Universidad Nacional de Ingeniería

Facultad de Ingeniería Geológica Minera y Metalúrgica

TESIS

Construcción de un modelo predictivo de productividad de las palas aplicando el aprendizaje automático random forest regressor de machine learning

Para obtener el título profesional de Ingeniero de Minas

Elaborado por

Victor Alexander Juarez Racchumi

0009-0002-3865-7051

Asesor

M.Sc. Jose Antonio Corimanya Mauricio

0000-0003-1078-4155

LIMA – PERÚ

2023

Citar/How to cite	Juarez Racchumi [1]
Referencia/Reference Estilo/Style: IEEE (2020)	[1] V. Juarez Racchumi, "*Construcción de un modelo predictivo de productividad de las palas aplicando el aprendizaje automático random forest regressor de machine learning*" [Tesis de pregrado]. Lima (Perú): Universidad Nacional de Ingeniería, 2023.

Citar/How to cite	(Juarez, 2023)
Referencia/Reference Estilo/Style: APA (7ma ed.)	Juarez, V. (2023). *Construcción de un modelo predictivo de productividad de las palas aplicando el aprendizaje automático random forest regressor de machine learning*. [Tesis de pregrado, Universidad Nacional de Ingeniería]. Repositorio institucional Cybertesis UNI.

Dedicatoria

A Dios por siempre darme oportunidades de mejora en la vida.

A mi madre desde el cielo, a mi padre, hermanos, sobrino y familia que siempre están apoyándome en cada decisión y desafío que adopto en esta vida.

Agradecimientos

Quiero agradecer a mi padre, hermanos, a los docentes y personal administrativo de la Escuela de Minas de la Universidad Nacional de Ingeniería, quienes me han ayudado en mi formación académica.

A mi gran amigo el Ing. Diego Torres Francia y la Ing. Lily Ponce Gago, quienes, en épocas de pandemia, cuando estaba sin trabajo, me dieron la oportunidad de participar en la entrevista laboral, para el cargo de operador de Planta, a pesar de no ser Ingeniero Metalúrgico, ya que, gracias a esa experiencia laboral, aprendí muchas cosas y poco a poco se me fueron dando oportunidades laborales.

Y también debo agradecer a las diferentes empresas en cuales he laborado por la confianza que me dieron para desenvolverme profesionalmente.

Resumen

La Tesis se enfoca en la creación de un modelo predictivo de productividad de palas con un nivel de confianza aceptable, que a diferencia del método tradicional, incluye no solo las variables de tonelaje minado, tiempo de carguío y tiempo hang; sino que considera también otras variables cuantitativas y cualitativas como el tiempo queue, tiempo spot, tiempo viaje cargado, tiempo queue en el destino, tiempo spot en el destino, tiempo de descarga, tiempo de viaje vacío, guardia, turno, mes, flota de pala, pala, efh vacío, efh cargado, material, tipo de material, fase de origen, fase de destino, hora, disponibilidad de camiones y palas.

El objetivo principal es mejorar la predicción de la productividad de palas aplicando Machine Learning, así como determinar el mejor modelo de predicción de la productividad de palas y determinar el grado de importancia de cada una de las variables independientes para nuestra variable predictora, a través de varias librerías y con ayuda del Jupyter Notebook, asegurando que nuestros resultados sean los más confiables.

La investigación comenzará con la hipótesis general que el modelo de productividad desarrollado con Random Forest Regressor de Machine Learning mejora la predicción de la productividad de palas que otros modelos.

Para el modelo de machine learning se utilizó todas las variables presentes en el DataSet, incluyendo las variables cualitativas, las cuales fueron categorizadas antes del entrenamiento.

De los modelos de machine learning el modelo que mejor predice la productividad de las palas es el “Random Forest Regressor” con una precisión del 99.7% para el entrenamiento y 91.1% para el DataSet de testeo; mientras que el peor modelo de machine learning es “Dummy Regressor” con - 0.03% de precisión. Además, la variable de mayor grado de importancia es la Flota de pala con 24.92%.

Palabras claves — Machine learning, rendom Forest, productividad, entrenamiento.

Abstract

The Thesis focuses is the creation of a predictive model of shovel productivity with an acceptable level of confidence, which, unlike the traditional method, includes not only the variables of mined tonnage, loading time and hang time; It also considers other quantitative and qualitative variables such as queue time, spot time, loaded travel time, queue time at the destination, spot time at the destination, dumping time, empty travel time, guard, shift, month, fleet of shovel, shovel, empty efh, loaded efh, material, material type, origin phase, destination phase, hour, truck and shovel availability.

The main objective is to improve the prediction of shovel productivity by applying Machine Learning, as well as to determine the best shovel productivity prediction model and determine the degree of importance of each of the independent variables for our predictor variable, through several libraries and with the help of Jupyter Notebook, ensuring that our results are the most reliable.

The research will begin with the general hypothesis that the productivity model developed with Machine Learning's Random Forest Regressor improves the prediction of blade productivity than other models.

For the machine learning model, all the variables present in the DataSet were used, including the qualitative variables, which were categorized before the training.

Of the machine learning models, the model that best predicts the productivity of the shovels is the "Random Forest Regressor" with an accuracy of 99.7% for training and 91.1% for the DataSet for testing; while the worst machine learning model is "Dummy Regressor" with -0.03% accuracy. In addition, the variable of greater degree of importance is the Shovel Fleet with 24.92%.

Keywords — Machine learning, random Forest, productivity, training.

Tabla de Contenido

Pág.

Lista de Tablas

Pág.

Lista de Figuras

Pág.

Introducción

La minería y la tecnología han sido complementarios desde el inicio de la operación, empezando desde la inclusión de equipos para el carguío y acarreo y mostrándose actualmente en las áreas de “Dispatch Mina” o en los “Centros integrados de operaciones”, en donde se almacenan miles de datos de las diferentes áreas involucradas en el proceso desde geología, pasando por mina como planta y terminando en los puertos con la comercialización del concentrado.

Actualmente el aprovechamiento de los datos registrados en los servidores de las empresas de los diferentes rubros, están permitiendo descubrir nuevas formas de negocio y mejoras de los logros ya alcanzados. En las grandes empresas mineras alrededor del mundo, la inteligencia artificial viene ganando cada año más importancia involucrándose desde la automatización de procesos operativos e informáticos, hasta creando modelos que permiten predecir y simular eventos para estimar resultados futuros y atacar las causas que podrían comprometer el desarrollo.

La implementación de modelos de predicción como de clasificación para diferentes áreas de la operación, permite descubrir el valor del análisis de la data, para la toma de decisiones instantáneas de manera rápida y efectiva. Actualmente existen librerías de Python que nos permite evaluar los mejores modelos de machine learning, siendo Random Forest Regressor el modelo más confiable para nuestra base de datos a modelar, el cual divide los datos en dos grupos siendo el principal el grupo de entrenamiento, el cual se somete a varias pruebas en interacciones a través de la creación de un bosque de árboles de decisión, el cual con ayuda del método de error cuadrático minimiza el error de estimación y asegura que la predicción alcance un grado de confianza aceptable, para luego realizar pruebas de predicción con el segundo grupo llamado DataSet de Testeo (grupo complementario al grupo de entrenamiento) y confirmar la confiabilidad.

En Unidad Mina Open Pit FIGMM, actualmente no se cuenta con un modelo de predicción de las productividades de las palas y se trabaja bajo el modelo clásico de cálculo, el cual solo contiene a las variables de tonelaje movido, tiempo de carguío y tiempo hang, tampoco se identifica cual es la variable más importante en el cálculo.

En el capítulo I de Planteamiento del Problema, trataremos generalidades de la unidad de estudio, geografía y ubicación. Posteriormente se realiza la Descripción del Problema de Investigación, los objetivos, las hipótesis, las variables involucradas e indicadores utilizados, los antecedentes de la investigación y la metodología de la investigación utilizada.

En el capítulo II de Marco Teórico y Conceptual, se presenta en Marco Teórico la Geología regional, local y la mineralización de la zona, también se muestra la flota de la Unidad minera y los parámetros de operación para el sistema de autonomía; mientras que en el Marco Conceptual se explica el modelo clásico de productividad de palas, conceptos generales de Dispatch, Machine Learnig y Random Forest Regressor.

El capítulo III de Levantamiento de Información se realizó la consulta Query para la extracción de los datos del servidor, se implementó la conexión y visualización de los datos en un dataframe en Jupyter notebook.

El capítulo IV de Modelo predictivo y Análisis de resultados, se empezó con la identificación de las variables, la detección de valores nulos, para continuar con el análisis estadístico descriptivo de las variables, detectando outliers, para finalmente preparar el DataSet a entrenar en el modelo de Random Forest Regressor, utilizando los mejores hiperparámetros y corroborrar la prueba de hipótesis.

La tesis finaliza con las conclusiones, recomendaciones, bibliografía y anexos.

Capítulo I. Parte introductoria del trabajo

1.1 Generalidades

1.1.1 Unidad de estudio

Open Pit FIGMM es una de las mayores inversiones en minería realizadas en el país, considerada como la mina digital y modelo del sector, produce concentrados de Cu y Mo. Actualmente está en etapa de operación comercial, teniendo como objetivo alcanzar las 300 mil toneladas de finos de Cu por año, aportando un crecimiento del 15% de la producción nacional.

Tiene un estimado de 1.7 mil millones de toneladas de reservas minerales, 8.9 millones de toneladas de cobre contenido a 0.53% TCu, y una vida de reserva de 36 años, con potencial de ampliarse más, dados sus recursos minerales adicionales, que representan 1.6 mil millones de toneladas, que contienen 6.1 millones de toneladas de cobre (a 0.38% TCu).

Su operación es a Tajo abierto, utilizando camiones autónomos CAT 794AC CMD, palas de producción CAT 7495 y CAT 6060BH. Utiliza el sistema Minestar de Caterpillar como software de administración de flotas. Actualmente cuenta con 2 botaderos, 1 chancadora y 2 Stops de mineral.

La primera producción de concentrado de Cu se logró en Julio del 2022 pasando por un periodo de prueba hasta setiembre del mismo año, concretándose como un hito empresarial para la región y el país, apoyándose en la tecnología y reafirmando su compromiso con la sociedad y el medio ambiente.

1.1.2 Ubicación y acceso

Unidad Minera: Open Pit FIGMM

Región: Moquegua

La mina se ubica en el departamento de Moquegua. Se encuentra a una altura de 3400 m.s.n.m. a 103.3 Kms. de Tacna. Sus coordenadas geográficas son:

Tabla 1

Coordenadas geográficas del Proyecto Minero Open Pit FIGMM.

Vértice	Norte	Este
1	8,108,312.57	327,466.78
2	8,108,405.32	326,471.69
3	8,108,803.36	326,508.79
4	8,108,710.61	327,503.89

Nota: fuente Instituto Geológico, Minero yMetalúrgico.

1.2 Descripcion del problema de investigacion

Vivimos en una sociedad en donde la tecnología nos permite tener las herramientas de análisis y mejora continua. Actualmente las herramientas de la Ciencia de los Datos es un tema que genera gran acogida a nivel mundial, ya que se convierte en una puerta de posibilidades de mejora del negocio, a través del tratamiento de una data histórica y del conocimiento especializado en el negocio en el cual se piensa implementar

El machine Learning es uno de los métodos analíticos de la ciencia de Datos. El cual por sí mismo sin intervención humana y en forma automatizada nos permite identificar los factores determinantes, la tendencia y la relación de nuestros datos, generando que nuestra data se enriquezca y logremos obtener las mejoras del negocio que se desean.

Actualmente las empresas buscan una manera de optimizar recursos, generar valor en el negocio a través de una mayor productividad de sus equipos de operación. En el caso del sector minero no es una excepción ya que cada una de sus actividades son críticas y el impacto que se pueda generar en alguna de ellas, afectará de manera significativa a la empresa.

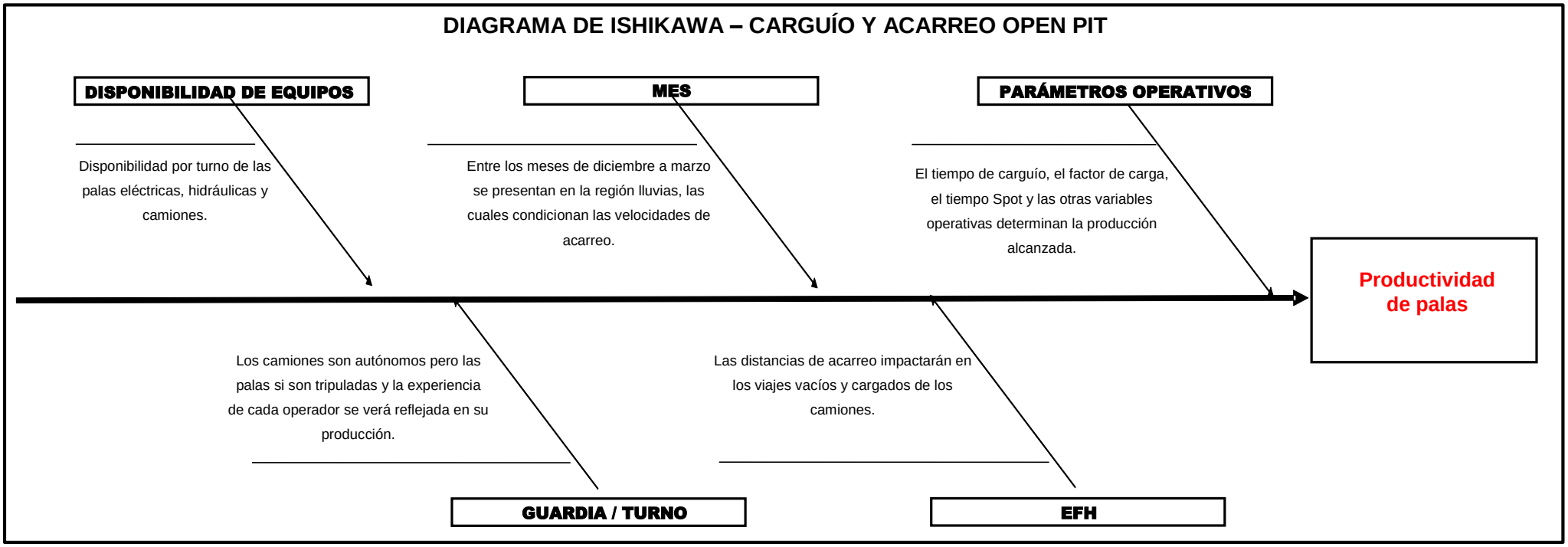
En minería, una de las actividades de mayor impacto económico es el carguío y acarreo de mineral y desmonte, en donde se determina la producción y se utilizan el mayor número de equipos de operación. Obteniéndose de ellos una data tan enriquecedora, la cual nos permite generar una serie de reportes a los interesados de las operaciones y a las gerencias de dichas empresas. Sin embargo, esta data es aún un mundo complejo de información, en la cual se podrían identificar el valor de importancia de cada uno de las variables que actúan en la operación, y un modelo que nos permita predecir las productividades de sus equipos, a través del uso de los modelos de machine learning.

En la Mina Open Pit FIGMM, se presentan problemas de alto tiempo hang, para poder resolver este problema, inicialmente analizaremos cada una de las variables que intervienen en el carguío y acarreo. Para ello realizamos un Diagrama de ISHIKAWA, en la cual se pudo ver a detalle las variables que influyen más en el aumento de la producción y nos lleva a realizarnos las siguientes preguntas.

¿Se podrá determinar un modelo matemático de productividad de palas con las variables de operación registradas en el sistema Dispatch usando Random Forest Regressor? ¿Será el modelo de Random Forest Regressor el de mayor grado de predicción que el de otros modelos de machine learning para determinar la productividad de palas? ¿Cuáles son las variables más importantes e influyentes en la productividad de palas?

Figura 1

Diagrama de ISHIKAWA del Proceso de Carguío y Acarreo en una operación Open Pit.



Nota: fuente elaboración propia.

Del diagrama de ISHIKAWA del Proceso de Carguío y acarreo; Figura 2; se concluye que se pueden trabajar en cinco aspectos.

Disponibilidad de equipos: La disponibilidad de la flota de carguío y acarreo entregada por el área de mantenimiento es uno de los principales condicionadores de la producción que se puede alcanzar por cada pala.

Guardia / turno: El factor humano (habilidad de los operadores) y las condiciones de trabajo (día o noche) afecta significativamente la producción, reduciendo la visualización de vías y aumentando las medidas de seguridad para evitar accidentes.

Mes: En la unidad minera Open Pit FIGMM entre los meses de diciembre a enero se presentan épocas de lluvia moderada a intensa, impactando en las condiciones de seguridad de las vías, reduciendo las velocidades alcanzadas por los camiones y aumentando el tiempo de acarreo, así como la restricción de algunas zonas de la mina por estabilidad de taludes.

EFH: Al presentar una relación inversa con respecto a la producción de una pala, es importante determinar el grado de influencia de manetra que permita tomar decisiones in situ y obtener mejores resultados.

Parámetros operativos: Entre los parámetros importantes tenemos el tiempo de Carguío, tiempo Spot, queue (%), hang (%), tiempo promedio de viaje vacío, tiempo promedio de viaje cargado y tiempo de descarga los cuales se reflejan en el uso de los equipos y en la producción que se pueda alcanzar.

1.3 Objetivos

1.3.1 Objetivo general

Mejorar la predicción de la productividad de palas aplicando Machine Learning.

1.3.2 Objetivos especificos

- Determinar el mejor modelo de predicción de la productividad de palas aplicando Machine Learning a través de algoritmos de Python.

- Determinar el grado de importancia de cada una de las variables independientes para nuestra variable predictora.

1.4 Hipotesis y variables

1.4.1 Hipótesis general

El modelo de productividad desarrollado con Random Forest Regressor de Machine Learning mejora la predicción de la productividad de palas que otros modelos.

1.4.2 Hipótesis especifica

- A mayor factor de carga, se obtendrá una mayor productividad de palas.
- A menor tiempo de carguío, tiempo queue, tiempo spot, tiempo cargado, tiempo descarga, tiempo vacío y tiempo hang se obtendrá una mayor productividad de las palas.
- Las variables factor de carga, tiempo de carguío, tiempo queue, tiempo spot, tiempo cargado, tiempo descarga, tiempo vacío y tiempo hang son más importantes e influyentes que las variables periodo, turno, guardia, flota SH, flota HT, fase de origen, tipo de material y fase de destino en la productividad de las palas.

1.4.3 Variables

Variable independiente: factor de carga, tiempo de carguío, tiempo queue, tiempo spot, tiempo cargado, tiempo descarga, tiempo vacío, tiempo hang, periodo, turno, guardia, flota SH, flota HT, fase de origen, tipo de material y fase de destino.

Variable dependiente: productividad de palas.

Tabla 2

Matriz de consistencia.

Problema General	Objetivo General	Hipótesis General	Variables	Indicadores
¿Se podrá determinar un modelo matemático de productividad de palas con las variables de operación registradas en el sistema Dispatch usando Random Forest Regressor?	Mejorar la predicción de la productividad de palas aplicando Machine Learning.	El modelo de productividad desarrollado con Random Forest Regressor de Machine Learning mejora la predicción de la productividad de palas que otros modelos.	Dependientes: Y1: Productividad de palas Independientes:	Productividad (Tonelaje/hor
Problema Especifico	**Objetivo Especifico**	**Hipótesis Especifica**		**Indicadores**
P1: ¿Será el modelo de Random Forest Regressor el de mayor grado de predicción que el de otros modelos de machine learning para determinar la productividad de palas? P2: ¿Cuáles son las variables más importantes e influyentes en la productividad de palas?	O1: Determinar el mejor modelo de predicción de la productividad de palas aplicando Machine Learning a través de algoritmos de Python. O2: Determinar el grado de importancia de cada una de las variables independientes para nuestra variable predictora.	H1: A mayor factor de carga, se obtendrá una mayor productividad de palas. H2: A menor tiempo de carguío, tiempo queue, tiempo spot, tiempo cargado, tiempo descarga, tiempo vacío y tiempo hang se obtendrá una mayor productividad de las palas. H3: Las variables factor de carga, tiempo de carguío, tiempo queue, tiempo spot, tiempo cargado, tiempo descarga, tiempo vacío y tiempo hang son más importantes e influyentes que las variables periodo, turno, guardia, flota SH, flota HT, fase de origen, tipo de material y fase de destino en la productividad de las palas.	X1: factor de carga X2: tiempo de carguío X3: queue (%) X4: tiempo spot X5: tiempo cargado X6: tiempo descarga X7: tiempo vacío X8: hang (%) X9: mes X10: turno X11: guardia X12: flota SH X13: flota HT X14: origen X15: material X16: destino X17: Disponibilidad X18: EFH	Indicadores de: X1: tonelaje cargado en cada viaje (Ton) X2, X4, X5, X6, X7: Tiempos en minutos. X3, X8, X17: Porcentajes X9: mes del año X10: día y no X11: A, B, C, D X12: CAT 7495 y CAT 6060BH X13: CAT 794AC CMD X14, X16: fase 1N, fase 1S, fase 2N, fase 2S y fase3N. X15: Mineral, desmonte volcánico, no volcánico y fill. X18: metros.

Nota: fuente elaboración propia.

1.5 Antecedentes investigativos

En el año 2020, Alí Soofastaei en su libro titulado “Data Analytics Applied to the Mining Industry”, describe los desafíos claves a los que se enfrenta el sector minero a medida que se transforma en una industria digital y proporciona las pautas sobre cómo se deben implementar el análisis de datos, así como los enfoques y métodos para mejorar la toma de decisiones.

En el año 2020, Chong Chong Qi en su artículo científico “Big Data Management in the Mining Industry” publicado en International Journal of Minerals, Metallurgy and Materials, concluye que la industria Minera se enfrenta a grandes desafíos como la disminución de los precios de los metales y la ley del mineral; siendo el mundo de las oportunidades muy pequeño y el Big Data una de las soluciones tecnológicas más prometedoras.

En el año 2017, Samuel Sebastián Cornejo Castro, en su trabajo de tesis titulado “Optimización - simulación de carguío y acarreo en tajo Abierto utilizando NSGAII y programación lineal entera” menciona que “Su modelo logró resolver el problema de creación de cronogramas de producción óptimos maximizando sus beneficios en parámetros excluyentes entre sí mismos (blending, tiempo de ejecución y balanceo de trabajo), donde el tiempo de cálculo fue razonable a través de la programación heurística y la lineal entera”.

En el año 2015, Gerardo William Mauricio Quiquia en su trabajo “Mejoramiento continuo en la gestión del ciclo de acarreo de camiones en minería a tajo abierto en Antamina, Cerro Verde, Toquepala, Cuajone, Yanacocha, Alto Chicama, Las Bambas, Cerro Corona, Antapacay y Pucamarca”, concluye que en el sector minero se cuenta con una gran base de información generada por el sistema de Gestión de Flota, la cual nos permitirá tomar mejores decisiones en las operaciones y lograr la optimización de los procesos.

En el año 2012, Matías Gil en su artículo “La Industria Minera en búsqueda de la eficiencia con Big Data” publicado en la revista América Economía muestra la necesidad de aplicar Modelos de Análisis de Datos con Big Data generando valor potencial al usar la totalidad de los datos que se generan en las operaciones mineras.

