8)) # Ajustar el tamaño del gráfico para una mejor visualización
ax = sns.barplot(y=tablaDIAS_PRECRIA['DÍAS DE PRECRIA (Intervalos)'].astype(str), # Eje Y para barras horizontales
                 x=tablaDIAS_PRECRIA['Frecuencia Relativa (%)'], color='#15297c') # Eje X

# Etiquetas dentro de cada barra mostrando el porcentaje
for i, v in enumerate(tablaDIAS_PRECRIA['Frecuencia Relativa (%)']):
    ax.text(v - 2, i, f'{v:.2f}%', # Posición x ajustada para estar dentro de la barra
            color='white', va='center', ha='right', fontsize=10) # Color blanco y alineación a la derecha

# Personalizacion estetica
plt.title('Frecuencia por Intervalos de DÍAS DE PRECRIA')
plt.xlabel('Frecuencia Relativa (%)') # Etiqueta del eje X
plt.ylabel('DÍAS DE PRECRIA (Intervalos)') # Etiqueta del eje Y

# Eliminar los ticks del eje X y los bordes
ax.set_xticks([])     # Eliminar el eje X para que solo se vean los porcentajes dentro de las barras
sns.despine(bottom=True, left=True) # Eliminar los bordes de abajo y de la izquierda

# Ajustar el diseño y mostrar el plot
plt.tight_layout() # Ajustar el espaciado

plt.show()
```

Mostrar el resultado oculto

```
import matplotlib.pyplot as plt
import seaborn as sns

#Paso 1. Asegurarse que no hay valores nulos
KG_HA_TOTALES = df['KG * HA TOTALES'].dropna()

#Paso 2. Calcular el numero de clases con la regla de Sturges
n = len(KG_HA_TOTALES)
k = int(np.ceil(1 + 3.322 * np.log10(n)))

#Paso 3. Crear intervalos con 2 decimales
min_val = KG_HA_TOTALES.min()
max_val = KG_HA_TOTALES.max()
bins = np.linspace(min_val, max_val, k + 1)
bins = np.round(bins, 2)

#Paso 4. Agrupar las clases
clases_kg_ha_totales = pd.cut(KG_HA_TOTALES, bins=bins, right=False)
freq_absKHT = clases_kg_ha_totales.value_counts().sort_index()
freq_relKHT = (freq_absKHT / len(KG_HA_TOTALES) * 100).round(2)
freq_acumKHT = freq_absKHT.cumsum()
freq_rel_acumKHT = freq_relKHT.cumsum().round(2)
```

```python
#Paso 5 y 6. Crear la Tabla
tablaKG_HA_TOTALES = pd.DataFrame({'KG * HA TOTALES (Intervalos)': freq_absKHT.index,
                        'Frecuencia Absoluta': freq_absKHT.values,
                        'Frecuencia Relativa (%)': freq_relKHT.values,
                        'Frecuencia Acumulada': freq_acumKHT.values,
                        'Frecuencia Relativa Acumulada (%)': freq_rel_acumKHT.values})

#Paso 7. Mostramos tabla
display(tablaKG_HA_TOTALES)
```

Mostrar el resultado oculto

```python
import matplotlib.pyplot as plt
import seaborn as sns

# Generar un gráfico de barras horizontal para tablaKG_HA_TOTALES
plt.figure(figsize=(12, 8)) # Ajustar el tamaño del gráfico para una mejor visualización
ax = sns.barplot(y=tablaKG_HA_TOTALES['KG * HA TOTALES (Intervalos)'].astype(str), # Eje Y para barras horizontales
                 x=tablaKG_HA_TOTALES['Frecuencia Relativa (%)'], color='#15297c') # Eje X

# Etiquetas dentro de cada barra mostrando el porcentaje
for i, v in enumerate(tablaKG_HA_TOTALES['Frecuencia Relativa (%)']):
    ax.text(v - 2, i, f'{v:.2f}%', # Posición x ajustada para estar dentro de la barra
            color='white', va='center', ha='right', fontsize=10) # Color blanco y alineación a la derecha

# Personalizacion estetica
plt.title('Frecuencia por Intervalos de KG * HA TOTALES')
plt.xlabel('Frecuencia Relativa (%)') # Etiqueta del eje X
plt.ylabel('KG * HA TOTALES (Intervalos)') # Etiqueta del eje Y

# Eliminar los ticks del eje X y los bordes
ax.set_xticks([])     # Eliminar el eje X para que solo se vean los porcentajes dentro de las barras
sns.despine(bottom=True, left=True) # Eliminar los bordes de abajo y de la izquierda

# Ajustar el diseño y mostrar el plot
plt.tight_layout() # Ajustar el espaciado

plt.show()
```

Mostrar el resultado oculto

```python
import matplotlib.pyplot as plt
import seaborn as sns

#Paso 1. Asegurarse que no hay valores nulos
PROMEDIO_CRECIMIENTO_SEMANAL = df['PROMEDIO CRECIMIENTO SEMANAL'].dropna()
```

```
#Paso 3. Crear intervalos con 2 decimales
min_val = PROMEDIO_CRECIMIENTO_SEMANAL.min()
max_val = PROMEDIO_CRECIMIENTO_SEMANAL.max()
bins = np.linspace(min_val, max_val, k + 1)
bins = np.round(bins, 2)

#Paso 4. Agrupar las clases
clases_promedio_crecimiento = pd.cut(PROMEDIO_CRECIMIENTO_SEMANAL, bins=bins, right=False)
freq_absPCS = clases_promedio_crecimiento.value_counts().sort_index()
freq_relPCS = (freq_absPCS / len(PROMEDIO_CRECIMIENTO_SEMANAL) * 100).round(2)
freq_acumPCS = freq_absPCS.cumsum()
freq_rel_acumPCS = freq_relPCS.cumsum().round(2)

#Paso 5 y 6. Crear la Tabla
tablaPROMEDIO_CRECIMIENTO = pd.DataFrame({'PROMEDIO CRECIMIENTO SEMANAL (Intervalos)': freq_absPCS.index,
                         'Frecuencia Absoluta': freq_absPCS.values,
                         'Frecuencia Relativa (%)': freq_relPCS.values,
                         'Frecuencia Acumulada': freq_acumPCS.values,
                         'Frecuencia Relativa Acumulada (%)': freq_rel_acumPCS.values})

#Paso 7. Mostramos tabla
display(tablaPROMEDIO_CRECIMIENTO)
```

Mostrar el resultado oculto

```
import matplotlib.pyplot as plt
import seaborn as sns

# Generar un gráfico de barras horizontal para tablaPROMEDIO_CRECIMIENTO
plt.figure(figsize=(12, 8)) # Ajustar el tamaño del gráfico para una mejor visualización
ax = sns.barplot(y=tablaPROMEDIO_CRECIMIENTO['PROMEDIO CRECIMIENTO SEMANAL (Intervalos)'].astype(str), # Eje Y para barras horizontales
                 x=tablaPROMEDIO_CRECIMIENTO['Frecuencia Relativa (%)'], color='#15297c') # Eje X

# Etiquetas dentro de cada barra mostrando el porcentaje
for i, v in enumerate(tablaPROMEDIO_CRECIMIENTO['Frecuencia Relativa (%)']):
    ax.text(v - 2, i, f'{v:.2f}%', # Posición x ajustada para estar dentro de la barra
            color='white', va='center', ha='right', fontsize=10) # Color blanco y alineación a la derecha

# Personalizacion estetica
plt.title('Frecuencia por Intervalos de PROMEDIO CRECIMIENTO SEMANAL')
plt.xlabel('Frecuencia Relativa (%)') # Etiqueta del eje X
plt.ylabel('PROMEDIO CRECIMIENTO SEMANAL (Intervalos)') # Etiqueta del eje Y

# Eliminar los ticks del eje X y los bordes
ax.set_xticks([])     # Eliminar el eje X para que solo se vean los porcentajes dentro de las barras
sns.despine(bottom=True, left=True) # Eliminar los bordes de abajo y de la izquierda