1.6 Metodologia

Tipo y Diseño

El enfoque de la presente investigación es Aplicada, porque se utiliza un Modelos de Machine Learning, el cual fue desarrollado previamente en otras investigaciones, pero se contextualizará al contexto del sector minero.

El Alcance es correlacional, ya que evaluaremos la producción alcanzada respecto al comportamiento de las variables de carguío y acarreo.

El presente trabajo está basado en el diseño de investigación no experimental, ya que se basará en registros de datos obtenidos del sistema Dispatch Mina, los cuales serán entrenados a través de modelos de Machine Learning para determinar su correlación

Etapas

Recolección de Datos

Los datos serán extraídos de los servidores del sistema Dispatch Mina, a través del uso de consultas con Querys en SQL Server, por medio de la conexión a la VPN de la empresa.

Procesamiento de la información

El procesamiento de la información se realizará siguiendo los siguientes pasos:

- Identificación de Variables
- Detección de Nulos
- Análisis Estadístico descriptivo de las variables
- Detección de Datos Outliers y limpieza de Datos
- Preparación de la Data a entrenar
- Selección de Modelo de Machine Learning

- Modelamiento de Datos
- Análisis de la información

Se analizarán los resultados del modelo, se identificarán mejoras en los procesos identificando las variables más influyentes para lograr el objetivo.

Capitulo II. Marcos teórico y conceptual

2.1 Marco teorico

2.1.1 Geología

Está constituido por rocas volcánicas, sedimentarias e intrusivas de edades desde el Mesozoico al Cuaternario. El proyecto Open Pit FIGMM se ubica en una zona con sismicidad alta.

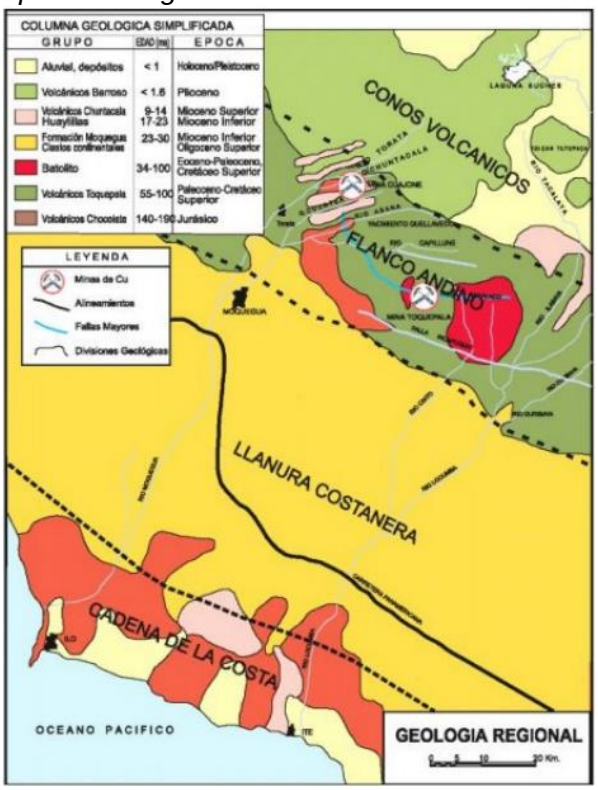
Geologia regional: está localizado en la zona Sur-Este de la Faja de pórfidos de cobre del Perú rodeado de un ambiente geológico de rocas volcánicas con riolitas en su parte basal y una intercalación de flujos andesíticos, brechas andesíticas y riolitas en la parte superior.

Las rocas más antiguas pertenecen al cretáceo superior y terciario inferior aflorando en la parte Oeste intruídas por un Plutón granodiorítico-tonalítico y este por un stock porfirítico dacítico. Luego de un largo período de erosión se depositaron los tufos e ignimbritas riolíticas de la formación Huaylillas.

Posteriormente se depositaron, sobre la superficie discordante; una secuencia intercalada de flujos volcánicos mayormente andesíticos, material piroclástico y clástico de la formación Barroso.

Figura 2

Columna geológica simplificada regional.



Nota: fuente Instituto Geológico Minero y Metalúrgico.

Geologia local: la unidad minera presenta las siguientes formaciones.

El pórfido riolítico se encuentra como xenolitos de diversos tamaños en las rocas ígneas intrusivas. Tiene textura porfirítica y posee estructura fluidal con fenocristales de ortosa y cuarzo, algunos de plagioclasa y biotita. Matriz con predominancia de ortosa; cuarzo y biotita. Pertenece al Cretáceo superior - Terciario inferior.

La granadiorita es la roca más predominante, intruye al pórfido riolítico en el área del yacimiento y a la secuencia volcánica superior al Sur y SW de área del depósito.

El pórfido monzonítico dacítico está asociado a los procesos de alteración de mineralización hipógena del depósito. Abarca la parte central del yacimiento y tiene forma oval (con dimensiones de 250 metros por 1.4 KM. respectivamente), con su eje mayor orientado al NW. Intruye al Plutón granodiorítico-tonalitico en forma normal, por diques y numerosas digitaciones. Su textura es porfirítica con abundancia de fenocristales de plagioclasa, cuarzo; algunos de ortosa y biotita. La ortosa muy raramente se presenta como

fenocristales. Siendo su matriz un agregado microgranular, de cuarzo, ortosa con algunos cristales de biotita subhedrales pseudo hexagonales. Los minerales accesorios son el zircón, magnetita y esfena. La relación plagioclasa ortosa varía en el stock, siendo la composición dacítica la que predomina sobre la monzonítica, presente principalmente en la parte profunda y central del yacimiento. Se observan numerosos xenolitos, así como la granodiorita constituyendo la principal roca albergante de la mineralización.

Presenta cuerpos intrusivos menores, principalmente diques, intruyen al stock pórfido monzonítico-dacítico y a la granodiorita-tonalita. Son rocas de textura porfirítica con fenocristales anhedrales a subhedrales de plagioclasa, ortosa, cuarzo y biotita con matriz mayormente microgranular, la proporción plagioclasa, ortosa, cuarzo varía, dando lugar a que la composición de estos cuerpos oscile de pórfido monzonítico cuarcífero a pórfido diorítico cuarcífero. Están asociados a una menor actividad hidrotermal ocurrida después de su emplazamiento y que nos indica la amplitud del proceso hidrotermal. Se observa una preferente ubicación espacial, de estos cuerpos interminerales, en el stock pórfido dacítico, los valores de cobre son inferiores a 0.4%.

En la parte norte y sur de la quebrada del río Asana, afloran los tufos e ignimbritas riolíticas de la formación Huaylillas, presentes en una superficie discordante de erosión sobre el conglomerado, rocas volcánicas e intrusivas, anteriores a la actividad hidrotermal hipógena. Su espesor aproximado es de 200 metros. Por su posición estratigráfica infrayacente a la formación Barroso, del Plioceno medio a superior, se le considera como de edad terciario superior, esta roca no posee gran fracturamiento como las rocas premineralización y mineralización.

Mineralización: en la zona lixiviada, limonita de 5 a 60 m verticales; la zona de sulfuros secundarios, encierra más del 50% de la reserva mineral. La distancia vertical varía de unos pocos metros a 102 m con leyes hasta 1.5% Cu. Tiene calcosina, como accesorios covelita, digenita y bornita. Los sulfuros secundarios se desarrollaron entre el Eoceno y el Plioceno, la erosión fue más lenta que la lixiviación; posterior al Pleistoceno,

la erosión fue mayor que la lixiviación; sin formación de sulfuros secundarios en ciertas áreas.

En la Zona de sulfuros primarios se tiene Calcopirita y pirita como minerales principales, molibdenita como accesorio; como trazas presenta esfalerita, galena, cubanita, mackinawita, pirrotina, bornita, marcasita y tetraedrita; oro y plata.

La mineralización diseminada predomina sobre la mineralización de relleno de venillas en el stockwork. La zona más rica de calcopirita está en el sector central y al noroeste del área mineralizada. La mineralización más intensa corresponde a las zonas más fracturadas.

2.1.2 Mina

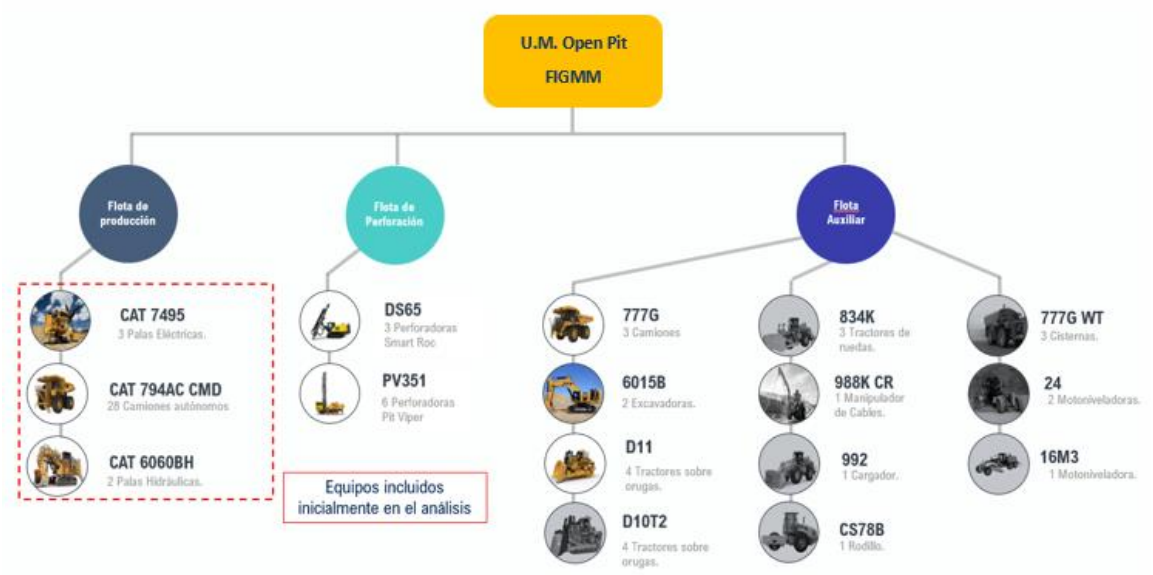
En UM. Open Pit FIGMM el método de explotación es a tajo abierto y se cuenta con:

3 flotas de producción (3 palas eléctricas CAT 7495, 2 palas hidráulicas CAT 6060BH y 28 camiones autónomos CAT 794AC CMD), 9 perforadoras (3 Smart Roc DS65 y 6 Pit Viper PV351), 11 flotas de equipos auxiliares (siendo los principales 3 flotas de camiones CAT 777G, 2 de excavadoras CAT 6015B y 4 de tractores sobre orugas CAT D11).

En la Figura 3, los equipos que están enmarcados en el rectángulo con líneas punteadas rojas son los que estarán incluidos en el análisis inicial.

Figura 3

Flota de operaciones de la UM. Open Pit FIGMM



Nota: fuente elaboración propia.

Consideraciones en la operación:

- La chancadora se ubica en la parte inferior del tajo.
- En la limpieza del material volado se dejan las crestas que serán limpiadas por la excavadora, todo esto por seguridad debido a la probabilidad de caída de material a la parte inferior del tajo donde se encuentra ubicada la chancadora.
- En el trabajo de preparación de rampas, inicialmente se acomoda la carga con los tractores de oruga, la finalidad es de generar la plataforma para que pueda entrar la pala y empezar a cargar a los camiones.
- Los camiones son autónomos y pueden detenerse por las siguientes razones:

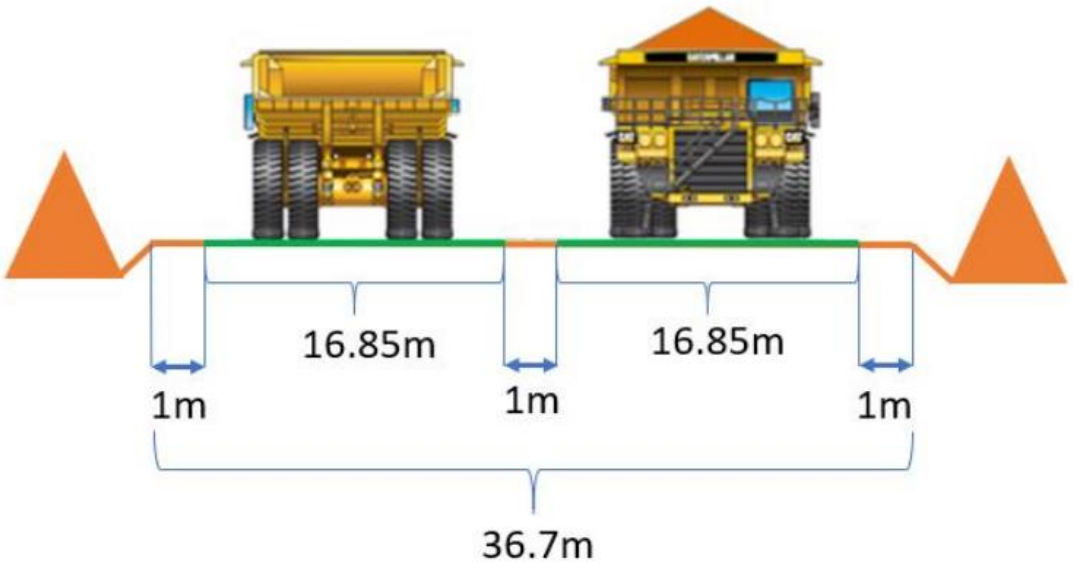
- Pérdida de señal de GPS del camión o de otro equipo que está transitando en la misma ruta, dentro de un radio de 300 m.
- Por desperfecto mecánico (evento health)
- Por falta de asignación (no tiene destino al cual ir)
- Evento de proximidad, cuando el equipo que transita en la misma ruta que el camión realiza maniobras erróneas que pueden hacer que se choquen.

- Los objetos que se pueden detectar pueden incluir objetos físicos, así como polvo, vapor o materiales que la luz no puede penetrar.
- Los camiones de la flota CAT 794 AC cuentan con un sensor que permite identificar elementos en la vía a una distancia no mayor de 100m.
- Los climas adversos afectan negativamente las velocidades de los camiones, por ejemplo, cuando hay época de luvias las condiciones de las carreteras son afectadas, el camión identifica estas condiciones y disminuye su velocidad. Cuando hay neblinas los camiones tambien son afectados ya que estos cuenta con un sensor que puede distinguir que es lo que hay al frente de estos, por lo tanto, como la distancia de detección es reducida los camiones disminuyen su velocidad.
- Las palas eléctricas CAT 7495 cuentan con un sensor que permite realizar el levantamiento topográfico de todo el frente de minado mediante un giro de 180°, también cuenta con un sistema que permite ubicar dos puntos de carguío en un minado doble y un punto de carguío en un minado simple, estos puntos sirven de target para que los camiones CAT 794 AC se posicionen antes de ser cargados.
- Durante el carguío de un camión, si el camión se mueve más de 160 m (525 ft), el sistema de carga útil del camión detectará que se ha completado el ciclo de carga. Además, no se registrarán las pasadas adicionales del cargador.
- El sistema de carga útil del camión volverá a pesar la carga útil cuando la máquina alcance los 8 km/h.
- El builder partiendo de las rutas principales se encarga de realizar rutas cercanas al frente de minado, es aquí donde la pala coloca sus puntos de minado, el sistema minestar identifica estos puntos y dibuja las rutas de aculatamiento más óptimas.

- La función del controller es la de asignar camiones hacia diferentes orígenes y destinos, controla el estado de los equipos, coordina con los supervisores de campo para controlar la seguridad y aumentar la eficiencia.
- Las palas eléctricas (CAT 7495) para realizar un carguío de lado doble necesitan frentes de minado con un ancho mayor a 90m y para realizar un carguío de lado simple necesita frentes de minado con un ancho mayor a 50m y menor a 90m.
- El mínimo ancho de minado para las palas eléctricas (CAT 7495) es de 50 m, menor a esta distancia las palas hidráulicas (CAT 6060BH) entran a minar, estos equipos tienen un ancho mínimo de 30 m, posteriormente para frentes con anchos menores a estos ya entra a minar las excavadoras (CAT 6015).

Figura 4

Diseño de una rampa en Open Pit



Nota: fuente Manual de Caterpillar

- El mínimo ancho de un carril para lograr un máximo perfomance, por parte del camión autónomo es de 1.75 veces el ancho del camión, siendo el ancho de este 9.63 m, para una vía de acarreo el ancho total deberá tener 36.7 m (Figura 4).

- El camión CAT 794AC CMD puede lograr tener una velocidad máxima cargado en bajada de 31 km/h y en vacío en subida 42 km/h.
- Se aplica la política de carga útil de Caterpillar conocida como “Política de Sobrecarga 10/10/20”, la cual nos indica que “No más del 10% de las cargas útiles pueden exceder el 110% de la carga útil objetivo del camión y ninguna carga útil debe exceder nunca el 120% de la máxima carga útil objetivo”.
- El camión no debe ser cargados en pendientes mayores a los 5%, ya que los datos registrados como el caso de los pesajes podrían no ser los correctos.

2.2 Marco conceptual

2.2.1 Modelo de producción estándar

Existen muchas definiciones de productividad, una de ellas es la de Joseph Prokopenko que en 1989 la define como la relación entre los resultados y el tiempo que lleva conseguirlos; es decir:

Productividad = Productos / Insumos

Bajo este concepto en la industria minera, se puede definir dos tipos de productividades para las palas:

2.2.1.1 Productividad efectiva. Es la relación entre las toneladas nominales cargadas y el tiempo efectivo de carga, incluyendo el tiempo de cuadrado. Esto es lo que se producirá si el hang fuera cero.

$$\text{Productividad efectiva (Ton/h)} = \frac{\text{Toneladas nominales}}{\text{T. carguío + T. cuadrado}}$$

2.2.1.2 Productividad horaria. Es la relación entre las toneladas nominales y el tiempo total productivo, que incluye el tiempo de carguío, el tiempo de de cuadrado y el tiempo hang.

$$\text{Productividad horaria (Ton/h)} = \frac{\text{Toneladas nominales}}{\text{T. carguío + T. cuadrado + T. Hang}}$$

2.2.2 Sistema administración flota mina

Los sistemas de gestión de flota mina son herramientas tecnológicas que permiten optimizar los procesos de operación y reducir los costos que en ella se generan. La gran mayoría de las empresas a tajo abierto cuentan con su sistema Dispatch que es un sistema de administración minera, que emplea tecnología moderna en comunicaciones, GPS (Sistema de Posicionamiento Global) y sistemas computacionales que manejan toda la información histórica y en tiempo real de la operación minera, proporcionando asignaciones óptimas y automáticas para camiones de acarreo. Permitiendo incrementar el tiempo efectivo de trabajo de palas y camiones y por ende su productividad. (Mauricio, 2014). Utiliza tres modelos matemáticos de programación.

La mejor Ruta: este algoritmo calcula el tiempo mínimo de un nodo a otro (punto virtual de ubicación), mediante una red de nodos que describen un árbol direccionado. Una vez realizado el cálculo de la mejor ruta, se entrega al siguiente subsistema (PL) la siguiente información acerca de las rutas de acarreo. (Mauricio, 2014)

Programación lineal: utiliza las soluciones entregadas por PL para generar asignaciones óptimas de equipos en tiempo real, utilizando el método 17 Simplex, este es un método matemático, que ayuda en la optimización, y así minimizar las necesidades de los camiones, siguiendo el concepto prioridades y exigencias. (Mauricio, 2014).

Programación dinámica: Es un proceso de optimización basado en el principio optimizante de Bellman's. Se tiene en consideración la disponibilidad de equipos, flujos de alimentación (puntos de carga y descarga), prioridad de palas, distancias de acarreo. Para generar la solución, Dispatch en lugar de asignar camiones a las palas que más lo requieren decide por asignar camiones a los equipos de carguío más necesitados en cualquier momento ya sea que requieran asignación o ya lo estén. (Mauricio, 2014). Para este proceso, el sistema genera dos listas, una en base a la PL, donde incluye rutas ordenadas por prioridad de tiempo y una lista de camiones que requieran asignación a través del tiempo. Es así como la Programación Dinámica (PD) establece las necesidades

de camiones óptimos sobre la base de los que requerirán asignación de carguío o bien puedan variar. (Mauricio, 2014).

La importancia del sistema Dispatch no es sólo en la parte operacional, sino también en la alimentación constante de grandes volúmenes de datos que quedan registrados en los servidores de las empresas; sin embargo, aún es un mundo por aprovechar, ya que en su mayoría son sólo usados para la presentación de reportes de KPIs a las distintas áreas de las operaciones de manera descriptiva dejando de lado la parte analítica.

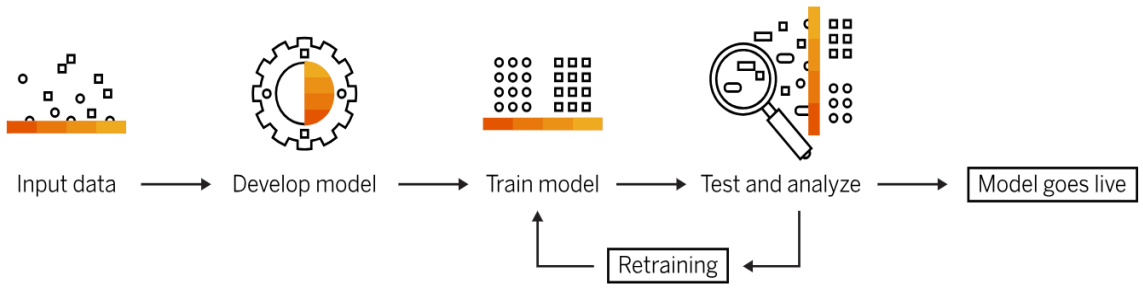
2.2.3 Machine learning

Es una rama de la inteligencia artificial (IA) y la informática que se centra en el uso de datos y algoritmos para imitar la forma en la que aprenden los seres humanos, con una mejora gradual de su precisión.

Mediante el uso de métodos estadísticos, los algoritmos se entrenan para hacer clasificaciones o predicciones, y descubrir información clave dentro de los proyectos de minería de datos. Esta información clave facilita la toma de decisiones dentro de las aplicaciones y las empresas, lo que afecta idealmente a las métricas de crecimiento.

Figura 5

Funcionamiento del proceso de machine learning.