# Ajustar el diseño y mostrar el plot
plt.tight_layout() # Ajustar el espaciado

plt.show()
```

[1]
```python
#Importamos las librerias a usar
import pandas as pd #Esta librería sirve para manipulación de datos
import numpy as np #Esta librería sirve para trabajar con valores nulos o perdidos
import matplotlib.pyplot as plt #Esta librería sirve para graficar
import seaborn as sns #Esta librería sirve para graficar
```

[2] 12 s
```python
#CARGAR LOS DATOS A COLAB#
from google.colab import files
uploaded = files.upload()
```

```
Elegir archivos Sin archivos seleccionados Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.
Saving DATOS CAMARON final.xlsx to DATOS CAMARON final.xlsx
```

[3] 1 s
```python
#LEER EL ARCHIVO DE DATOS
df = pd.read_excel("DATOS CAMARON final.xlsx", header=1)
df.head()
```

Mostrar el resultado oculto

[4] 0 s
```python
dfcolumns = df.columns.str.strip() # Clean column names by stripping whitespace

columns_to_drop = ['ZONA']
for i in range(1, 14):
    columns_to_drop.append(f'PESO SEMA {i}')
    columns_to_drop.append(f'SUPERV SEMANA {i}')
for i in range(1, 15):
    columns_to_drop.append(f'semana {i}')

existing_columns_to_drop = [col for col in columns_to_drop if col in df.columns]

df_dropped = df.drop(columns=existing_columns_to_drop, axis=1)
display(df_dropped.head())
```

Mostrar el resultado oculto

[ ]
```python
df_dropped.info()
```

Mostrar el resultado oculto

[5] 0 s
```python
#Boxplot FINCA Y PESO FINAL
plt.figure(figsize=(10, 6))
sns.boxplot(x='FINCA', y='PESO FINAL GRAMOS', data=df, color='#00BFFF')
plt.title('Boxplot PESO FINAL por FINCA')
plt.xlabel('FINCA')
plt.ylabel('PESO FINAL')
plt.show()
```

```python
#VIOLINPLOT, ES UN SUSTITUO DE BOXPLOT, TE PUEDE ENSEÑAR QUE TANTO SE PUEDE AJUSTAR UN GRAFICO SEGUN SU TIPO de datos y de lectura
#Habitacion compartida en relacion satisfaccion total del cliente
plt.figure(figsize=(10, 6))
sns.violinplot(x='FINCA', y='DIAS TOTALES', data=df, color='#008000')
plt.title('Violinplot FINCA vs DIAS TOTALES')
plt.xlabel('FINCA')
plt.ylabel('DIAS TOTALES')
plt.grid(True)
plt.show()
```

Mostrar el resultado oculto

```python
plt.figure(figsize=(10, 6))
sns.scatterplot(x='DÍAS DE PRECRIA', y='PESO FINAL GRAMOS',hue = 'SOBREVIVE FINAL (POR ENCIMA DE LA MEDIANA DE 70,08%)',
                data=df, color='#FF0000')
```

Mostrar el resultado oculto

```python
plt.figure(figsize=(10, 6))
sns.scatterplot(x='SUPER FINAL %', y='DÍAS DE PRECRIA', hue= 'TIPO DE ALIMENTO',
                data=df, color='#FF0000')
```

Mostrar el resultado oculto

```python
plt.figure(figsize=(10, 6))
sns.scatterplot(x='PESO FINAL GRAMOS', y='PESO INICIAL GRAMOS', hue= 'TIPO DE ALIMENTO',
                data=df, color='#FF0000')
```

Mostrar el resultado oculto

```python
plt.figure(figsize=(10, 6))
sns.scatterplot(x='KG * HA TOTALES', y='PROMEDIO CRECIMIENTO SEMANAL',
                data=df, color='#FF0000')
```

Mostrar el resultado oculto

Anexo 4 Script Modelo en Python

```python
# Cargar datos
df = pd.read_excel("DATOS CAMARON final.xlsx", header=1)
```