Nota: fuente https://www.sap.com/latinamerica/products/artificial-intelligence/what-is-machine-learning.html

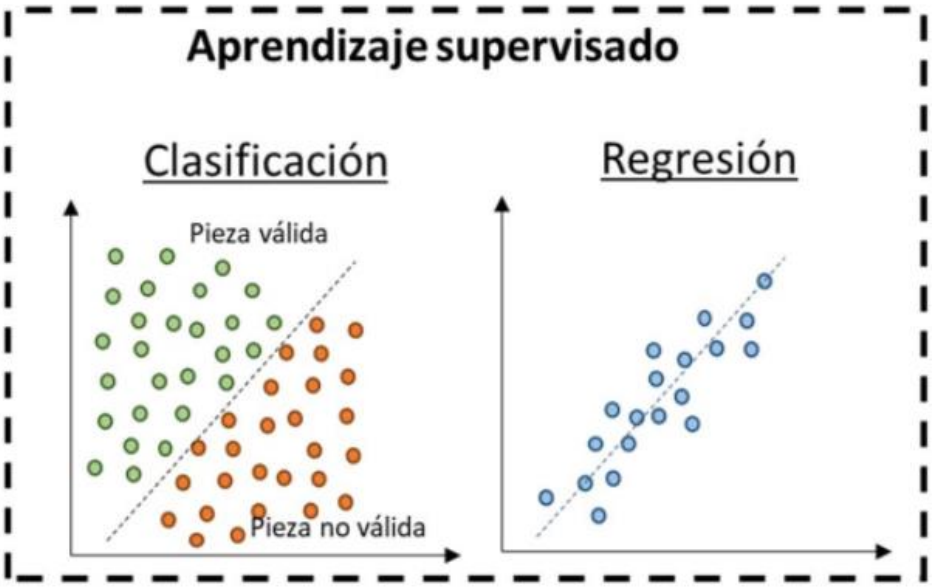
2.2.4 Métodos de machine learning

2.2.4.1 Supervisado. Usa conjuntos de datos etiquetados entrenando algoritmos para clasificar datos o predecir resultados con precisión. A medida que se introducen datos

de entrada en el modelo, este adapta sus pesos hasta que se haya ajustado correctamente. Esto ocurre como parte del proceso de validación cruzada para asegurarse de que el modelo evite el sobreajuste o el subajuste. El aprendizaje supervisado permite a las organizaciones resolver una amplia variedad de problemas del mundo real a escala. Algunos métodos utilizados en el aprendizaje supervisado son las redes neuronales, Naïve Bayes, la regresión lineal, la regresión logística, el bosque aleatorio y la máquina de vectores de soporte (SVM).

Figura 6

Machine learning supervisado



Nota: fuente *https://www.interempresas.net/MetalMecanica/Articulos/385970-Monitorizacion-de-defectos-y-machine-learning-en-la-tecnologia-aditiva-L-PBF.html*

2.2.4.2 No supervisado. Utiliza algoritmos de machine learning para analizar y agrupar en clústeres conjuntos de datos sin etiquetar. Estos algoritmos descubren agrupaciones de datos o patrones ocultos sin necesidad de ninguna intervención humana. La capacidad de este método para descubrir similitudes y diferencias en la información lo convierten en ideal para el análisis de datos exploratorios, las estrategias de venta cruzada, la segmentación de clientes y el reconocimiento de imágenes y patrones. También se utiliza para reducir el número de características de un modelo mediante el proceso de reducción de dimensionalidad. El análisis de componentes principales (PCA) y la

descomposición en valores singulares (SVD) son dos de los enfoques más habituales para realizar este proceso. Otros algoritmos utilizados en el aprendizaje no supervisado son las redes neuronales, la agrupación en clúster de medias K y los métodos de agrupación probabilística.

Figura 7

Machine learning no supervisado

Nota: fuente *https://www.interempresas.net/MetalMecanica/Articulos/385970-Monitorizacion-de-defectos-y-machine-learning-en-la-tecnologia-aditiva-L-PBF.html*

2.2.4.3 Semisupervisado. Se convierte en una solución viable cuando hay grandes cantidades de datos crudos y no estructurados. Este modelo consiste en introducir pequeñas cantidades de datos etiquetados para aumentar los data sets sin etiquetar. Esencialmente, los datos etiquetados actúan para dar un inicio de funcionamiento al sistema y pueden mejorar considerablemente la velocidad y precisión del aprendizaje. Un algoritmo de aprendizaje semisupervisado instruye a la máquina para que analice los datos etiquetados según propiedades correlativas que podrían aplicarse a los datos no etiquetados.

2.2.4.4 Por refuerzo. La máquina recibe la respuesta de referencia y aprende encontrando correlaciones entre todos los resultados correctos. El modelo de aprendizaje por refuerzo no incluye una respuesta de referencia, sino que más bien introduce un conjunto de acciones permitidas, reglas y estados finales potenciales. Cuando el objetivo deseado del algoritmo es fijo o binario, las máquinas pueden aprender mediante el ejemplo.

Pero en los casos en los que el resultado deseado es mutable, el sistema debe aprender por experiencia y recompensa. En los modelos de aprendizaje por refuerzo, la "recompensa" es numérica y se programa dentro del algoritmo como algo que el sistema busca recopilar.

2.2.5 Principales algoritmos del machine learning

Redes neuronales: simulan el funcionamiento del cerebro humano, con un gran número de nodos de proceso vinculados. Son eficaces para reconocer patrones y juegan un importante papel en aplicaciones como, por ejemplo, la conversión al lenguaje natural, el reconocimiento de imágenes, el reconocimiento del habla y la creación de imágenes.

Regresión lineal: se utiliza para predecir valores numéricos, con base en una relación lineal entre diferentes valores. Por ejemplo, la técnica podría servir para prever los precios de la vivienda en función de los datos históricos de la zona.

Regresión logística: este algoritmo de aprendizaje supervisado hace predicciones para variables de respuesta categórica, como respuestas "sí/no" a las preguntas. Se puede utilizar para aplicaciones como la clasificación de correo no deseado y el control de calidad de una línea de producción.

Agrupación en clústeres: mediante el aprendizaje no supervisado, los algoritmos de agrupación en clúster pueden identificar patrones en los datos para que puedan ser agrupados. Los ordenadores pueden servir a los científicos de datos para identificar las diferencias entre los elementos de datos que los humanos hayan pasado por alto.

Árboles de decisión: se pueden utilizar para predecir valores numéricos (regresión) y para clasificar datos en categorías. Los árboles de decisión utilizan una secuencia de ramificaciones de decisiones vinculadas que se pueden representar con un diagrama de árbol. Una de las ventajas de los árboles de decisión es que son fáciles de validar y auditar, a diferencia de la caja negra de la red neuronal.

Bosques aleatorios: predice un valor o categoría combinando los resultados de una serie de árboles de decisión.

2.2.6 Random Forest Regressor

Determina un subconjunto aleatorio de características, lo que garantiza una baja correlación entre los árboles de decisión. Ésta es una diferencia clave entre los árboles de decisión y los bosques aleatorios. Mientras que los árboles de decisión consideran todas las posibles divisiones de características, los bosques aleatorios solo seleccionan un subconjunto de esas características.

Tiene tres hiperparámetros principales, que deben configurarse antes del entrenamiento:

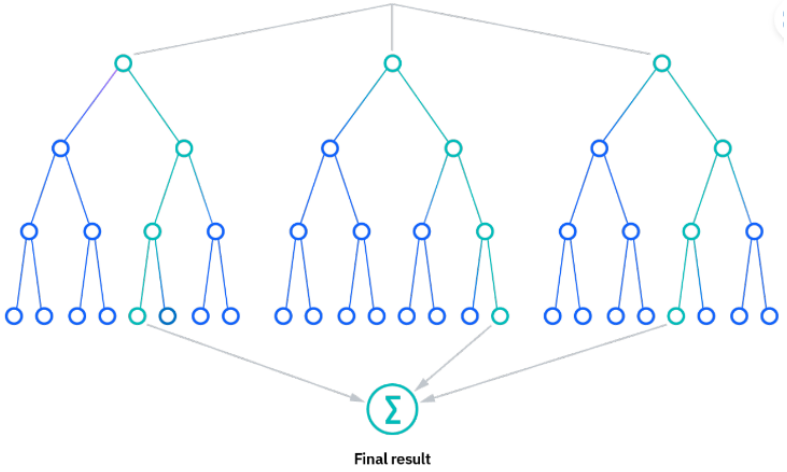
- Tamaño del nodo
- Cantidad de árboles
- Cantidad de características muestreadas

A partir de ahí, el clasificador de random forest se puede utilizar para solucionar problemas de regresión o clasificación. Se compone de un conjunto de árboles de decisión, y cada árbol del conjunto se compone de una muestra de datos extraída de un conjunto de entrenamiento con reemplazo, llamada muestra de arranque.

De esa muestra de entrenamiento, un tercio se reserva como datos de prueba, lo que se conoce como muestra fuera de la bolsa (oob), a la que volveremos más adelante. Luego, se inyecta otra instancia de aleatoriedad a través del agrupamiento de características, lo que agrega más diversidad al conjunto de datos y reduce la correlación entre los árboles de decisión. Dependiendo del tipo de problema, la determinación de la predicción variará. Para una tarea de regresión, se promediarán los árboles de decisión individuales, y para una tarea de clasificación, un voto mayoritario, es decir, la variable categórica más frecuente, arrojará la clase predicha. Finalmente, la muestra de oob se utiliza para la validación cruzada, finalizando esa predicción.

Figura 8

Funcionalidad del algoritmo de Random Forest Regressor



Nota: fuente *https://www.ibm.com/mx-es/topics/random-forest*

Capitulo III. Desarrollo del trabajo de investigación

3.1 Creación de la consulta Query

Para obtener los datos del servidor se generó la siguiente consulta Query en el programa SQL Server.

```
select * from (select (dateadd(hour,-5, [STARTTIME_UTC])) AS STARTTIME ,
(dateadd(hour,-5, [ENDTIME_UTC])) AS ENDTIME, ([PAYLOAD]/1000) as PAYLOAD,
(select M.NAME from [MSSUMM].[dbo].[CYCLE_DIM_MACHINE] M where M.OID = PRIMARYMACHINE) as PRIMARYMACHINENAME,
(select M.CLASSNAME from [MSSUMM].[dbo].[CYCLE_DIM_MACHINE] M where M.OID = PRIMARYMACHINE) as PRIMARYMACHINECLASSNAME ,
(select M.NAME from [MSSUMM].[dbo].[CYCLE_DIM_MACHINE] M where M.OID = SECONDARYMACHINE) as SECONDARYMACHINENAME,
(select M.CLASSNAME from [MSSUMM].[dbo].[CYCLE_DIM_MACHINE] M where M.OID = SECONDARYMACHINE) as SECONDARYMACHINECLASSNAME,
(select G.CREWID from [MSSUMM].[dbo].[CYCLE_DIM_CREW] G where G.OID = CREW) as GUARDIA,
(select MA.NAME from [MSSUMM].[dbo].[CYCLE_DIM_MATERIAL] MA where MA.OID = LOADERMATERIAL) as MATERIAL,
(select MA.GROUPLEVEL1 from [MSSUMM].[dbo].[CYCLE_DIM_MATERIAL] MA where MA.OID = LOADERMATERIAL) as TIPO_MATERIAL,
```

```
(select BL.FASE from [MSSUMM].[dbo].[CYCLE_DIM_BLOCK] BL where BL.OID = SOURCEBLOCK) as FASE_ORIGEN,
(select DE.NAME from [MSSUMM].[dbo].[CYCLE_DIM_DESTINATION] DE where DE.OID = SINKDESTINATION) as FASE_DESTINO,
QVC_TIEMPOOPERATIVO, LOADEDEFHDISTANCE, EMPTYEFHDISTANCE,
CYCLETYPE, WAITFORDUMPDURATION, CYCLEDURATION,
TRAVELLINGEMPTYDURATION, TRAVELLINGFULLDURATION, LOADINGDURATION,
QUEUINGATSOURCEDURATION, SPOTTINGATSOURCEDURATION,
QUEUINGATSINKDURATION, SPOTTINGATSINKDURATION, DUMPINGDURATION,
QUEUINGDURATION, SPOTTINGDURATION, HANGTIMEDURATION,
OPERATINGTIME, DELAYTIME, AVAILABLETIME, QUEUEATSINKDURATION,
QVC_PRODUCCIONSECUNDARIA, QVC_PERDIDADEPRODTIME,
QVC_PERDIDAINTERNA, QVC_PERDIDAEXTERNA, QVC_MANTOPERACIONAL,
QVC_MANTNOPROGRAMADO, QVC_MANTPROGRAMADO,
QVC_EVNOCONTROLABLES, QVC_NOPROGPARAPRODUCIR,
QVC_PRODUCCIONPRIMARIA, QVC_PERDIDADEPRODUCCION,
QVC_TIEMPOUTILIZADO, QVC_TIEMPONOUTILIZADO, QVC_TIEMPOPERDIDO,
QVC_OTROTIEMPO, QVC_TIEMPOMANTENIMIENTO,
QVC_TIEMPONOPROGRAMADO, QVC_TIEMPOPROGRAMADO,
QVC_TIEMPODISPONIBLE, QVC_TIEMPONODISPONIBLE,
QVC_TIEMPONOCONTROLABLE,
QVC_TIEMPOCALENDARIO from CYCLE_FACT_MAIN) temp
where STARTTIME>=  '2021-12-31 07:00:00' order by STARTTIME
```

3.2 Conexión al servidor a través del Jupyter Notebook

Para mostrar los datos extraídos utilizamos Python en Jupyter Notebook, ya que muestra una plataforma amigable para el análisis de datos y la aplicación de modelos de Machine Learning.

Figura 9

Conexión del Jupyter Notebook al servidor.

Conectando al servidor

In [1]:
```
import pandas as pd
import pyodbc
conn = pyodbc.connect('Driver={SQL Server};'
                      'Server=AISQVCFMSDB03;'
                      'Database=MSSUMM;'
                      'Trusted_Connection=yes;')
```

Nota: fuente elaboración propia.

3.3 Creación del Dataset en Jupyter Notebook

Hacemos uso de la libería pandas y de la herramienta Dataframe.

Figura 10

Importando datos del servidor.

In [2]:
```
#cursor = conn.cursor()

#sql_query = pd.read_sql_query('''
#select * from (
 #  select (dateadd(hour,-5, [STARTTIME_UTC])) AS STARTTIME ,
  #     (dateadd(hour,-5, [ENDTIME_UTC])) AS ENDTIME,
   #     ([PAYLOAD]/1000) as PAYLOAD,
    #    --CALENDAR,
     #   (SELECT M.NAME FROM [MSSUMM].[dbo].[CYCLE_DIM_MACHINE] M  WHERE M.OID = PRIMARYMACHINE) AS PRIMARYMACHINENAME,
      #  (SELECT M.CLASSNAME FROM [MSSUMM].[dbo].[CYCLE_DIM_MACHINE] M  WHERE M.OID = PRIMARYMACHINE) AS PRIMARYMACHINECLASSNAME
       # (SELECT M.NAME FROM [MSSUMM].[dbo].[CYCLE_DIM_MACHINE] M  WHERE M.OID = SECONDARYMACHINE) AS SECONDARYMACHINENAME,
        #(SELECT M.CLASSNAME FROM [MSSUMM].[dbo].[CYCLE_DIM_MACHINE] M  WHERE M.OID = SECONDARYMACHINE) AS SECONDARYMACHINECLASSN
#----(SELECT G.CREWID FROM [MSSUMM].[dbo].[CYCLE_DIM_CREW] G  WHERE G.OID = CREW) AS GUARDIA,
#----(SELECT MA.NAME FROM [MSSUMM].[dbo].[CYCLE_DIM_MATERIAL] MA  WHERE MA.OID = LOADERMATERIAL) AS MATERIAL,
#----(SELECT MA.GROUPLEVEL1 FROM [MSSUMM].[dbo].[CYCLE_DIM_MATERIAL] MA  WHERE MA.OID = LOADERMATERIAL) AS TIPO_MATERIAL,
#----(SELECT BL.FASE FROM [MSSUMM].[dbo].[CYCLE_DIM_BLOCK] BL  WHERE BL.OID = SOURCEBLOCK) AS FASE_ORIGEN,
#----(SELECT DE.NAME FROM [MSSUMM].[dbo].[CYCLE_DIM_DESTINATION] DE  WHERE DE.OID = SINKDESTINATION) AS FASE_DESTINO,
 #      QVC_TIEMPOOPERATIVO,
#----LOADEDEFHDISTANCE,
#----EMPTYEFHDISTANCE,
#----CYCLETYPE,
#----WAITFORDUMPDURATION,
#----CYCLEDURATION,
#----TRAVELLINGEMPTYDURATION,
#----TRAVELLINGFULLDURATION,
#----LOADINGDURATION,
#----QUEUINGATSOURCEDURATION,
#----SPOTTINGATSOURCEDURATION,
#----QUEUINGATSINKDURATION,
#----SPOTTINGATSINKDURATION,
#----DUMPINGDURATION,
```

Nota: fuente elaboración propia.

Figura 11

Visualización del DataSet.

```
In [5]: df = pd.read_csv('data.csv')
        #df = d1
        df.tail()
```

```
C:\Users\Victor\AppData\Local\Temp\ipykernel_11196\635077756.py:1: DtypeWarning: Columns (8) have mixed types. Specify dtype option on import or set low_memory=False.
  df = pd.read_csv('data.csv')
```

Out[5]:

	Unnamed: 0	STARTTIME	ENDTIME	PAYLOAD	PRIMARYMACHINENAME	PRIMARYMACHINECLASSNAME	SECONDARYMACHINENAME	SECONDARY
1422327	1422327	2023-08-09 16:57:02.000	2023-08-09 16:59:23.000	269.299988	SH001	CAT 7495	HT005	
1422328	1422328	2023-08-09 16:58:09.000	2023-08-09 17:01:01.000	204.699997	SH003	CAT 7495	HT004	
1422329	1422329	2023-08-09 16:58:49.000	2023-08-09 17:01:12.000	327.100006	SH002	CAT 7495	HT028	
1422330	1422330	2023-08-09 16:59:23.000	2023-08-09 17:02:32.000	285.600006	SH001	CAT 7495	HT003	
1422331	1422331	2023-08-09 17:01:12.000	2023-08-09 17:03:59.000	322.799988	SH002	CAT 7495	HT002	

5 rows × 56 columns

Nota: fuente elaboración propia.

Capitulo IV. Análisis y discusión de resultados

4.1 Identificación de variables

Nuestro DataSet tiene 31 variables entre las cuales está la variable dependiente “Productividad”. Se cuenta con 28,170 registros de viajes, siendo algunos de ellos de tipo numéricos (enteros, decimales) y objectos (texto), como se evidencia en la Figura 12.

Figura 12

Información del DataSet.

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 28170 entries, 7 to 80898
Data columns (total 31 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   AÑO                         28170 non-null  int64
 1   MES                         28170 non-null  int64
 2   FECHA                       28170 non-null  object
 3   TURNO                       28170 non-null  object
 4   GUARDIA                     28170 non-null  object
 5   HORA                        28170 non-null  int64
 6   SECONDARYMACHINECLASSNAME   28170 non-null  object
 7   SECONDARYMACHINENAME        28170 non-null  object
 8   PRIMARYMACHINECLASSNAME     28170 non-null  object
 9   MATERIAL                    28170 non-null  object
 10  TIPO_MATERIAL               28170 non-null  object
 11  FASE_ORIGEN                 28170 non-null  object
 12  FASE_DESTINO                28170 non-null  object
 13  TONELAJE                    28170 non-null  float64
 14  TIEMPO QUEUE                28170 non-null  float64
 15  TIEMPO SPOT                 28170 non-null  float64
 16  TIEMPO DE CARGUÍO           28170 non-null  float64
 17  TIEMPO VIAJE CARGADO        28170 non-null  float64
 18  TIEMPO QUEUE DESTINO        28170 non-null  float64
 19  TIEMPO SPOT DESTINO         28170 non-null  float64
 20  TIEMPO DE DESCARGA          28170 non-null  float64
 21  TIEMPO VIAJE VACÍO          28170 non-null  float64
 22  FACTOR CARGA                28170 non-null  float64
 23  EFH CARGADO                 28170 non-null  float64
 24  EFH VACÍO                   28170 non-null  float64
 25  DISPONIBILIDAD HT (%)       28170 non-null  float64
 26  TIEMPO HANG                 28170 non-null  float64
 27  DISPONIBILIDAD SH (%)       28170 non-null  float64
 28  TON                         28170 non-null  float64
 29  CICLO                       28170 non-null  float64
 30  PRODUCTIVIDAD               28170 non-null  float64
dtypes: float64(18), int64(3), object(10)
memory usage: 6.9+ MB
```

Nota: fuente elaboración propia.

4.2 Detección de nulos

A partir de la función “isna ()” de la libería pandas, calculamos que la cantidad de valores nulos es 0 para cada variable.

Figura 13

Cantidad de nulos por cada variable del DataSet.

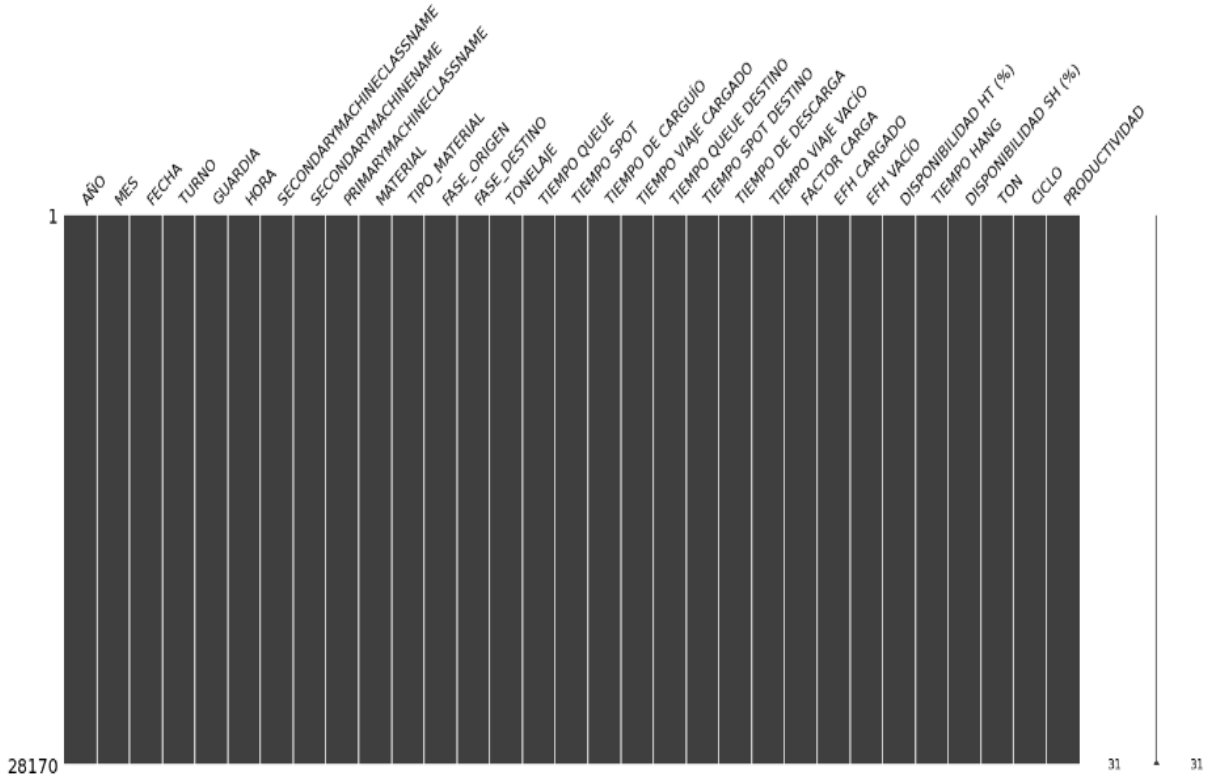
```
AÑO                          0
MES                          0
FECHA                        0
TURNO                        0
GUARDIA                      0
HORA                         0
SECONDARYMACHINECLASSNAME    0
SECONDARYMACHINENAME         0
PRIMARYMACHINECLASSNAME      0
MATERIAL                     0
TIPO_MATERIAL                0
FASE_ORIGEN                  0
FASE_DESTINO                 0
TONELAJE                     0
TIEMPO QUEUE                 0
TIEMPO SPOT                  0
TIEMPO DE CARGUÍO            0
TIEMPO VIAJE CARGADO         0
TIEMPO QUEUE DESTINO         0
TIEMPO SPOT DESTINO          0
TIEMPO DE DESCARGA           0
TIEMPO VIAJE VACÍO           0
FACTOR CARGA                 0
EFH CARGADO                  0
EFH VACÍO                    0
DISPONIBILIDAD HT (%)        0
TIEMPO HANG                  0
DISPONIBILIDAD SH (%)        0
TON                          0
CICLO                        0
PRODUCTIVIDAD                0
dtype: int64
```

Nota: fuente elaboración propia.

Corroboramos que nuestro DataSet no presenta valores nulos de manera visual con la libería “Missingno”, donde los valores nulos se deben mostrar en líneas blancas por cada variable para línea de registro de datos, caso contrario se verán todo uniforme y sin espacios como queda en la Figura 14 evidenciado.

Figura 14

Visualización de nulos por cada variable del DataSet.



Nota: fuente elaboración propia.

4.3 Análisis estadístico descriptivo de las variables

Realizando un análisis descriptivo inicial de cada una de las variables numéricas, encontramos su media, desviación estándar, valor mínimo, valor máximo y cuartiles, los cuales están detallados en la Tabla 3.

Tabla 3

Análisis descriptivo de las variables numéricas.

Variables	Media	std	Mín	25%	50%	75%	Máx
Tiempo queue (min)	2.95	140.44	0.00	1.38	2.49	3.95	28.08
Tiempo spot (min)	0.69	16.01	0.00	0.57	0.64	0.75	9.68
Tiempo de carguío (min)	1.93	53.14	0.02	1.20	1.55	2.60	10.07
Tiempo viaje cargado (min)	9.55	276.42	0.00	6.00	8.79	12.51	50.32
Tiempo queue destino (min)	0.82	80.32	0.00	0.00	0.31	1.10	27.20
Tiempo spot destino (min)	0.77	47.67	0.00	0.54	0.61	0.68	23.97
Tiempo de descarga (min)	0.83	34.75	0.00	0.67	0.74	0.83	24.25
Tiempo de viaje vacío (min)	8.21	243.88	0.00	4.96	7.58	10.88	42.27
Factor de carga (t)	310.10	14.09	3.30	303.70	311.38	318.30	368.40
EFH cargado (km)	3.86	1897.56	0.00	2.45	3.79	4.95	35.43
EFH vacío (km)	3.88	2098.34	0.00	2.28	3.67	4.93	24.38

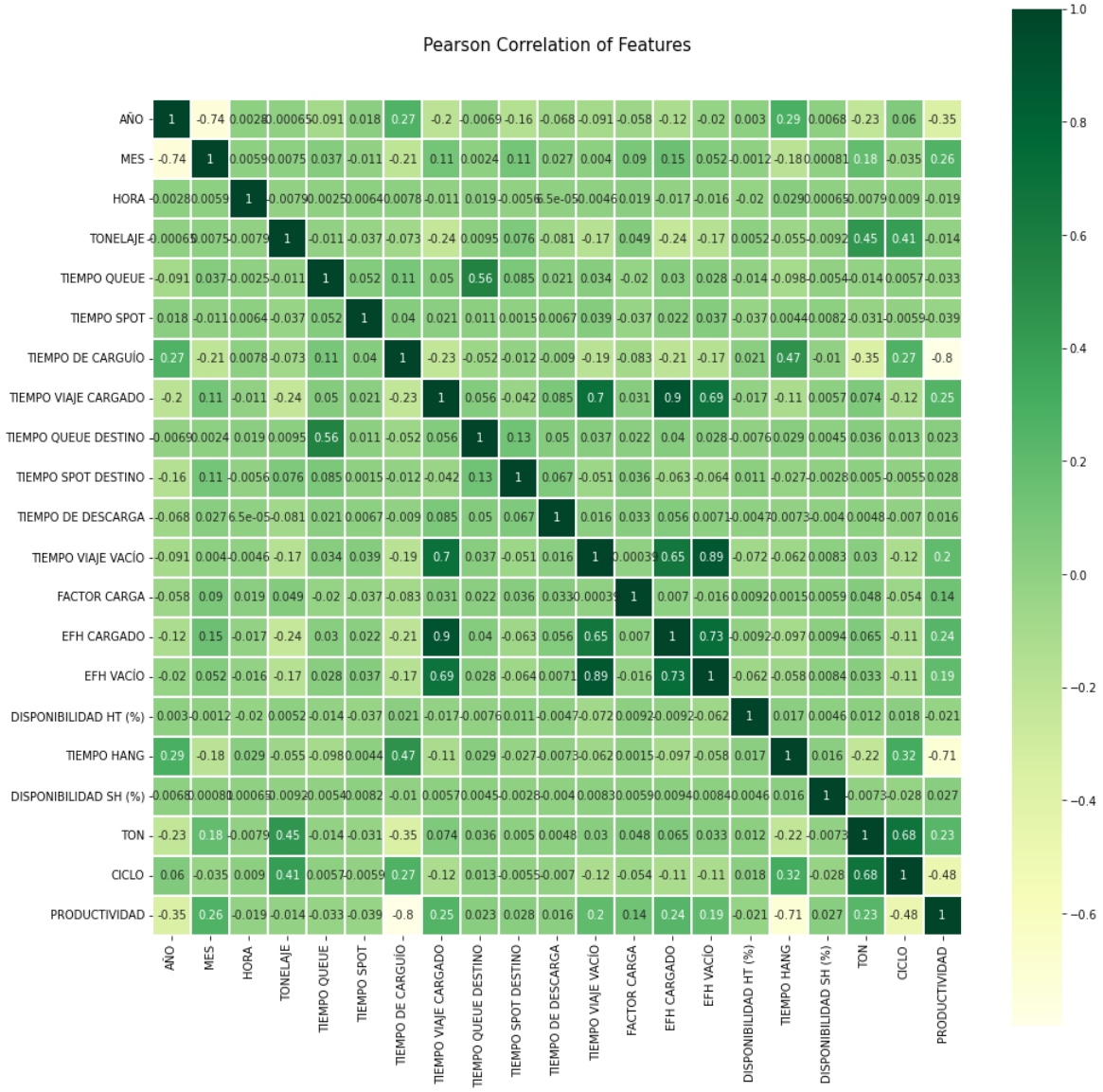
Disponibilidad de HTs (%)	98.98	6.62	3.82	100.00	100.00	100.00	100.00
Tiempo hang (min)	1.84	59.80	0.00	1.18	1.52	2.15	6.81
Disponibilidad de SH (%)	99.94	1.10	61.22	100.00	100.00	100.00	100.00
Productividad (t/h)	4781.68	1819.42	2000.06	2946.81	5197.44	6045.19	10769.43

Nota: fuente elaboración propia.

Para entender la relación entre las variables graficamos una matriz de correlación.

Figura 15

Correlación entre las variables del Dataset.



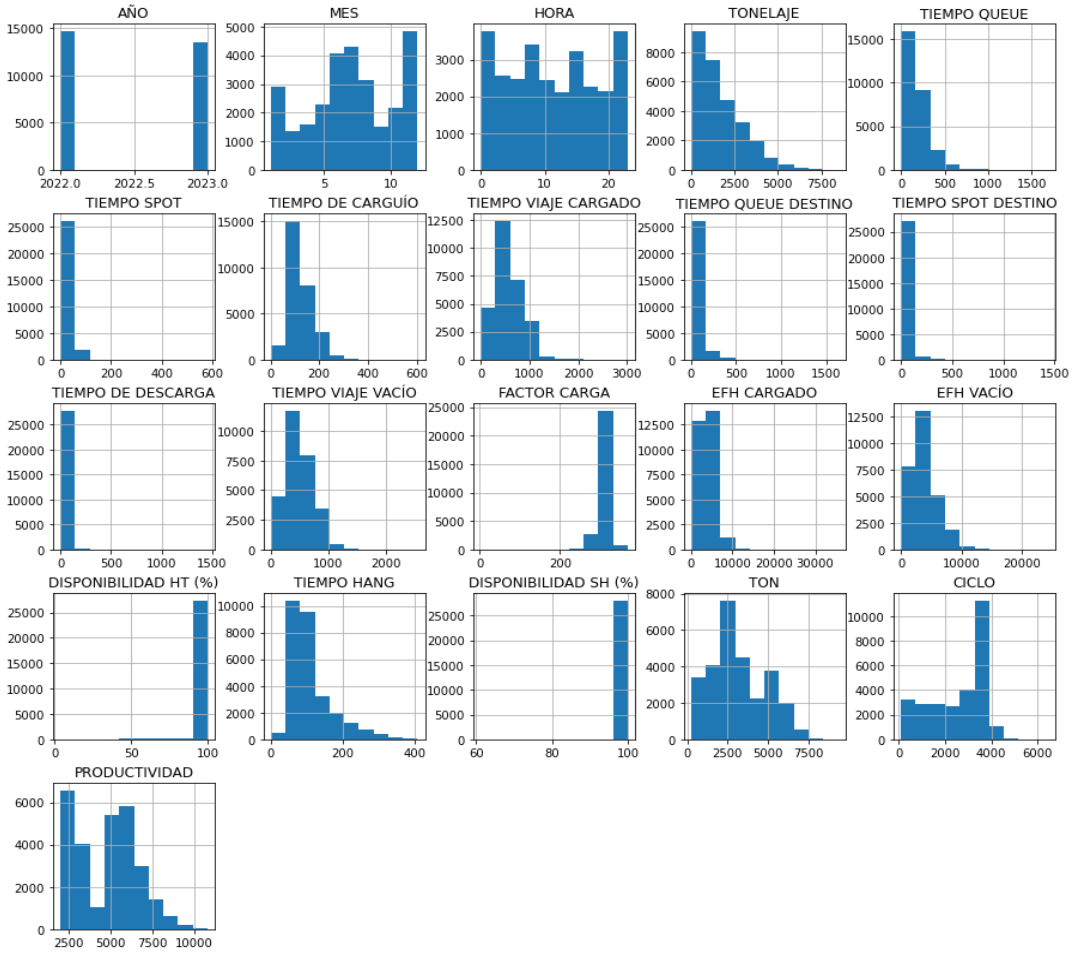
Nota: fuente elaboración propia.

Observamos que la correlación es baja y solo en la diagonal de nuestra matriz es alta, por lo cual el modelo de regresión lineal queda descartado, al no cumplir una de las condiciones iniciales para su aplicación.

Para entender el comportamiento de las variables realizamos un diagrama general de distribución del DataSet (los tiempos están en segundos).

Figura 16

Distribución de las variables del Dataset.

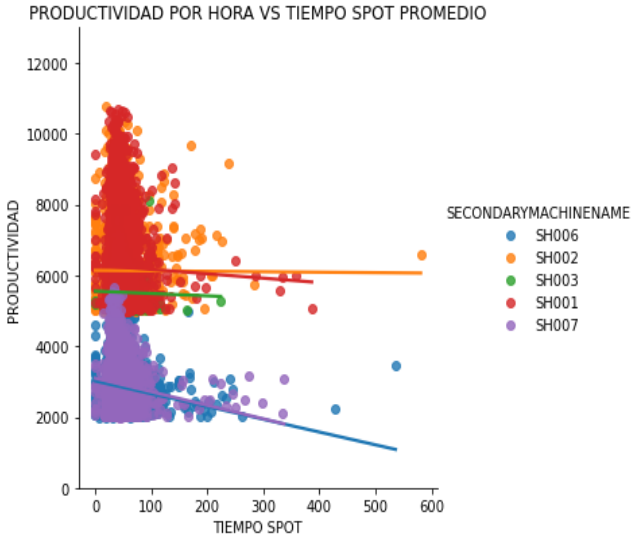


Nota: fuente elaboración propia.

Para un análisis más a detalle realizamos gráficos de dispersión, observando que la productividad de las palas SH001, SH002 y SH003 están por encima de los 5,000 t/h ya que forman la flota CAT 7495 (Palas eléctricas), mientras que las SH006 y SH007 logran alcanzar un rango menor a los 6,000 t/h pertenciendo a la flota CAT 6060BH (Palas hidráulicas).

Figura 17

Dispersión de la productividad vs el tiempo spot (tiempo de cuadrado del camión).

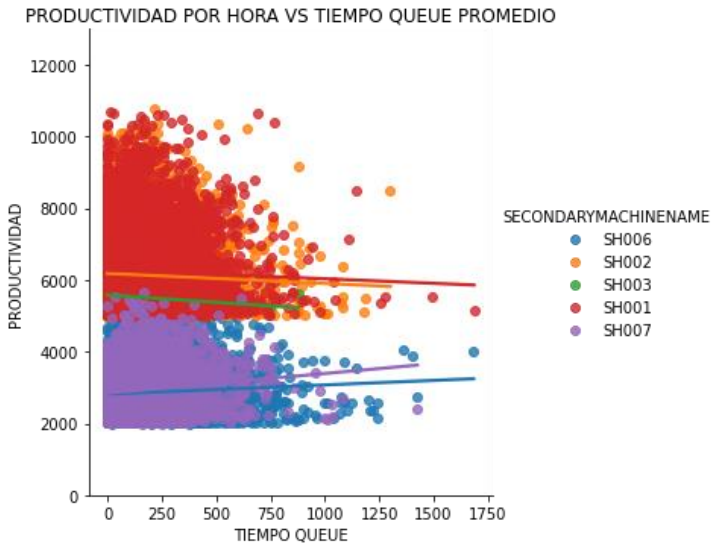


Nota: fuente elaboración propia.

Se evidencia una relación inversa entre ambas variables, la concentración de los datos se encuentra cuando el tiempo spot es menor de los 120 s (2 minutos).

Figura 18

Dispersión de la productividad vs el tiempo queue (tiempo de cola del camión).

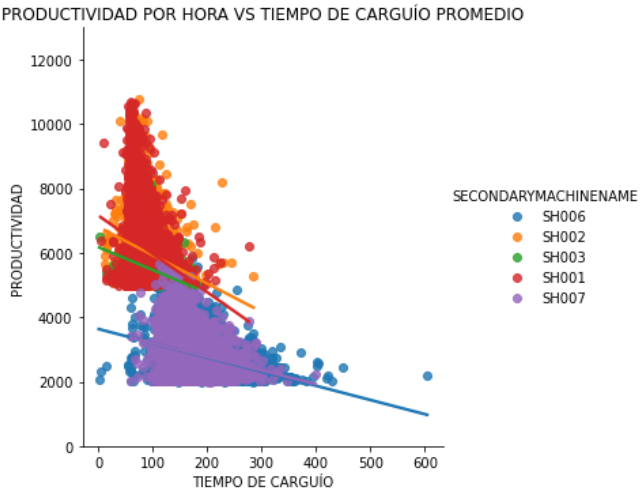


Nota: fuente elaboración propia.

Presenta una relación inversa entre ambas variables, la concentración de los datos se encuentra cuando el tiempo queue es menor de los 750 s (12.5 minutos).

Figura 19

Dispersión de la productividad vs el tiempo de carguío.

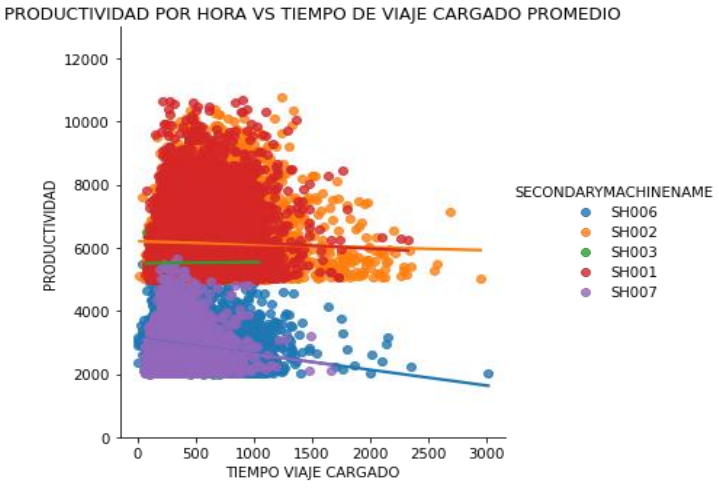


Nota: fuente elaboración propia.

Presenta una relación inversa entre ambas variables, la concentración de los datos se encuentra cuando el tiempo de carguío es menor de los 120 s (2 minutos) para los CAT 7495 y 300 s (5 minutos) para los CAT 6060BH.

Figura 20

Dispersión de la productividad vs el tiempo de viaje cargado.

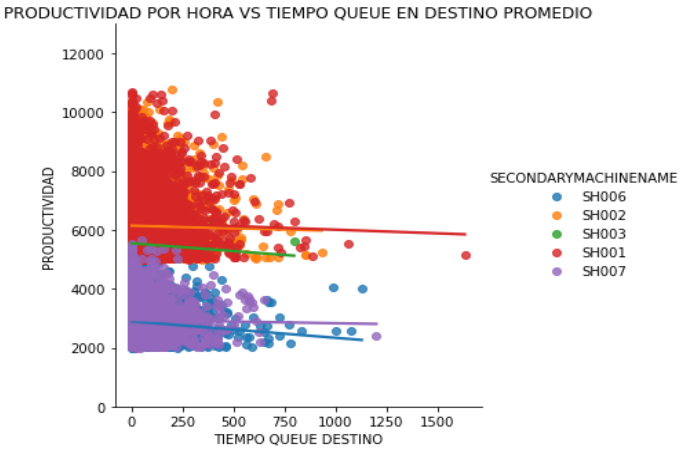


Nota: fuente elaboración propia.

Presenta una relación inversa entre ambas variables, la concentración de los datos se encuentra cuando el tiempo de viaje cargado es menor de los 1200 s (20 minutos).

Figura 21

Dispersión de la productividad vs el tiempo queue en el destino (tiempo de cola del camión).

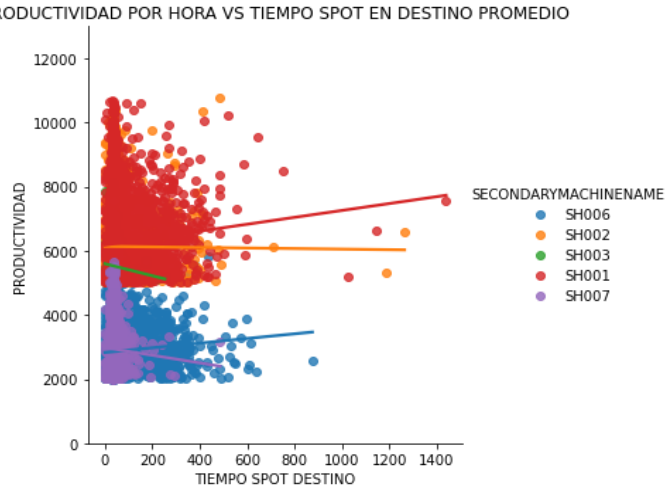


Nota: fuente elaboración propia.

Presenta una relación inversa entre ambas variables, la concentración de los datos se encuentra cuando el tiempo queue en el destino es menor de los 300 s (5 minutos).

Figura 22

Dispersión de la productividad vs el tiempo spot en el destino (tiempo de cuadrado).

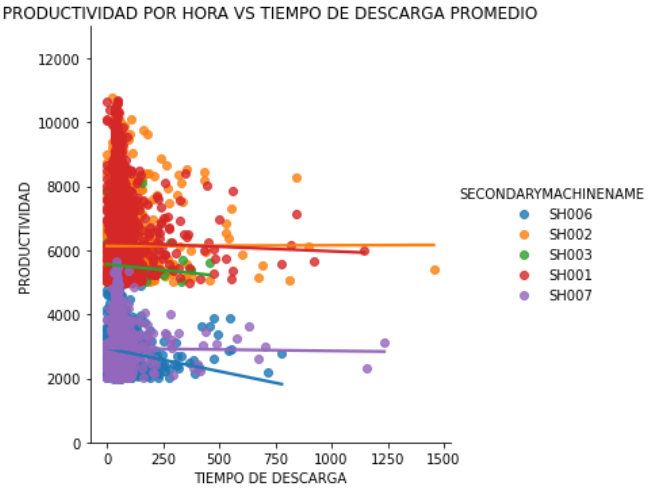


Nota: fuente elaboración propia.

Presenta una relación inversa entre ambas variables, la concentración de los datos se encuentra cuando el tiempo spot en el destino es menor de los 240 s (4 minutos).

Figura 23

Dispersión de la productividad vs el tiempo de descarga.

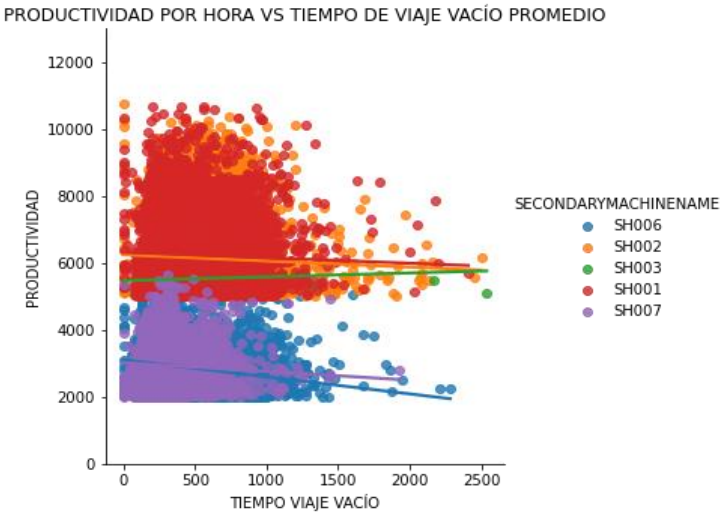


Nota: fuente elaboración propia.

Presenta una relación inversa entre ambas variables, la concentración de los datos se encuentra cuando el tiempo de descarga es menor de los 180 s (3 minutos).

Figura 24

Dispersión de la productividad vs el tiempo de viaje vacío.

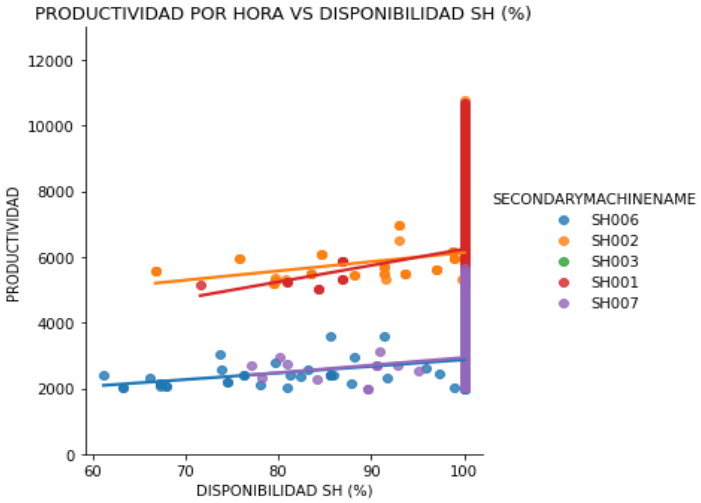


Nota: fuente elaboración propia.

Presenta una relación inversa entre ambas variables, la concentración de los datos se encuentra cuando el tiempo de viaje vacío es menor de los 1200 s (20 minutos).

Figura 25

Dispersión de la productividad vs la disponibilidad de las palas (%).

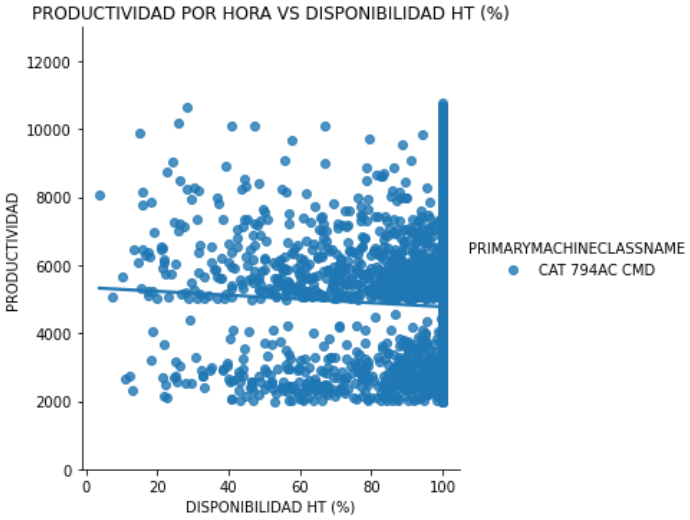


Nota: fuente elaboración propia.

Presenta una relación directa entre ambas variables, siendo el objetivo el 100%.

Figura 26

Dispersión de la productividad vs la disponibilidad de los camiones (%).

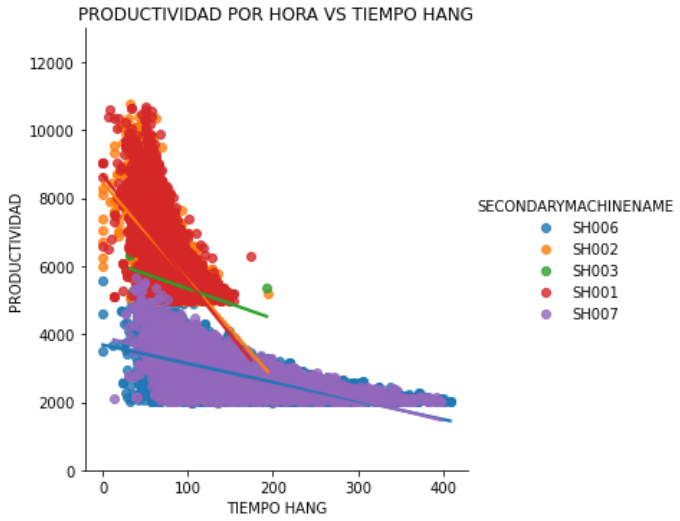


Nota: fuente elaboración propia.

Presenta una relación directa entre ambas variables, siendo el objetivo el 100%

Figura 27

Dispersión de la productividad vs el tiempo hang (tiempo sin camiones) de las palas.

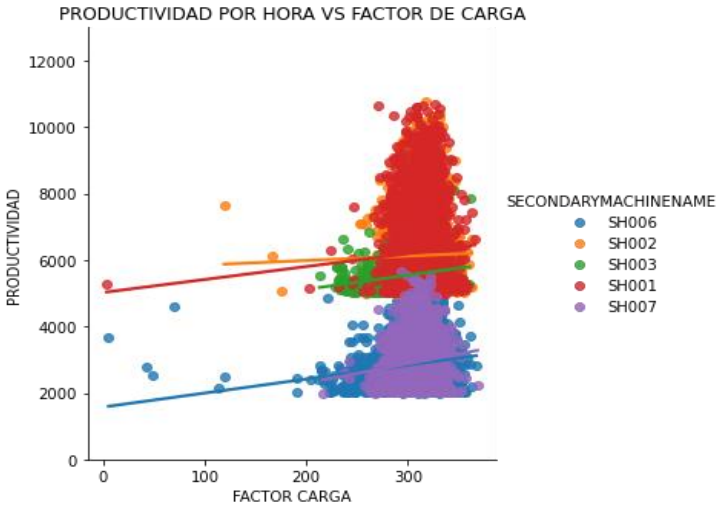


Nota: fuente elaboración propia.