```python
# Seleccionar columnas relevantes
cols = [
    "COD GEN",
    "TIPO DE ALIMENTO",
    "LABORATORIO",
    "DENSIDAD",
    "HP/HA",
    "DIAS TOTALES",
    "PESO FINAL GRAMOS",
    "SOBREVIVE FINAL (POR ENCIMA DE LA MEDIANA DE 70,08%)"
]

df = df[cols].copy()
```

```python
# Eliminar filas sin variable objetivo o con datos críticos faltantes
df = df.dropna(subset=["SOBREVIVE FINAL (POR ENCIMA DE LA MEDIANA DE 70,08%)", "DENSIDAD", "HP/HA", "DIAS TOTALES"])
df["SOBREVIVE FINAL (POR ENCIMA DE LA MEDIANA DE 70,08%)"] = df["SOBREVIVE FINAL (POR ENCIMA DE LA MEDIANA DE 70,08%)"].astype(int)
print(f"Filas válidas para modelado: {len(df)}")
```

```
Filas válidas para modelado: 109
```

Regresión logística

```python
# Codificar variables categóricas
df_encoded = pd.get_dummies(
    df,
    columns=["COD GEN", "TIPO DE ALIMENTO", "LABORATORIO"],
    drop_first=True,
    dtype=int
)
```

```python
# Definir X e y
y = df_encoded["SOBREVIVE FINAL (POR ENCIMA DE LA MEDIANA DE 70,08%)"]
X = df_encoded.drop(columns=["SOBREVIVE FINAL (POR ENCIMA DE LA MEDIANA DE 70,08%)"])
```

```python
# Dividir datos
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42, stratify=y)
```

```python
# Entrenar modelo SIN escalado
model = LogisticRegression(max_iter=1000, random_state=42, class_weight='balanced')
```

```python
# Entrenar modelo SIN escalado
model = LogisticRegression(max_iter=1000, random_state=42, class_weight='balanced')

# Filtrar solo columnas numéricas para X_train y X_test.
# Esto elimina las columnas de tipo 'object' (cadenas) que no fueron codificadas
# (e.g., 'PISCINA', 'ZONA', 'FINCA', 'FECHA DE SIEMBRA') y que causan el ValueError.
X_train = X_train.select_dtypes(include=[np.number])
X_test = X_test.select_dtypes(include=[np.number])

model.fit(X_train, y_train)
```

Mostrar el resultado oculto

```python
# Evaluar
y_pred = model.predict(X_test)
y_proba = model.predict_proba(X_test)[:, 1]

print("📊 Métricas del modelo (sin escalado):")
print(classification_report(y_test, y_pred, target_names=["<70.08%", "≥70.08%"]))
print(f"AUC-ROC: {roc_auc_score(y_test, y_proba):.3f}")
print(f"Accuracy: {model.score(X_test, y_test):.3f}")
print(f"Precision: {model.score(X_test, y_test):.3f}")
print(f"Recall: {model.score(X_test, y_test):.3f}")
print(f"F1-score: {model.score(X_test, y_test):.3f}")
```

Mostrar el resultado oculto

```python
plt.figure(figsize=(6, 5))
sns.heatmap(confusion_matrix(y_test, y_pred), annot=True, fmt='d', cmap='Blues',
            xticklabels=["<70.08%", "≥70.08%"], yticklabels=["<70.08%", "≥70.08%"])
plt.title("Matriz de confusión")
plt.ylabel("Real")
plt.xlabel("Predicho")
plt.show()
```

Mostrar el resultado oculto

```python
# Curva ROC
fpr, tpr, _ = roc_curve(y_test, y_proba)
plt.figure(figsize=(6, 5))
plt.plot(fpr, tpr, color='darkorange', lw=2, label=f'ROC (AUC = {roc_auc_score(y_test, y_proba):.3f})')
plt.plot([0,1], [0,1], color='navy', lw=1, linestyle='--')
plt.xlabel('Tasa de falsos positivos')
plt.ylabel('Tasa de verdaderos positivos')
plt.title('Curva ROC')
plt.legend()
plt.grid(True)
plt.show()
```