Presenta una relación inversa entre ambas variables, la concentración de los datos se encuentra cuando el tiempo hang es menor de los 100 s (1,7 minutos) para los CAT 7495 y 240 s (4 minutos) para los CAT 6060BH.

Figura 28

Dispersión de la productividad vs el factor de carga de los camiones.

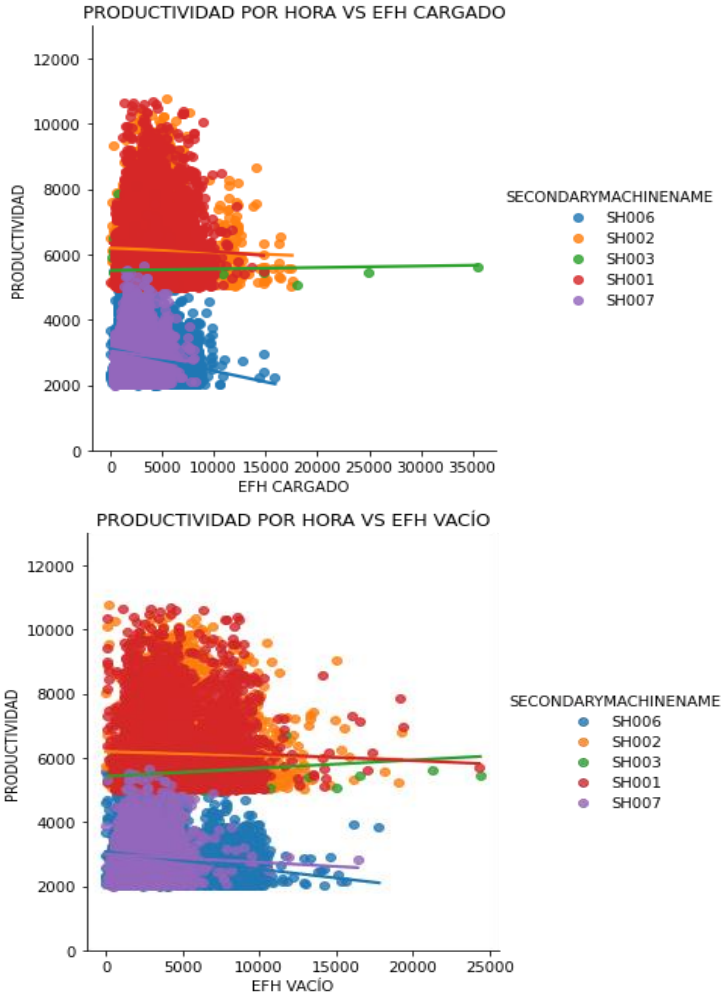


Nota: fuente elaboración propia.

Presenta una relación directa entre ambas variables, la concentración de los datos se encuentra alrededor de las 300 toneladas.

Figura 29

Dispersión de la productividad vs EFH (distancia horizontal equivalente) cuando está cargado y vacío.



Nota: fuente elaboración propia.

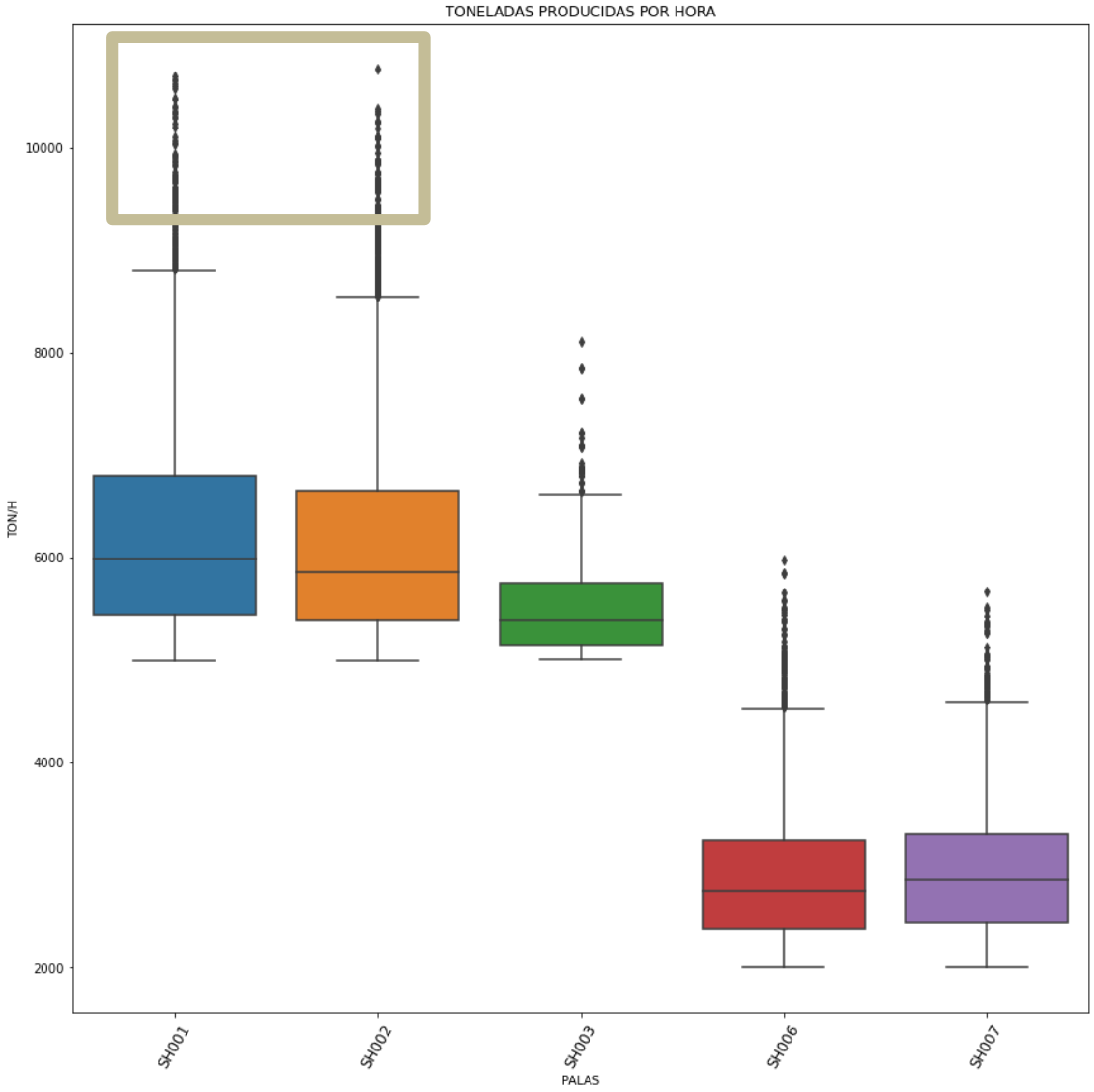
Presenta una relación inversa entre ambas variables, la concentración de los datos se encuentra cuando es menor a los 8,000 m (8 km) cuando está cargado y 10,000 m (10 km) cuando está vacío.

4.4 Detección de Outliers y limpieza de datos

Para validar los outliers para cada variable se construyó un diagrama de caja o bigotes.

Figura 30

Diagrama de cajas para las toneladas producidas por cada pala.

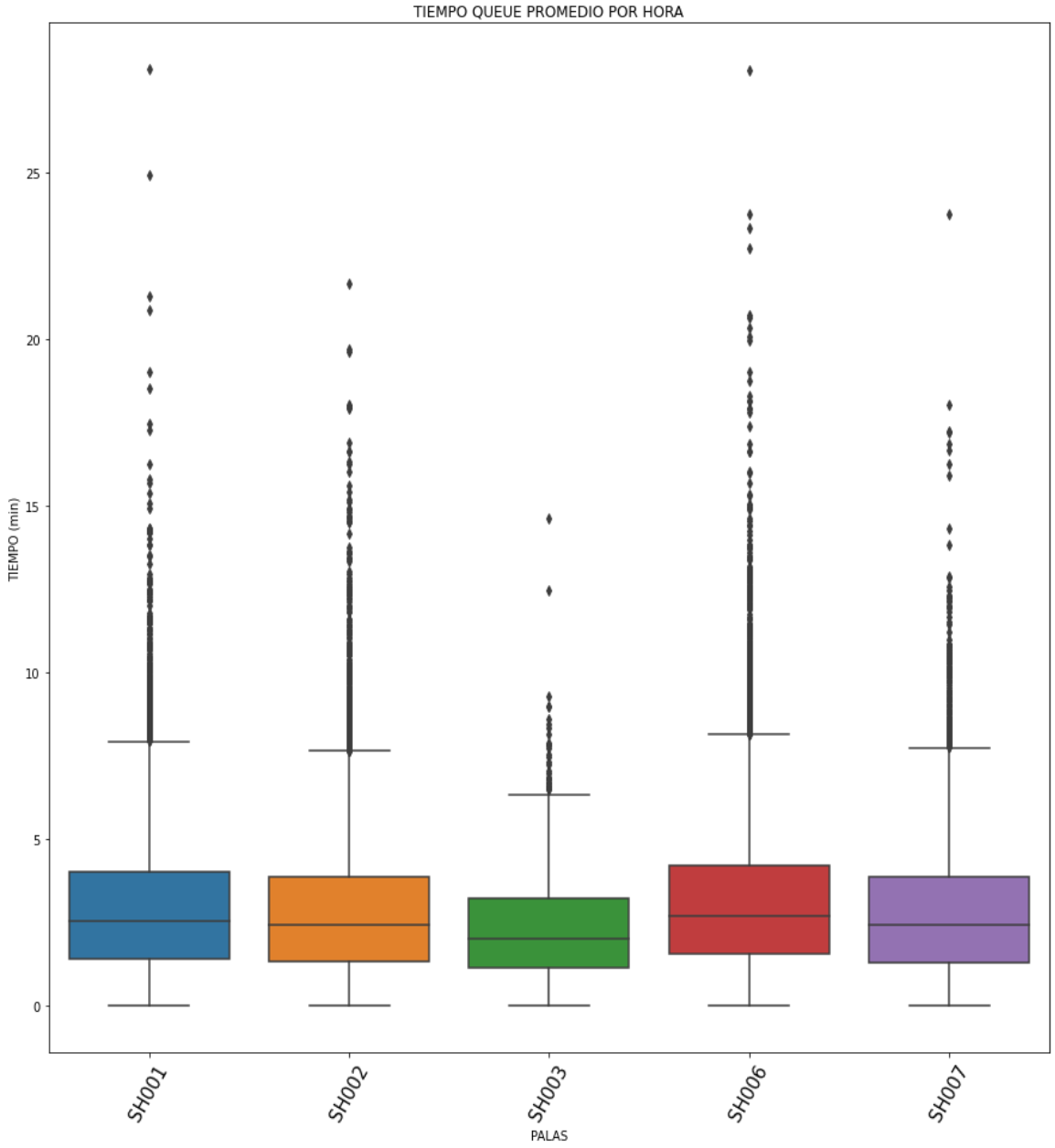


Nota: fuente elaboración propia.

Se observa que 0.87% de los tonelajes producidos por las palas CAT 7495 son mayores a los 9,000 t/h, los cuales son considerados outliers ya que la operación minera es nueva y rara vez sobrepasa los este límite, dada las condiciones de operaciones actuales.

Figura 31

Diagrama de cajas para el tiempo queue promedio por cada pala.

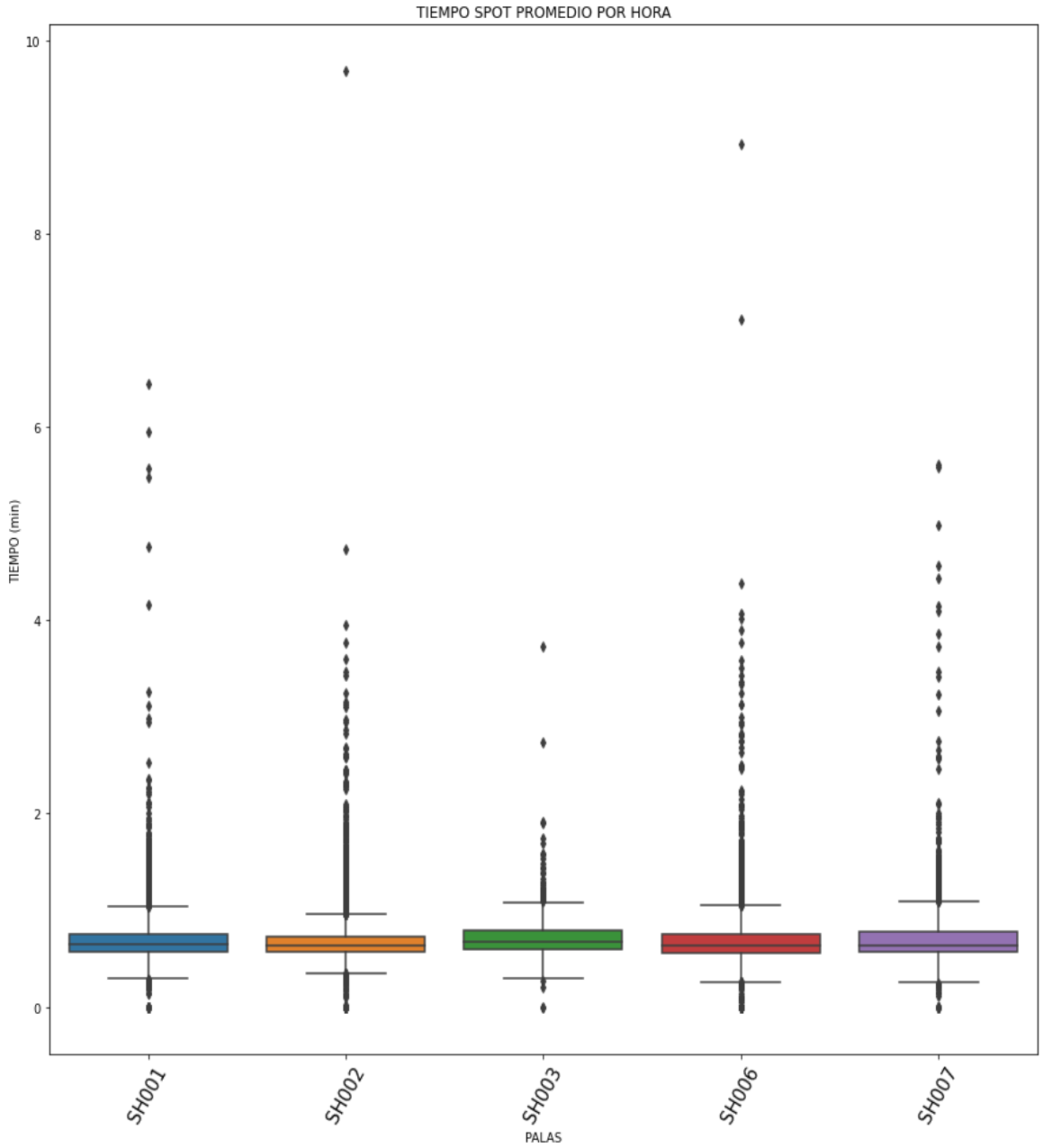


Nota: fuente elaboración propia.

El 25% de los tiempos queue son menores de 1 minuto y 75% son menores o iguales a los 8 minutos en promedio.

Figura 32

Diagrama de cajas para el tiempo spot promedio por cada pala.

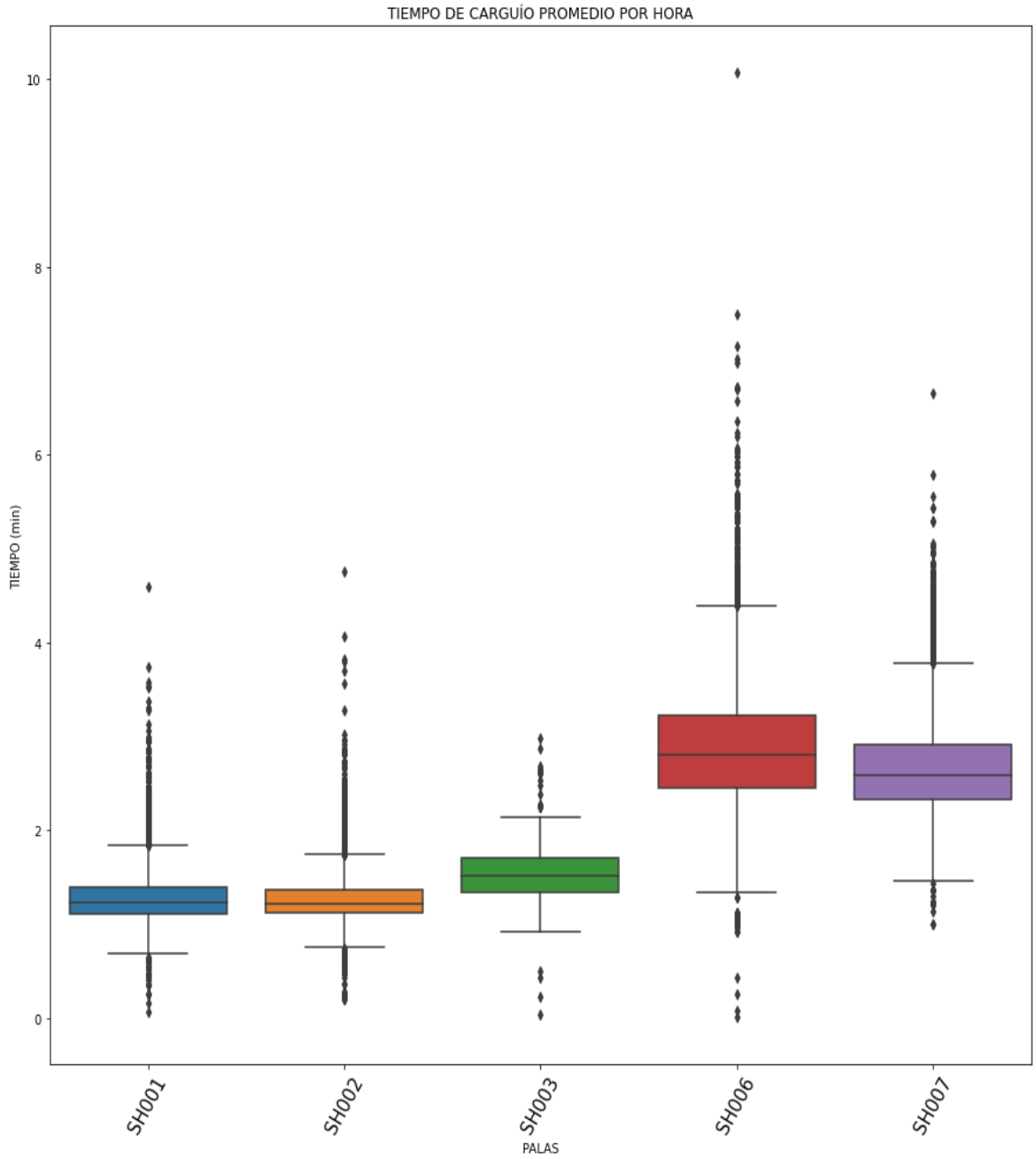


Nota: fuente elaboración propia.

El 50% de los tiempos spot son menores de ½ minuto y 75% son menores o iguales a los 1 minutos en promedio.

Figura 33

Diagrama de cajas para el tiempo de carguío promedio por cada pala.

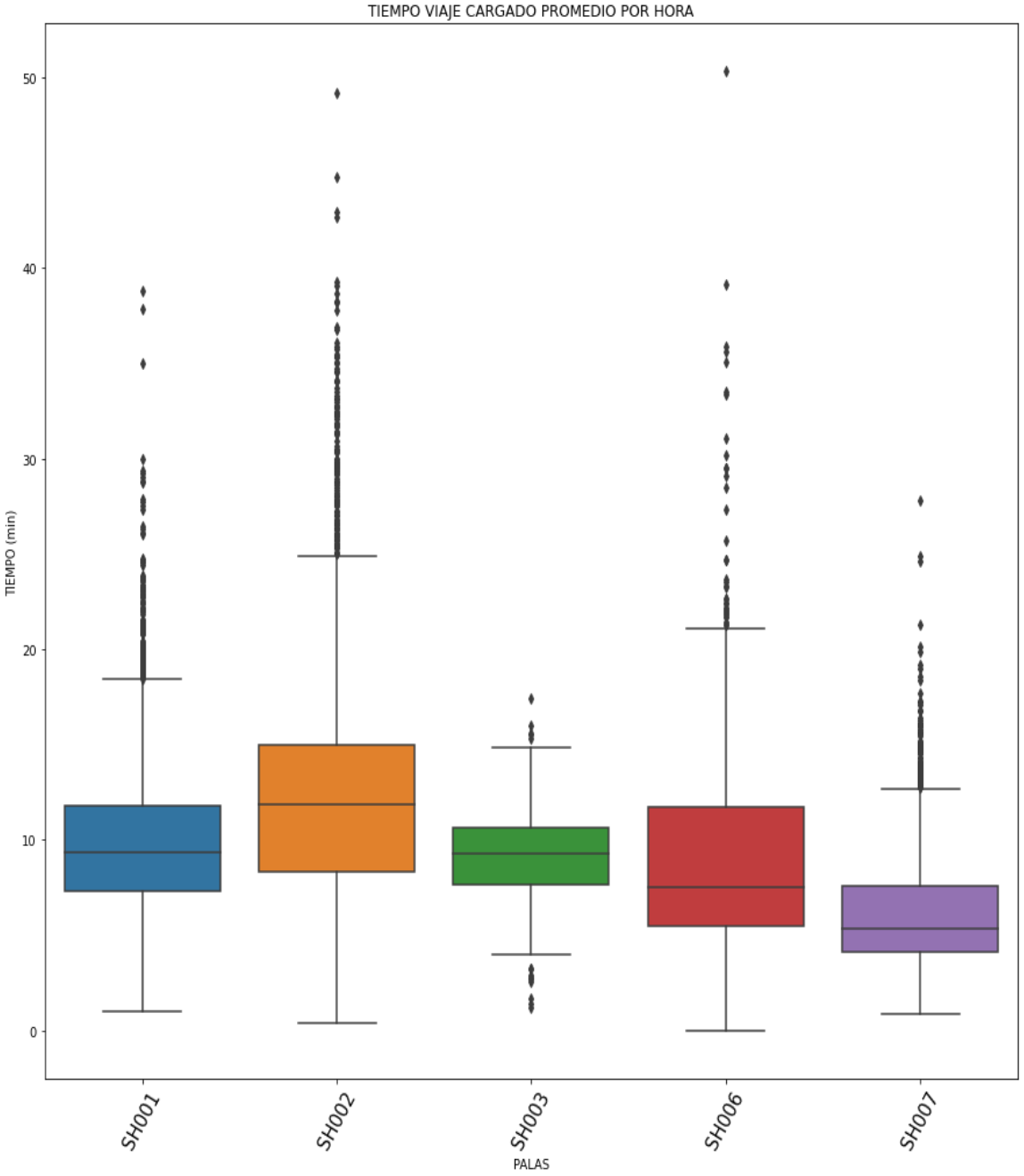


Nota: fuente elaboración propia.

El 50% de los tiempos carguío para las palas CAT 7495 son menores de 1 minuto y 75% son menores o iguales a los 2 minutos en promedio; mientras que para las palas CAT 6060BH 25% son menores de 1.5 minutos y el 75% menores a 4 minutos.

Figura 34

Diagrama de cajas para el tiempo de viaje cargado promedio por pala.

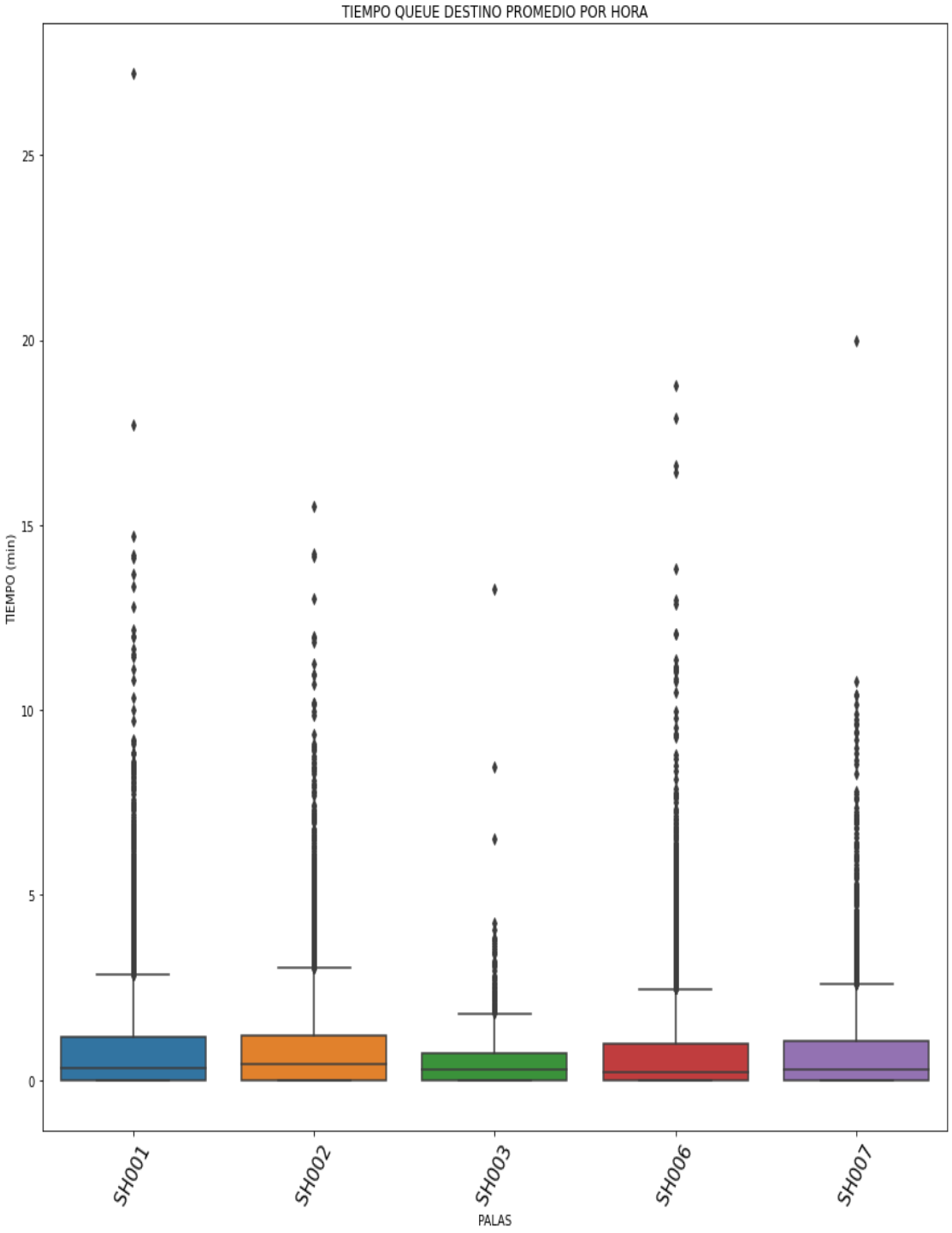


Nota: fuente elaboración propia.

Para la SH001 el 50% de los tiempos de viaje cargado promedio son menores a 9 minutos y el 75% menores a 18 minutos; para la SH002 el 75% es menor a 25 minutos; para la SH003 el 75% es menor a 15 minutos; mientras que para la SH006 75% es menor a 22 minutos y para la SH007 es menor o igual a 13 minutos.

Figura 35

Diagrama de cajas para el tiempo queue promedio en el destino por pala.

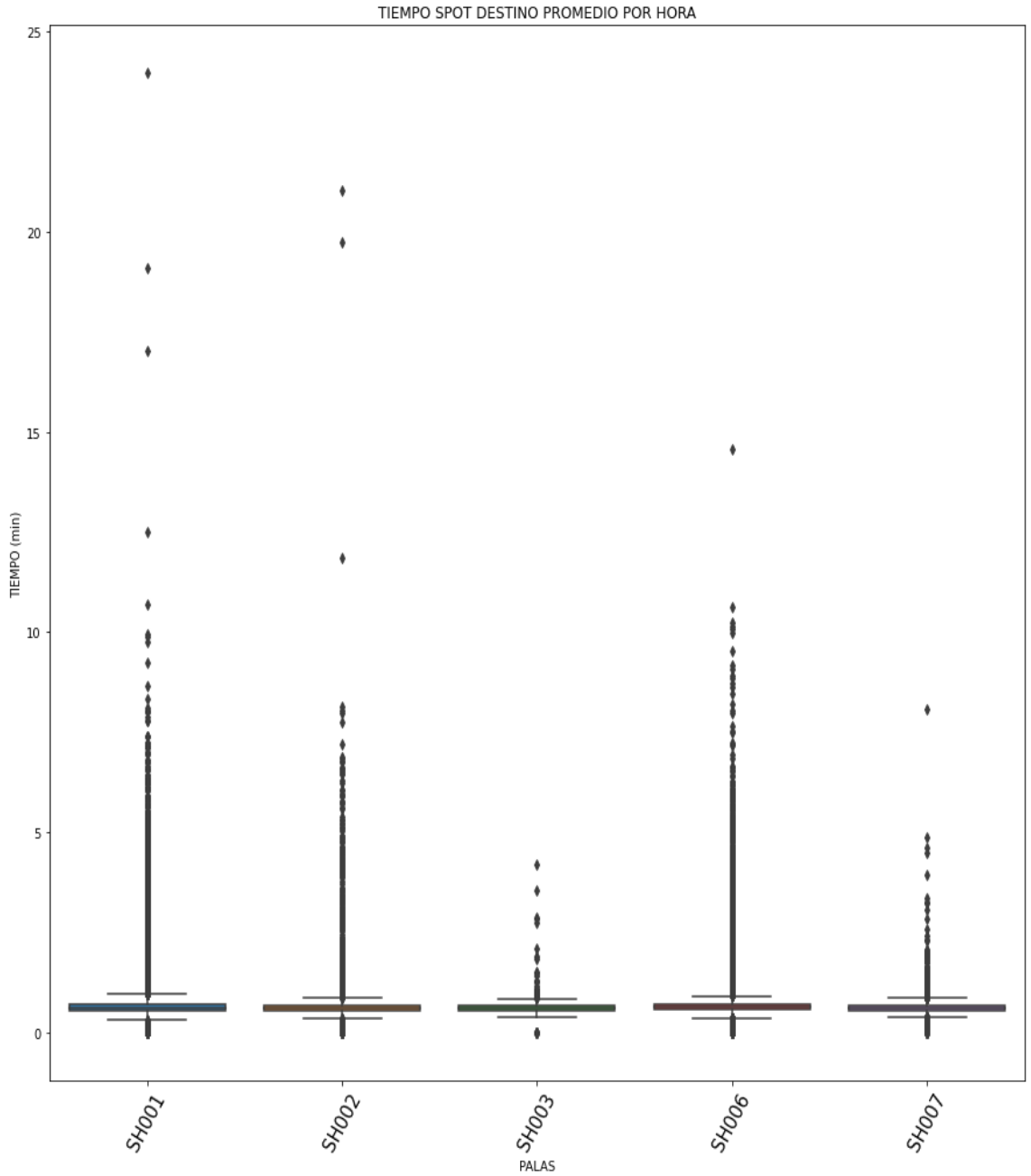


Nota: fuente elaboración propia.

El 75% de los tiempos queue en el destino para las SH001 y SH002 están por debajo de 3 minutos; para las palas SH003, SH006 y SH007 por debajo de ½ minuto.

Figura 36

Diagrama de cajas para el tiempo spot promedio en el destino por pala.

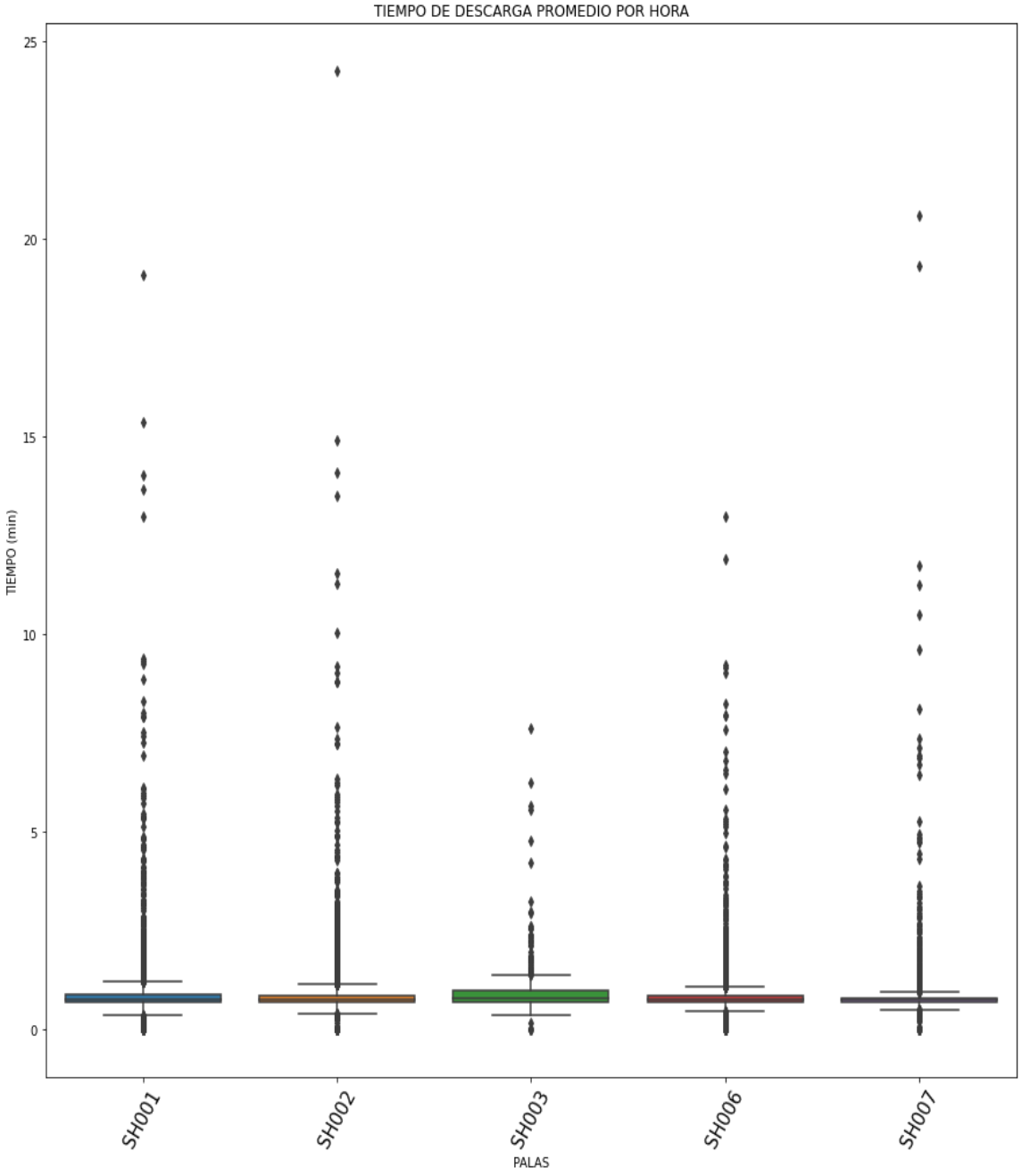


Nota: fuente elaboración propia.

El 25% de los tiempos spot en el destino están por debajo del ½ minuto y el 75% por debajo de 1.2 minutos.

Figura 37

Diagrama de cajas para el tiempo de descarga promedio por cada pala.

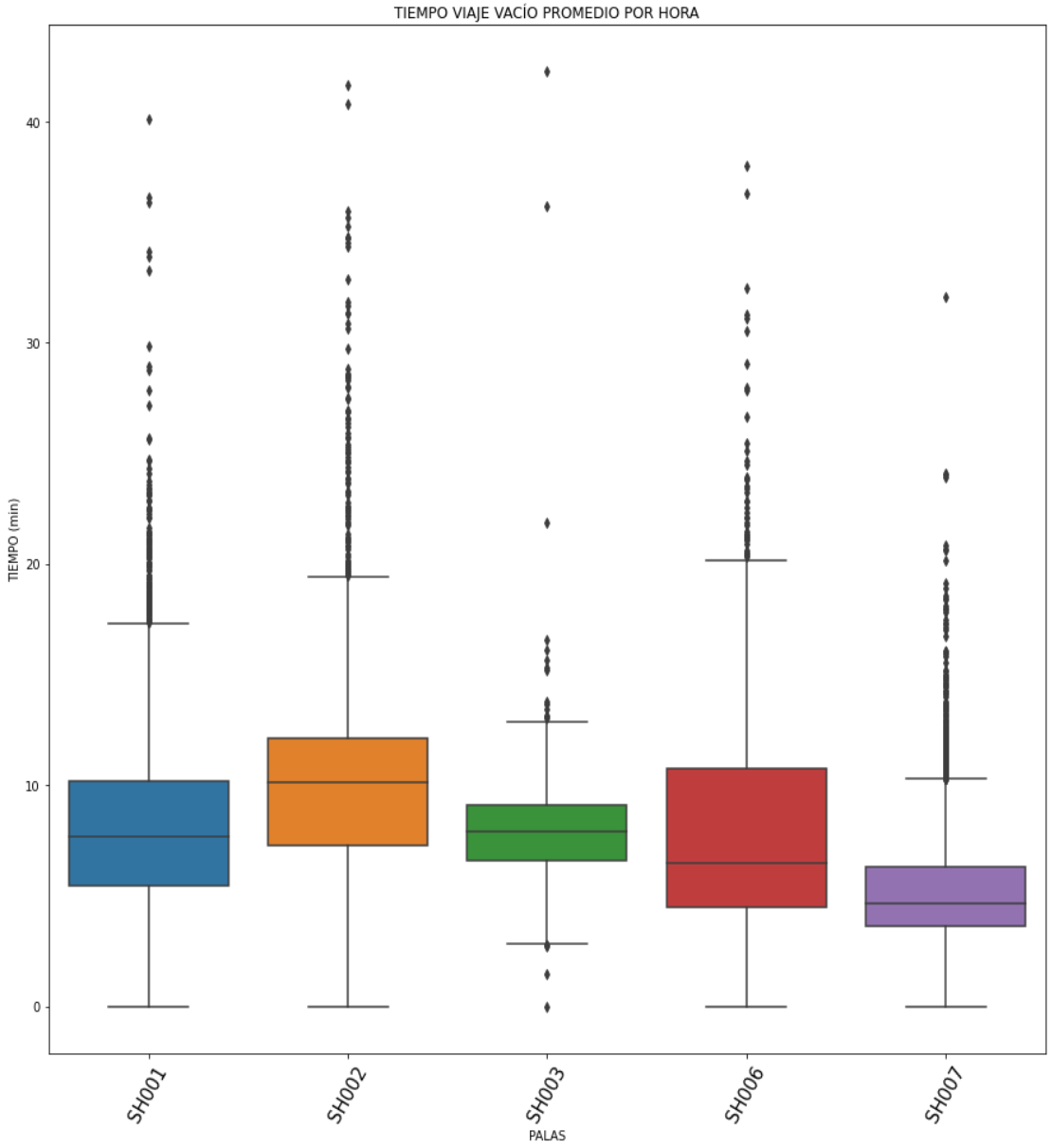


Nota: fuente elaboración propia.

El 25% de los tiempos de descarga están por debajo del ½ minuto y el 75% por debajo de 1 minuto.

Figura 38

Diagrama de cajas para el tiempo de viaje vacío promedio por cada pala.

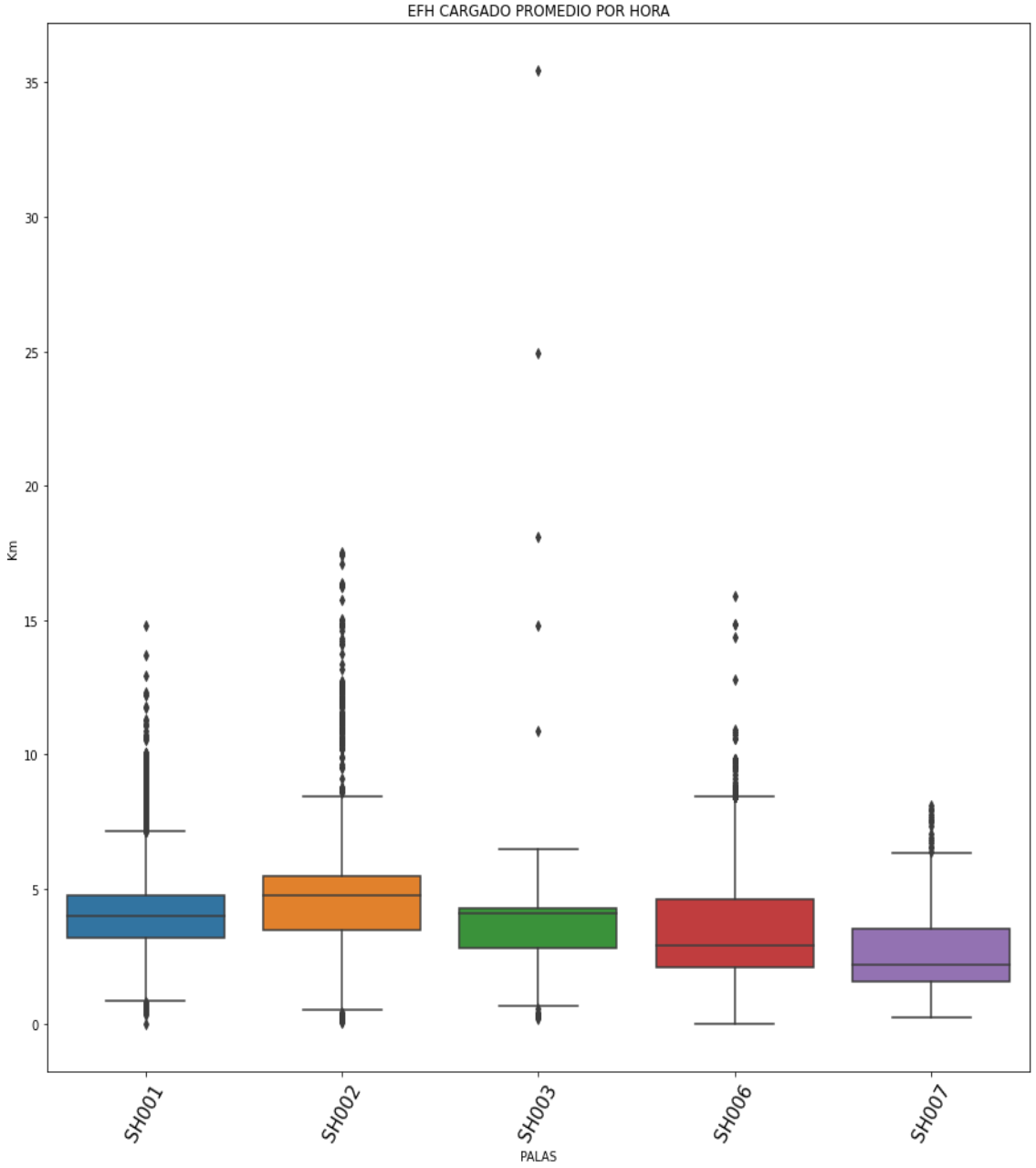


Nota: fuente elaboración propia.

Para la SH001 el 50% de los tiempos de viaje vacío promedio son menores a 8 minutos y el 75% menores a 18 minutos; para la SH002 el 75% es menor a 20 minutos; para la SH003 el 75% es menor a 12 minutos; mientras que para la SH006 75% es menor 20 minutos y para la SH007 es menor o igual a 10 minutos.

Figura 39

Diagrama de cajas para el EFH promedio cuando está cargado por pala.

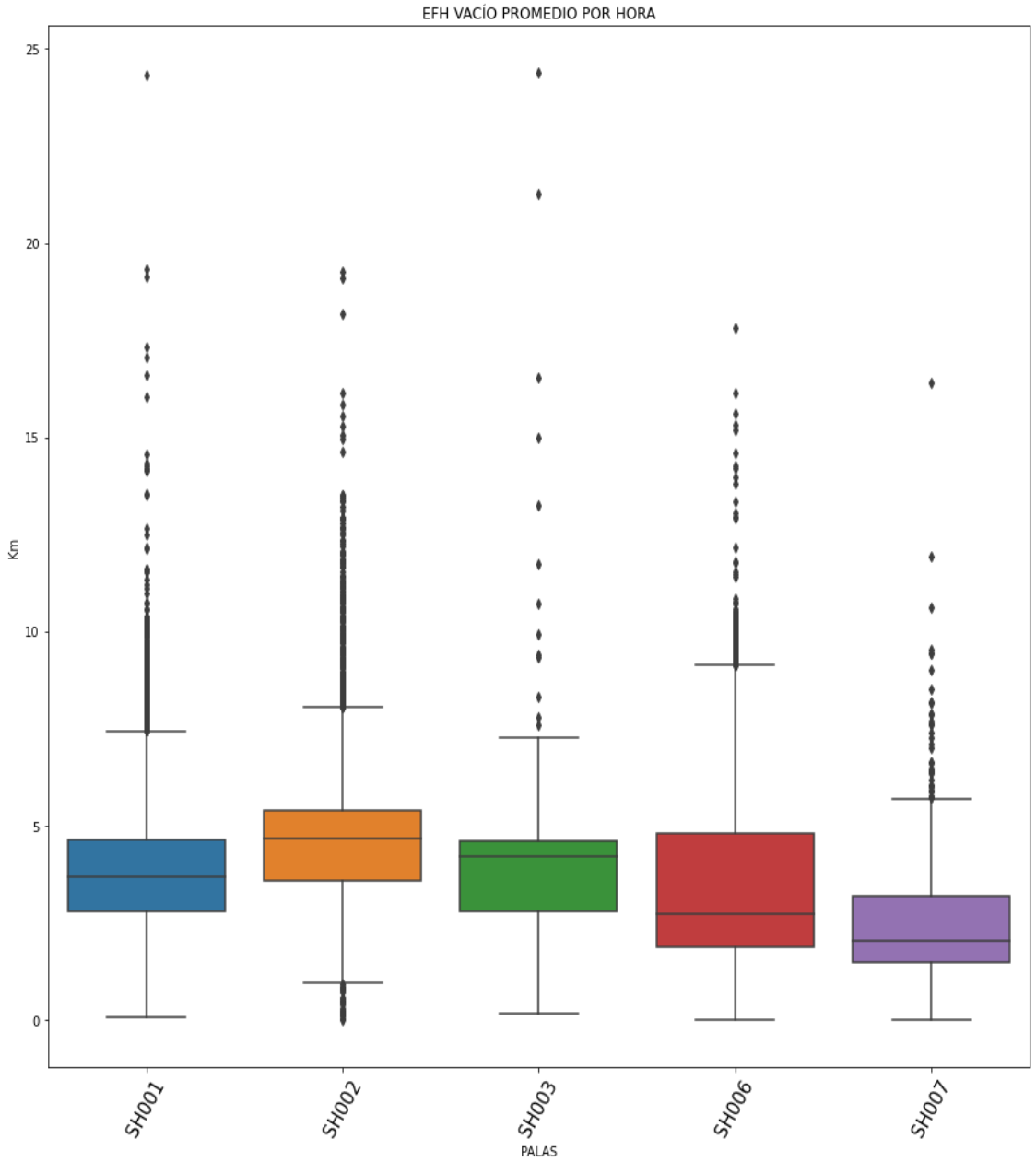


Nota: fuente elaboración propia.

Para la SH001 el 50% de los EFH promedio son menores a 4 km y el 75% menores a 7 km; para la SH002 el 75% es menor a 8 km; para la SH003 el 75% es menor a 6 km; mientras que para la SH006 75% es menor 8 km y para la SH007 es menor o igual a 6 km.

Figura 40

Diagrama de cajas para el EFH promedio cuando está vacío por pala.

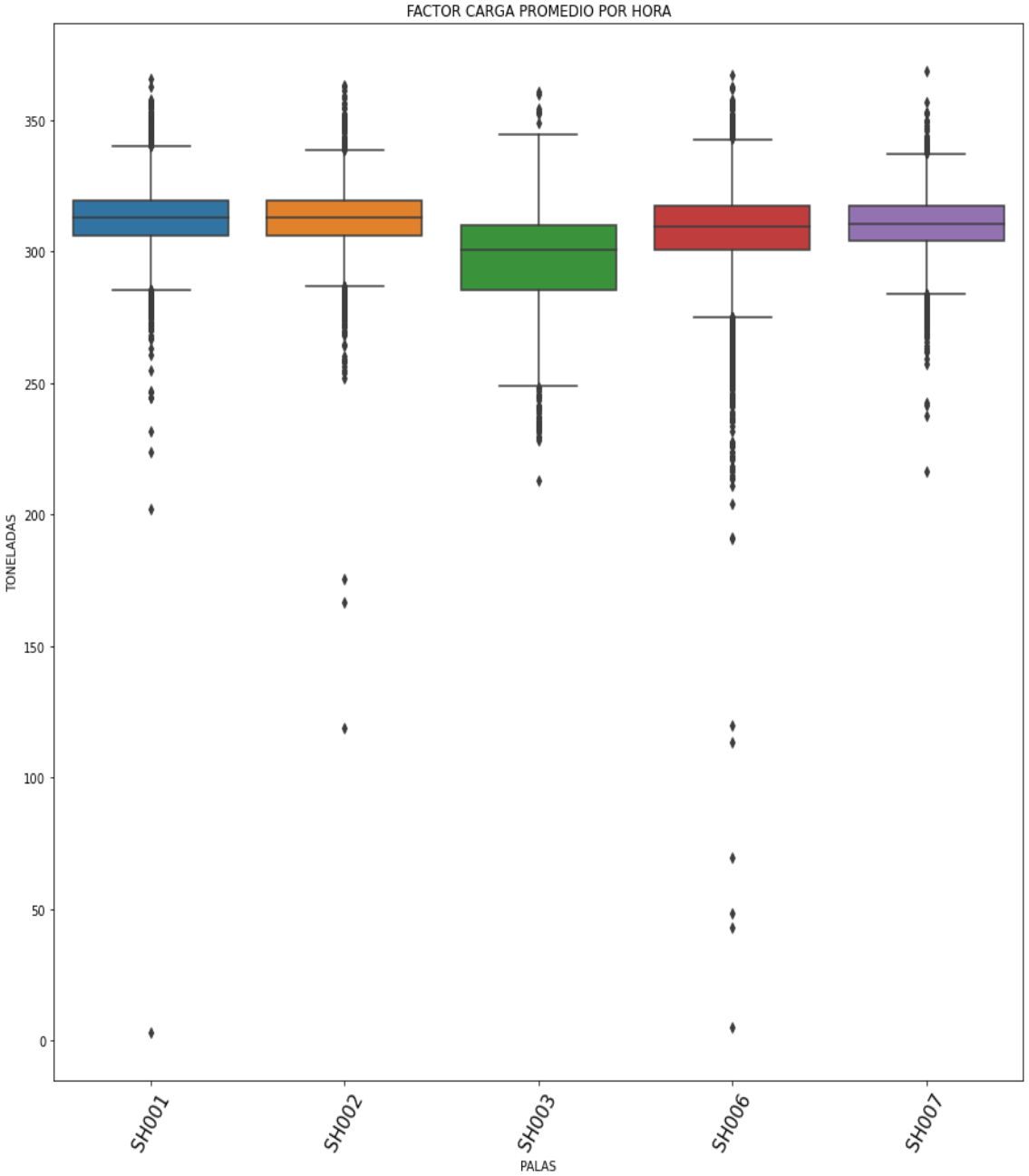


Nota: fuente elaboración propia.

Para la SH001 el 50% de los EFH promedio son menores a 4 km y el 75% menores a 7.5 km; para la SH002 el 75% es menor a 8 km; para la SH003 el 75% es menor a 7 km; mientras que para la SH006 75% es menor 9 km y para la SH007 es menor o igual a 6 km.

Figura 41

Diagrama de cajas para el factor de carga promedio por cada pala.

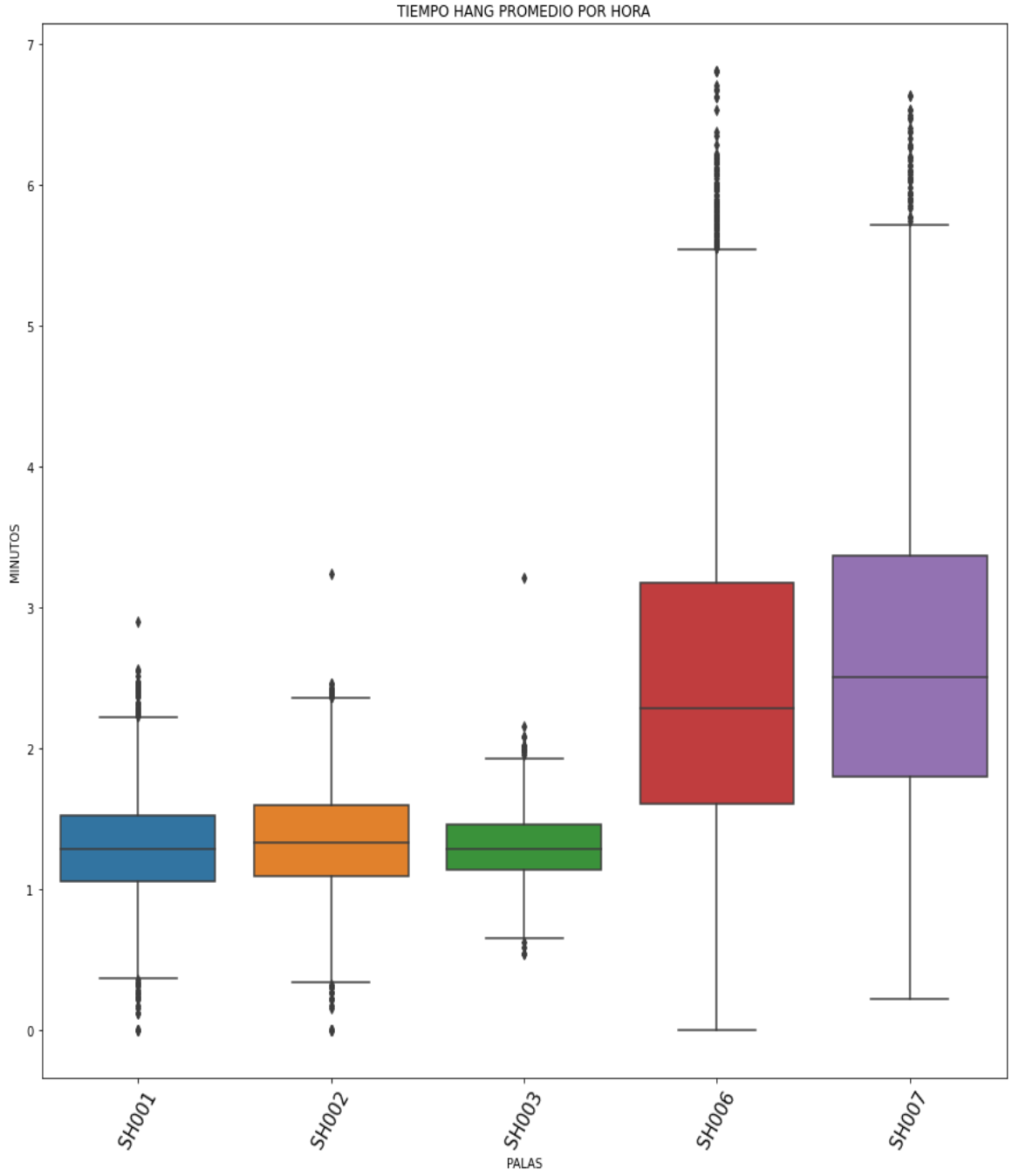


Nota: fuente elaboración propia.

Para la SH001 el 25% de los factores de carga promedio son menores a 280 ton y el 75% menores a 340 ton; para la SH002 el 75% es menor a 330 ton; para la SH003 y SH006 el 75% es menor a 350 ton; mientras que para la SH007 es menor o igual a 330 ton.

Figura 42

Diagrama de cajas para el tiempo hang promedio por cada pala.



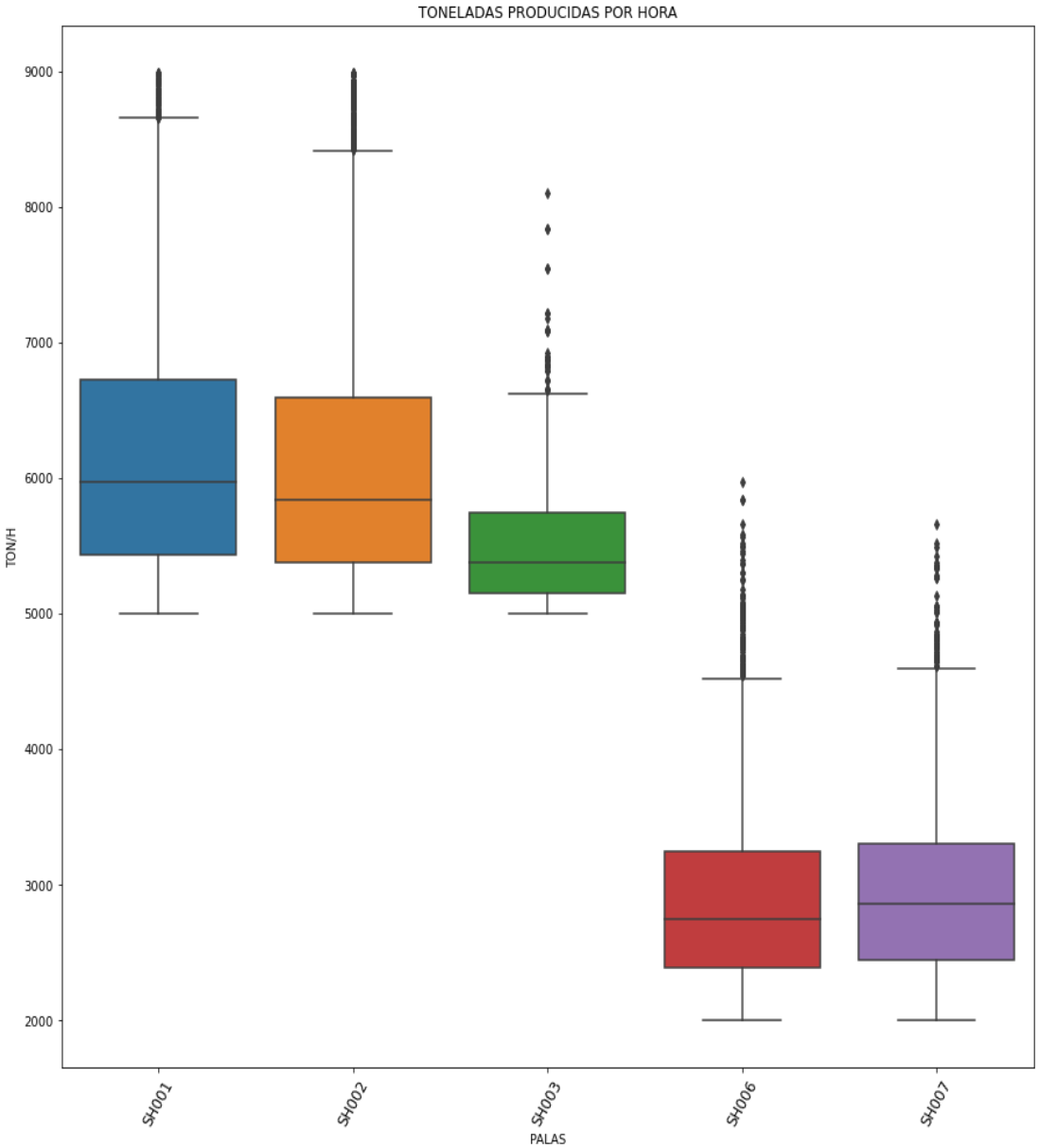
Nota: fuente elaboración propia.

El 75% de los tiempos hang promedio para las palas SH001 y SH002 están por debajo de 2.1 minutos; para la SH003 son menores de 2 minutos; para las palas SH006 y SH007 son menores de 5.8 minutos.

Para mantener una homogénea eliminaremos los outliers de material movido por hora que equivalen al 0.87 % no afectando la cantidad de datos, pero si ayudando a tener un modelamiento más coherente.

Figura 43

Diagrama de caja del tonelaje producido por hora sin considerar los outliers.



Nota: fuente elaboración propia.

4.5 Preparación de Dataset a entrenar

Antes de modelar debemos comenzar por convertir las variables categóricas en numéricas con ayuda del diccionario que se diseñó como se muestra en la Tabla 4.

Tabla 4

Diccionario de variables categóricas.

Variables	Valor	Código de la variable
Turno	Día	1
	Noche	2
Secondarymachineclassname	CAT 7495	2
	CAT 6060BH	3
Secondarymachinename	SH001	3
	SH002	4
	SH003	5
	SH006	6
	SH007	7
Primarymachineclassname	CAT 794AC CMD	2
Material	WC	1
	WA	2
	SSA	3
	SSB	4
	WV	5
	WO	6
	HYB	7
Tipo de material	Desmonte	1
	Mineral	2
Fase de Origen	F01N	1
	T05S	2
	F01S	3
	F02N	4
	Adhoc	5
	B01A	6
	T02D	7
	T02N	8
	S01N	9
	B01O	10
	T04N	11
	T06S	12
	F03S	13
	S01S	14
	F03B	15
Fase destino	unknown	0
	RVINPIT, RVINPIT01, RVINPIT_3825, RVINPIT_Garza, RVINPIT_3795, RVINPIT_3780, RVINPIT_3765, RVINPIT_3900, RVINPIT_3750, RVINPIT_3855, RVIMPIT_SOBRECARGAS, RVINPIT_ORE, RVINPIT_3600, RVINPIT_3784, RVINPIT_3784_1, RVINPIT_3784_2, RVINPIT_3585_F1S,	1

	RVINPIT_3784_3, rvinpit_3784_3, RVINPIT_3870, RVINPIT_3870_1, RVINPIT_3555, INPIT_3790, INPIT_3790_1 y RVINPIT_3795_1	
	ASANA_3465, ASANA_3465_PISO, ASANA_3465_1, ASANA_3470, ASANA_3645, ASANA_3480, ASANA_3465_03, ASANA_3450, ASANA_3455, ASANA_3455_1, ASANA_3455_2, ASANA_3455_4, ASANA_3435, ASANA_3435_1, ASANA_3435_2, ASANA_3455_3, ASANASUR_3525, ASANANSUR_3525_1, ASANASUR_3525_1 y ASANA_3465_2	2
	Tramo 05_3590, Tramo 05_3590_01, RVINPIT_Tramo5, TRAMO_05_3640, RVINPIT_Tramo5_1 y TR_05_BC_3610_50	3
	STKASAN_3480_HG, STKASAN_3480_2, STKASAN_3465_HG, STKASAN_3480_1, SkAsan_3465_LG, STKASAN_3465_LG, STKASAN_3500_HG, STKASAN_3465_LG_01, STKASAN_3500_LG, STKASAN_3500_HG1, STKASAN_3500_LG1, STKASAN_3500_HG2, STKASAN_3485_LG, STKASAN_3485_LG2, STKASAN_3525_LG, STKASAN_3525_LG_1, STKASAN_3510_HG, STKASAN_3520_HG, STKASAN_3505_LG, STKASAN_3505_LG1, STKASAN_3510_HG_2, STKASAN_3505_HG, STKASAN_3505_HG1, STKASAN_3490_HG, STKASAN_3505_HG2, STKASAN_3505_HG3, STKASAN_3505_HG4 y STKASAN_3475_HG1.	4
	ALTAR_3677, ALTAR_3677_Paddock, ALTAR_3650_02, ALTAR_3630, ALTAR_3630_01, ALTARANITO_3675, ALTARANITO_3675_02, ALTAR_3630_02, ALTAR_3620, ALTAR_3620_01, ALTAR_3660, ALTAR_3600_01, ALTAR_3660_01, ALTAR_3600, ALTAR_3600_02, ALTAR_3580, ALTAR_3580_01, ALTAR_BC_3630_3580, ALTAR_ZAR_3675, ALTAR_3560, ALTAR_3670_02, ALTAR_3670, ALTAR_3670_01, ALTAR_3560_1, ALTAR_3660_02, ALTAR_3675, ALTAR_3515_1, ALTAR_3515_2, ALTARANITO_3675_04, ALTAR_3675_02, ALTAR_3675_01, ALTAR_3515_3, ALTAR_3545, ALTAR_3545_1, ALTAR_3545_3, ALTAR_3555_1, ALTAR_3555, ALTAR_3555_3, ALTARRAMP_3545, ALTARRAMP_3545_1, ALTAR_3570, ALTAR_3570_1 y ALTAR_3570_2.	5
	T2D_3625, TRAMO_2D_3700_1, TRAMO_2D_3700, TRAMO2D_BC_3790, TR_2D_BC_01, TR_2D_BC, TR_2D_BC_3795 y TR_2D_BC_3795_01.	6
	TRAMO_07_3610, TRAMO_07_3570, RVINPIT_Tramo7, TR_07_BC_3610, INPIT_TRO7_3613, INPIT_TR07_3613_1, INPITTR07_3613_1 y STK_TR7_3613_HG	7
	RVINPIT_Tramo4, TRAMO_04_3940 y TR_04_BC_3465	8

	BotaderoSur_3540, BotadSur_3540_01, BotSur_3540, BotSur_3540_01, BOTSUR_3540_01, BOTSUR_3540, BOTSUR_3525, BOTSUR_3525_01, BOTSUR_3525_02, SUR_ZAR_3525, BOTINPIT_3613.	9
	Chancadora_entrada, Chancadora1 y Chancadora2.	10
	RVEXPIT	11
	STKSUR_3525_LG, STKSUR_3525_LG1, STKSUR_3525_LG2 y STKSUR_3525_LG3.	12
	STKRAMPA_3530_LG, STKRAMPA_3530_LG1, STKRAMPA_3530_LG2 y STKRAMPA_3505_LG1	13
	TR_06_BC_3575	14
	STK_TR7_3613	15
	STK_3490_HG y STK_3490_HG_1	16

Nota: fuente elaboración propia.

Figura 44

Código de parametrización de las variables categóricas a numéricas.

```
In [112]: resultados['COD_TURNO'] = resultados['TURNO'].map( {'DÍA': 1, 'NOCHE': 2} ).astype(int)
resultados['COD_FLOTA_SH'] = resultados['SECONDARYMACHINECLASSNAME'].map( {'CAT 7495': 2,
                                                                           'CAT 6060BH': 3} ).astype(int)
resultados['COD_SH'] = resultados['SECONDARYMACHINENAME'].map( {'SH001': 3,'SH002': 4,
                                                                'SH003': 5,'SH006': 6,'SH007': 7} ).astype(int)
#resultados['COD_FLOTA_HT'] = resultados['PRIMARYMACHINECLASSNAME'].map( {'CAT 794AC CMD': 2} ).astype(int)
resultados['COD_MATERIAL'] = resultados['MATERIAL'].map( {'WC': 1, 'WA': 2,'SSA': 3,'SSB': 4, 'WV': 5,
                                                          'WO': 6,'HYB': 7} ).astype(int)
resultados['COD_TIPO_MATERIAL'] = resultados['TIPO_MATERIAL'].map( {'Desmonte': 1, 'Mineral': 2} ).astype(int)
resultados['COD_FASE_ORIG'] = resultados['FASE_ORIGEN'].map( {'F01N': 1, 'T05S': 2,'F01S': 3,'F02N': 4, 'Adhoc': 5,
                                                              'B01A': 6,'T02D': 7,
                                                            'T02N': 8, 'S01N': 9,'B01O': 10,'T04N': 11, 'T06S': 12,
                                                              'F03S': 13,'S01S': 14,'F03B': 15 } ).astype(int)
resultados['COD_DESTINO'] = resultados['FASE_DESTINO'].map( {'RVINPIT':1, 'unknown':0, 'ASANA_3465':2, 'Tramo 05_3590':3,
       'ASANA_3465_PISO':2, 'Tramo 05_3590_01':3, 'ASANA_3465_1':2,
       'STKASAN_3480_HG':4, 'ALTAR_3677':5, 'RVINPIT_Tramo1':1, 'T2D_3625':6,
       'ASANA_3470':2, 'RVINPIT01':1, 'RVINPIT_Tramo5':3, 'TRAMO_07_3610':7,
       'STKASAN_3480_2':4, 'TRAMO_07_3570':7, 'ALTAR_3677_Paddock':5,
       'RVINPIT_Tramo4':8, 'STKASAN_3465_HG':4, 'TRAMO_2D_3700_1':6,
       'ALTAR_3650_02':5, 'ALTAR_3630':5, 'TRAMO_05_3640':3, 'ALTAR_3630_01':5,
       'RVINPIT_Tramo5_1':3, 'STKASAN_3480_1':4, 'BotaderoSur_3540':9,
       'TRAMO_2D_3700':6, 'ASANA_3645':2, 'BotadSur_3540_01':9,
       'ALTARANITO_3675':5, 'ALTARANITO_3675_02':5, 'BotSur_3540':9,
       'BotSur_3540_01':9, 'BOTSUR_3540_01':9, 'BOTSUR_3540':9, 'TRAMO_04_3940':8,
       'ALTAR_3630_02':5, 'RVINPIT_3825':1, 'SkAsan_3465_LG':4,
       'STKASAN_3465_LG':4, 'STKASAN_3500_HG':4, 'STKASAN_3465_LG_01':4,
       'STKASAN_3500_LG':4, 'STKASAN_3500_HG1':4, 'STKASAN_3500_LG1':4,
       'ASANA_3480':2, 'STKASAN_3500_HG2':4, 'ALTAR_3620':5, 'RVINPIT_Garza':1,
       'STKASAN_3485_LG':4, 'ALTAR_3620_01':5, 'Chancadora_entrada':10,
       'RVINPIT_3795':1, 'Chancadora1':10, 'RVINPIT_3780':1, 'STKASAN_3485_LG2':4,
       'RVEXPIT':11, 'ALTAR_3660':5, 'ALTAR_3600_01':5, 'ALTAR_3660_01':5,
       'STKASAN_3525_LG':4, 'STKASAN_3525_LG_1':4, 'STKSUR_3525_LG':12,
       'STKSUR_3525_LG1':12, 'ALTAR_3600':5, 'STKASAN_3510_HG':4,
       'STKASAN_3520_HG':4, 'ALTAR_3600_02':5, 'RVINPIT_Tramo7':7,
       'STKASAN_3505_LG':4, 'STKASAN_3505_LG1':4, 'BOTSUR_3525':9,
       'BOTSUR_3525_01':9, 'RVINPIT_3765':1, 'RVINPIT_3900':1,
       'STKASAN_3510_HG_2':4, 'ALTAR_3580':5, 'ALTAR_3580_01':5,
       'ASANA_3465_03':2, 'ALTAR_BC_3630_3580':5, 'ALTAR_ZAR_3675':5,
       'RVINPIT_3750':1, 'STKSUR_3525_LG2':12, 'ALTAR_3560':5, 'RVINPIT_3855':1,
       'Chancadora2':10, 'ALTAR_3670_02':5, 'TRAMO2D_BC_3790':6,
       'STKASAN_3505_HG':4, 'ALTAR_3670':5, 'STKASAN_3505_HG1':4, 'TR_2D_BC_01':6,
       'TR_2D_BC':6, 'TR_2D_BC_3795':6, 'TR_2D_BC_3795_01':6, 'TR_04_BC_3465':8,
       'BOTSUR_3525_02':9, 'ALTAR_3670_01':5, 'ALTAR_3560_1':5,
```

Nota: fuente elaboración propia.

A continuación, seleccionamos generamos el DataSet final que entrará en el modelamiento, el cual solo tendrá variables numéricas (eliminando las categóricas y dejando las que fueron parametrizadas).

Generamos un ordenamiento interno de las columnas dejando la variable objetivo “Productividad” al final del dataset, con lo cual obtendríamos las siguientes columnas: Año, Mes, Fecha, Turno, Guardia, hora, Secondarymachineclassname, Secondarymachinename, material, tipo de material, fase origen, fase destino, tonelaje, tiempo queue, tiempo spot, tiempo de carguío, tiempo de viaje cargado, tiempo queue en el destino, tiempo spot en el destino, tiempo de descarga, tiempo de viaje vacío, factor de carga, efh cargado, efh vacío, disponibilidad de HTs (%), tiempo hang, disponibilidad SH (%), Cod_turno, Cod_flota_SH, Cod_SH, Cod_material, Cod_tipo_material, Cod_fase_origen, Cod_destino, Productividad.

Figura 45

Dataset final para ser ingresado al modelamiento.

	MES	GUARDIA	HORA	TIEMPO QUEUE	TIEMPO SPOT	TIEMPO DE CARGUÍO	TIEMPO VIAJE CARGADO	TIEMPO QUEUE DESTINO	TIEMPO SPOT DESTINO	TIEMPO DE DESCARGA	...	TIEMPO HANG	DISPONIBILIDAD SH (%)	COD_TURN
7	6	2	9	184.285714	44.428571	140.285714	363.142857	33.142857	35.142857	64.142857	...	131.555556	100.0	
8	6	2	9	258.000000	46.000000	122.000000	366.000000	33.000000	33.000000	57.000000	...	131.555556	100.0	
9	6	2	10	214.583333	56.666667	133.416667	374.000000	8.333333	41.750000	60.666667	...	101.181818	100.0	
23	6	2	14	318.500000	45.000000	174.000000	400.833333	0.000000	43.333333	64.333333	...	103.000000	100.0	
45	6	3	0	40.142857	44.285714	162.428571	383.142857	6.857143	37.428571	62.142857	...	137.900000	100.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
80888	8	1	15	163.000000	45.000000	83.000000	292.000000	0.000000	42.000000	40.000000	...	74.500000	100.0	
80893	8	1	16	137.000000	38.166667	151.166667	485.500000	56.333333	30.333333	31.333333	...	200.142857	100.0	
80894	8	1	16	40.000000	35.000000	147.000000	428.000000	0.000000	40.000000	38.000000	...	200.142857	100.0	
80897	8	1	16	263.769231	39.538462	76.538462	457.538462	83.692308	36.769231	37.384615	...	73.764706	100.0	
80898	8	1	16	314.000000	39.000000	76.000000	359.000000	0.000000	0.000000	0.000000	...	73.764706	100.0	

27923 rows × 25 columns

Nota: fuente elaboración propia.

4.6 Evaluación y selección del mejor modelo de Machine Learning

Actualmente PyCaret, que es una librería de aprendizaje automático de código abierto en Python automatiza los flujos de trabajo de aprendizaje automático. Siendo una herramienta integral de gestión de modelos y aprendizaje automático que acelera exponencialmente el ciclo de experimentación y lo hace más productivo, permitiendo obtener los mejores modelos de machine learning que se desean determinar.

Basado en esta librería se hizo el análisis del mejor modelo en Jupyter Notebook, obtenido el Ranking de los mejores modelos predictivos de productividad de palas de nuestros datos.

Figura 46

Código de programación para determinar un ranking de modelos predictivos de productividad de palas.