```python
# Coeficientes
coef_df = pd.DataFrame({
    'Variable': X_train.columns, # Changed from X.columns to X_train.columns
    'Coeficiente (β)': model.coef_[0]
}).sort_values(by='Coeficiente (β)', key=abs, ascending=False)

print("\n🔍 Ecuación logística estimada:")
print("logit(P) = β₀ + β₁·X₁ + β₂·X₂ + ...")
print(f"Intercepto (β₀): {model.intercept_[0]:.4f}\n")

print("Principales coeficientes (valores originales):")
print(coef_df.head(15).to_string(index=False))
```

Mostrar el resultado oculto

```python
# Extraer niveles únicos para widgets
from IPython.display import clear_output

# Opciones únicas
gen_opts = sorted(df["COD GEN"].dropna().unique())
alim_opts = sorted(df["TIPO DE ALIMENTO"].dropna().unique())
lab_opts = sorted(df["LABORATORIO"].dropna().unique())

# Widgets (valores en escala natural)
gen_w = widgets.Dropdown(options=gen_opts, description="Genética:")
alim_w = widgets.Dropdown(options=alim_opts, description="Alimento:")
lab_w = widgets.Dropdown(options=lab_opts, description="Laboratorio:")
dens_w = widgets.FloatSlider(value=269, min=72, max=472, step=5, description="Densidad (PL/m²):")
hp_w = widgets.FloatSlider(value=7.4, min=4.5, max=16.7, step=0.5, description="HP/HA (kg/ha):")
dias_w = widgets.IntSlider(value=70, min=50, max=90, step=1, description="Días totales:")
peso_w = widgets.FloatSlider(value=25, min=10, max=35, step=0.5, description="Peso final (g):")

output = widgets.Output()

def calcular_supervivencia(b):
    with output:
        clear_output(wait=True)
        try:
            # Crear entrada SIN 'PESO FINAL GRAMOS' (no está en el modelo)
            input_df = pd.DataFrame({
                "COD GEN": [gen_w.value],
                "TIPO DE ALIMENTO": [alim_w.value],
                "LABORATORIO": [lab_w.value],
                "DENSIDAD": [dens_w.value],
                "HP/HA": [hp_w.value],
                "DIAS TOTALES": [dias_w.value]
                # ⚠️ PESO FINAL GRAMOS se excluye: no fue incluido en el modelo entrenado
            })

            # Codificar igual que en entrenamiento
            input_enc = pd.get_dummies(
```

```
            # Codificar igual que en entrenamiento
            input_enc = pd.get_dummies(
                input_df,
                columns=["COD GEN", "TIPO DE ALIMENTO", "LABORATORIO"],
                dtype=int
            )

            # Alinear EXACTAMENTE con las columnas usadas en el entrenamiento
            input_enc = input_enc.reindex(columns=X_train.columns, fill_value=0)

            # Predecir
            prob = model.predict_proba(input_enc)[0, 1]
            pred_label = "✅ Alta (≥70.08%)" if prob >= 0.5 else "❌ Baja (<70.08%)"

            print(f"🔹 Probabilidad de supervivencia ≥70.08%: {prob:.2%}")
            print(f"🔹 Clasificación: {pred_label}")

        except Exception as e:
            print("⚠️ Error en el cálculo.")
            print(f"Detalles: {e}")
            import traceback
            traceback.print_exc()

btn = widgets.Button(description="Calcular Supervivencia")
btn.on_click(calcular_supervivencia)

ui = widgets.VBox([
    widgets.HBox([gen_w, alim_w, lab_w]),
    widgets.HBox([dens_w, hp_w]),
    widgets.HBox([dias_w, peso_w]),
    btn,
    output
])

display(ui)
```

Genética: O-OMTP42GO(100%) Alimento: T Laboratorio: Mejia Luzuriaga Jorge Luis - La

Densidad (... 347.00 HP/HA (kg/... 7.40

Días totales: 66 Peso final (... 25.00

Calcular Supervive...

🔹 Probabilidad de supervivencia ≥70.08%: 99.71%
🔹 Clasificación: ✅ Alta (≥70.08%)

```
# ============================================================
# GENERADOR AUTOMÁTICO DE MEJORES COMBINACIONES
# ============================================================

import pandas as pd
import numpy as np

print("\n🔎 Generando combinaciones óptimas según el modelo...\n")

# Si existe umbral óptimo lo usa, si no usa 0.5
THRESHOLD = globals().get("best_threshold", 0.5)

# -------- 1️⃣ Valores reales para categorías ----------
cat_cols = ["COD GEN", "TIPO DE ALIMENTO", "LABORATORIO"]

cat_combos = (
    df[cat_cols]
    .dropna()
    .drop_duplicates()
    .reset_index(drop=True)
)

# -------- 2️⃣ Rango inteligente para variables numéricas ----------
dens_vals = np.linspace(df["DENSIDAD"].quantile(0.25),
                        df["DENSIDAD"].quantile(0.75), 3)

hpha_vals = np.linspace(df["HP/HA"].quantile(0.25),
                        df["HP/HA"].quantile(0.75), 3)

dias_vals = [int(df["DIAS TOTALES"].median())]
peso_vals = np.linspace(df["PESO FINAL GRAMOS"].quantile(0.50),
                        df["PESO FINAL GRAMOS"].quantile(0.90), 3)

# -------- 3️⃣ Crear escenarios ----------
scenarios = []

for _, row in cat_combos.iterrows():
    for d in dens_vals:
        for h in hpha_vals:
            for di in dias_vals:
                for p in peso_vals:
                    scenarios.append({
                        "COD GEN": row["COD GEN"],
                        "TIPO DE ALIMENTO": row["TIPO DE ALIMENTO"],
                        "LABORATORIO": row["LABORATORIO"],
                        "DENSIDAD": float(d),
                        "HP/HA": float(h),
```

```python
# -------- 3️⃣ Crear escenarios ----------
scenarios = []

for _, row in cat_combos.iterrows():
    for d in dens_vals:
        for h in hpha_vals:
            for di in dias_vals:
                for p in peso_vals:
                    scenarios.append({
                        "COD GEN": row["COD GEN"],
                        "TIPO DE ALIMENTO": row["TIPO DE ALIMENTO"],
                        "LABORATORIO": row["LABORATORIO"],
                        "DENSIDAD": float(d),
                        "HP/HA": float(h),
                        "DIAS TOTALES": int(di),
                        "PESO FINAL GRAMOS": float(p)
                    })

scenarios_df = pd.DataFrame(scenarios)

# -------- 4️⃣ Codificar igual que entrenamiento ----------
scenarios_enc = pd.get_dummies(
    scenarios_df,
    columns=["COD GEN", "TIPO DE ALIMENTO", "LABORATORIO"],
    dtype=int
)

scenarios_enc = scenarios_enc.reindex(columns=X_train.columns, fill_value=0)

# -------- 5️⃣ Predecir ----------
proba = model.predict_proba(scenarios_enc)[:, 1]
scenarios_df["PROBABILIDAD"] = proba
scenarios_df["CLASE"] = (proba >= THRESHOLD).astype(int)

# -------- 6️⃣ Ordenar y mostrar mejores ----------
top_results = (
    scenarios_df
    .sort_values("PROBABILIDAD", ascending=False)
    .head(20)
    .reset_index(drop=True)
)