```
from pycaret.regression import *
rg=setup(data=resultados,target='PRODUCTIVIDAD')
compare_models()
```

	Description	Value
0	Session id	5430
1	Target	PRODUCTIVIDAD
2	Target type	Regression
3	Original data shape	(28170, 31)
4	Transformed data shape	(28170, 53)
5	Transformed train set shape	(19719, 53)
6	Transformed test set shape	(8451, 53)
7	Ordinal features	3
8	Numeric features	20
9	Categorical features	10
10	Preprocess	True
11	Imputation type	simple
12	Numeric imputation	mean
13	Categorical imputation	mode
14	Maximum one-hot encoding	25
15	Encoding method	None
16	Fold Generator	KFold
17	Fold Number	10
18	CPU Jobs	-1
19	Use GPU	False
20	Log Experiment	False
21	Experiment Name	reg-default-name
22	USI	c2ac

Nota: fuente elaboración propia.

Los resultados obtenidos a través de PyCaret se ven reflejados en la Tabla 5.

Tabla 5

Ranking de modelos predictivos de productividad de palas con Machine Learning para los datos analizados.

Modelo	MAE	MSE	RMSE	R2 (%)
Random Forest Regressor	65.58	9285.04	96.36	99.71
Gradient Boosting Regressor	261.66	146795.09	382.99	95.59
Decision Tree Regressor	223.69	208518.54	455.69	93.74
Linear Regression	445.57	368561.81	606.75	88.94
Ridge Regression	445.55	368549.82	606.74	88.94
Bayesian Ridge	445.17	368507.75	606.70	88.94
Lasso Least Angle Regression	445.22	369011.26	607.11	88.93
Lasso Regression	445.22	369011.31	607.11	88.93
AdaBoost Regressor	505.02	389041.24	623.47	88.31
Elastic Net	451.71	401576.60	633.36	87.95
Huber Regressor	434.77	535946.42	731.37	83.92
K Neighbors Regressor	453.88	548404.85	739.95	83.54
Orthogonal Matching Pursuit	488.36	594954.44	770.86	82.15
Passive Aggressive Regressor	572.85	598053.91	767.79	82.09
Dummy Regressor	1593.32	3330630.48	1824.88	-0.03

Nota: fuente elaboración propia, producto de los resultados obtenidos en Python.

Observamos que el modelo de mayor predición para nuestro DataSet es el modelo de machine learning “Random Forest Regressor” con una precisión de 99.71% para el modelo de entrenamiento y el de menor el “Dummy Regressor” con – 0.03%.

4.7 Cálculo de los mejores hiperparámetros del modelo

Todo modelo de Machine learning se alimenta condiciones o hiperparámetros para analizar y entrenar la información, Random Forest Regressor, que es un metaestimador, agrupa los árboles de decisión de clasificación en diferentes subclases, para utilizar el promedio que permita mejorar la precisión predictiva y monitoree el error de sobreajuste, por ende es importante el tamaño de la subclase con el hiperparámetro max_samples; por que de no controlarlo, se usará el conjunto de datos para diseñar cada árbol de decisión.

Los hiperparámetros que debemos tener en cuenta, de acuerdo a Scikit-learn son:

- n_estimators: es el número de árboles que se desea implementar en el modelamiento.
- Criterion: es el error de un agrupamiento, el cual muestra la calidad como será tratado. Estos pueden ser "squared_error" si el error es cuadrático medio, que es igual a la reducción de la varianza para la selección de características, "friedman_mse", si el error es cuadrático medio con la puntuación de Friedman permite el potencial splits, "absolute_error" si es absoluto medio, a través de la mediana para los nodos terminales, y "poisson" para la desviación de Poisson. El entrenamiento con "absolute_error" es más engorroso de procesar que el "squared_error".
- max_ Depth: permite determinar la profundidad máxima del árbol. Si se selecciona la opción Ninguno, los nodos alcanzan todas las hojas hasta sean puras o que contengan menos de min_samples_split simples.
- min_samples_split: Indica el número mínimo de muestras que se requieren para dividir un nodo interno. Si se elije int, se considera min_samples_splitcomo como número mínimo; en caso de elegir flotante, el min_samples_splites será una fracción y es el mínimo de muestras para cada división.
- min_samples_leaf: Es el mínimo de muestras que se requieren para estar en un nodo hoja. Sirve para ayudar al efecto de suavizar el modelo, más aún en regresión.
- max_features: Es la cantidad de características que debemos evaluar al buscar el mejor árbol.
- max_leaf_nodes: ayuda a la reducción relativa de la impureza. Si por defecto dejamos en Ninguno, se considerará un número ilimitado de nodos en las hojas de los árboles.

- random_state: manipula la aleatoriedad del arranque de las muestras en los árboles como el muestreo de las características cuando busquemos el mejor árbol de decisión.

La librería Sklearn nos ayuda a calcular los mejores hiperparámetros del modelo a través de la herramienta “RandomizedSearchCV”, que implementa un método de "ajuste" y "puntuación" en cada nodo.

A través de la implementación de los hiperparámetros “score_samples”, “predict”, “predict_proba”, “decision_function”, “transform” y “inverse_transform” se utilizan métodos que optimizan la búsqueda de validación cruzada sobre la configuración determinada.

No se utilizan todos los valores de los parámetros, sino que se fija las configuraciones para determinar los valores que adquirán los parámetros de las distribuciones. La variable que determina las configuraciones de parámetros es el n_iter.

Cuando todos los parámetros se presenten en una lista, se realizará un muestreo sin reemplazo. Se debe utilizar distribuciones continuas para parámetros continuos.

Para calcularlos se determinó los siguientes rangos:

- test_size = 0.25: es decir el 25% de los datos integrarán el grupo de testeo y el modelo será entrenado con el 75% restante.
- Random_state = 38
- n_estimators = [200, 400, 600, 800, 1000, 1200, 1400, 1600, 1800, 2000]
- max_features = [‘auto’,’sqrt’]
- max_depth = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100, None]
- min_samples_leaf = [1, 2, 4]
- Bootstrap = [True, False]

Se programó en Jupyter definiendo los parámetros como se mencionó.

Figura 47

Código de programación para aplicar la validación cruzada y determinar los mejores hiperparámetros para nuestro modelo de Random Forest Regressor.