print("⏳ TOP 20 MEJORES COMBINACIONES ESTIMADAS\n")
display(top_results)
```

DECLARACIÓN Y

AUTORIZACIÓN

Nosotros, **Acán Guamán, Luis Slayder** con **C.C: 0605410810** y **Franco Vásquez, Carlos Antonio** con C.C: **0926410648** autores del trabajo de titulación: **Análisis de supervivencia de la producción del camarón segmentado por genética y alimentación.** previo a la obtención del título de **Licenciado de Negocios Internacionales** en la Universidad Católica de Santiago de Guayaquil.

1.- Declaro tener pleno conocimiento de la obligación que tienen las instituciones de educación superior, de conformidad con el Artículo 144 de la Ley Orgánica de Educación Superior, de entregar a la SENESCYT en formato digital una copia del referido trabajo de titulación para que sea integrado al Sistema Nacional de Información de la Educación Superior del Ecuador para su difusión pública respetando los derechos de autor.

2.- Autorizo a la SENESCYT a tener una copia del referido trabajo de titulación, con el propósito de generar un repositorio que democratice la información, respetando las políticas de propiedad intelectual vigentes.

Guayaquil, **13 de febrero de 2026**

LOS AUTORES:

f._____

Acán Guamán, Luis Slayder

C.C: 0605410810

f._____

Franco Vásquez, Carlos Antonio

C.C: 0926410648

REPOSITORIO NACIONAL EN CIENCIA Y TECNOLOGÍA

FICHA DE REGISTRO DE TESIS/TRABAJO DE TITULACIÓN

TEMA Y SUBTEMA:	**Análisis de supervivencia de la producción del camarón segmentado por genética y alimentación.**		
AUTOR(ES)	**Acán Guamán, Luis Slayder** **Franco Vásquez, Carlos Antonio**		
REVISOR(ES)/TUTOR(ES)	**Carrera Buri, Félix Miguel Mgs**		
INSTITUCIÓN:	Universidad Católica de Santiago de Guayaquil		
FACULTAD:	**Facultad de ciencias económicas y empresariales**		
CARRERA:	**Negocios Internacionales**		
TITULO OBTENIDO:	**Licenciado en Negocios Internacionales**		
FECHA DE PUBLICACIÓN:	**13 de febrero de 2026**	**No. DE PÁGINAS:**	**155 p.**
ÁREAS TEMÁTICAS:	**Machine learning, Producción Acuícola, Inteligencia de Negocios, Economía agrícola, Ingesta de nutrientes, Nutrición acuícola.**		
PALABRAS CLAVES/ KEYWORDS:	**Códigos genéticos, Supervivencia, Regresión, clasificación, logística, genética, producción, alimentación.**		

RESUMEN/ABSTRACT: La investigación, denominada "Análisis de supervivencia de camarón segmentado por genética y alimentación", busca determinar la probabilidad de supervivencia del camarón en el sector productivo ecuatoriano según variables productivas como la genética y el tipo de alimento. Esta investigación surge de la necesidad de crear una herramienta que permita establecer cuáles son los mejores proveedores que, mediante una base histórica, logran que las líneas genéticas alcancen cierto peso con determinado tipo de alimentación, además de identificar cuánto sería su tiempo óptimo de cosecha.

Metodológicamente, el estudio se abordó desde un enfoque cuantitativo, de modelado supervisado, específicamente de regresión logística, para predecir la probabilidad de supervivencia del camarón a partir de datos históricos de producción. Para verificar los resultados, se utilizaron métricas como la matriz de confusión, accuracy, precisión, recall, F1 score, curva ROC y AUC, las cuales midieron el rendimiento del modelo.

Además, se incorporaron indicadores productivos que refuerzan el análisis técnico.

Los resultados indican que la supervivencia no está ligada a las formas de alimentación, sino a la combinación de la genética y el proveedor que las desarrolla. Este análisis hace evidente la necesidad de certificar tanto a los proveedores como a las genéticas con las que se ha trabajado.

ADJUNTO PDF:	☒ SI	☐ NO
CONTACTO CON AUTOR/ES:	**Teléfono:** +593-960086367 +593-982259539	E-mail: caranfra6@mail.com Slayderlg.lg3@gmail.com
CONTACTO CON LA INSTITUCIÓN (C00RDINADOR DEL PROCESO UTE)::	**Nombre: Freire Quintero Cesar Enrique**	
	Teléfono: +593 990090702	
	E-mail: cesar.freire@cu.ucsg.edu.ec	

SECCIÓN PARA USO DE BIBLIOTECA	
Nº. DE REGISTRO (en base a datos):	
Nº. DE CLASIFICACIÓN:	
DIRECCIÓN URL (tesis en la web):	