```
from sklearn.model_selection import train_test_split
train_features, test_features, train_labels, test_labels = train_test_split(X, Y, test_size = 0.25, random_state = 38)
```

```
from sklearn.ensemble import RandomForestRegressor
rf = RandomForestRegressor(random_state = 38)
from pprint import pprint
# Mire los parámetros usados ••por nuestro bosque actual
print('Parámetros actualmente en uso:\n')
print(rf.get_params())
```

```
Parámetros actualmente en uso:

{'bootstrap': True, 'ccp_alpha': 0.0, 'criterion': 'squared_error', 'max_depth': None, 'max_features': 'auto', 'max_leaf_node
s': None, 'max_samples': None, 'min_impurity_decrease': 0.0, 'min_samples_leaf': 1, 'min_samples_split': 2, 'min_weight_fractio
n_leaf': 0.0, 'n_estimators': 100, 'n_jobs': None, 'oob_score': False, 'random_state': 38, 'verbose': 0, 'warm_start': False}
```

```
from sklearn.model_selection import RandomizedSearchCV
import numpy as np
# Number of trees in random forest
n_estimators = [int(x) for x in np.linspace(start = 200, stop = 2000, num = 10)]
# Number of features to consider at every split
max_features = ['auto', 'sqrt']
# Maximum number of levels in tree
max_depth = [int(x) for x in np.linspace(10, 100, num = 10)]
max_depth.append(None)
# Minimum number of samples required to split a node
min_samples_split = [2, 5, 10]
# Minimum number of samples required at each leaf node
min_samples_leaf = [1, 2, 4]
# Method of selecting samples for training each tree
bootstrap = [True, False]

# Create the random grid
random_grid = {'n_estimators': n_estimators,
               'max_features': max_features,
               'max_depth': max_depth,
               'min_samples_split': min_samples_split,
               'min_samples_leaf': min_samples_leaf,
               'bootstrap': bootstrap}
print(random_grid)
```

```
{'n_estimators': [200, 400, 600, 800, 1000, 1200, 1400, 1600, 1800, 2000], 'max_features': ['auto', 'sqrt'], 'max_depth': [10,
20, 30, 40, 50, 60, 70, 80, 90, 100, None], 'min_samples_split': [2, 5, 10], 'min_samples_leaf': [1, 2, 4], 'bootstrap': [True,
False]}
```

```
# Use the random grid to search for best hyperparameters
# First create the base model to tune
rf = RandomForestRegressor()
# Random search of parameters, using 3 fold cross validation,
# search across 100 different combinations, and use all available cores
rf_random = RandomizedSearchCV(estimator=rf, param_distributions=random_grid,
                               n_iter = 100, scoring='neg_mean_absolute_error',
                               cv = 3, verbose=2, random_state=38, n_jobs=-1)

# Fit the random search model
rf_random.fit(train_features, train_labels)
```

```
Fitting 3 folds for each of 100 candidates, totalling 300 fits
```

Nota: fuente elaboración propia.

Luego de procesar la validación cruzada, logramos obtener dichos parámetros cuyos valores se ven reflejados en la Tabla 6.

Tabla 6

Mejores hiperparámetros para el modelo de Random Forest Regressor.

Hiperparámetro	Mejor valor
Test_size	0.25
Random_state	38
n_estimators	2000
min_samples_split	4
min_samples_leaf	2
max_features	'sqrt'
max_depth	100

Nota: fuente elaboración propia.

4.8 Aplicación del Modelo Random Forest Regressor

Una vez obtenido los mejores parámetros, estos fueron ingresados en el modelo de Random Forest Regressor.

Figura 48

Código del modelamiento de Random Forest Regressor.

```
X = resultados3.drop(['PRODUCTIVIDAD'],axis=1)
Y = resultados3['PRODUCTIVIDAD']
#visualizamos los datos
X.head()
```

	MES	GUARDIA	HORA	TIEMPO QUEUE	TIEMPO SPOT	TIEMPO DE CARGUÍO	TIEMPO VIAJE CARGADO	TIEMPO QUEUE DESTINO	TIEMPO SPOT DESTINO	TIEMPO DE DESCARGA	...	DISPONIBILIDAD HT (%)	TIEMPO HANG	DISPONIBILIDA SH (
7	6	2	9	184.285714	44.428571	140.285714	363.142857	33.142857	35.142857	64.142857	...	100.0	131.555556	10(
8	6	2	9	258.000000	46.000000	122.000000	366.000000	33.000000	33.000000	57.000000	...	100.0	131.555556	10(
9	6	2	10	214.583333	56.666667	133.416667	374.000000	8.333333	41.750000	60.666667	...	100.0	101.181818	10(
23	6	2	14	318.500000	45.000000	174.000000	400.833333	0.000000	43.333333	64.333333	...	100.0	103.000000	10(
45	6	3	0	40.142857	44.285714	162.428571	383.142857	6.857143	37.428571	62.142857	...	100.0	137.900000	10(

5 rows × 24 columns

```
from sklearn.model_selection import train_test_split
train_features, test_features, train_labels, test_labels = train_test_split(X, Y, test_size = 0.25, random_state = 38)
```

```
#training and testing the random forest
from sklearn.ensemble import RandomForestRegressor
rf_reg2 = RandomForestRegressor(n_estimators= 2000,
 min_samples_split=4,
 min_samples_leaf= 2,
 max_features= 'sqrt',
 max_depth= 100,
 bootstrap= False)
regressor = rf_reg2.fit(train_features,train_labels)
Y_pred = regressor.predict(test_features)
```

Nota: fuente elaboración propia.

Una vez obtenido los mejores parámetros, estos fueron ingresados en el modelo de Random Forest Regressor.

Obteniéndose los siguientes resultados:

- 99.7% de precisión en el modelo de entrenamiento.
- 91.1% de precisión en el modelo de testeo.
- 65.58 de error absoluto medio.
- 9285.04 de error medio cuadrado.
- 96.36 de error cuadrático medio.

4.9 Pruebas

Para validar la hipótesis se verifica lo comentado en la Tabla 5 y se comparan los datos obtenidos de los primeros 10 valores arrojados por el modelo y sus respectivos registros del dataset de testeo.; así como también con la data de entrenamiento.

Tabla 7

Comparación de los valores predecidos y los registrados en el testeo.

Y	1	2	3	4	5	6	7	8	9	10
Y_predict	5524.2	7276.4	3620.8	2872.8	6229.5	5872.1	7259.1	5560.1	2958.6	5856.0
Y_test	5434.1	8584.3	2209.0	2583.0	6061.1	5626.7	8441.4	5062.1	3164.0	6503.4

Nota: fuente elaboración propia.

Tabla 8

Comparación de los valores predecidos y los registrados en el entrenamiento.

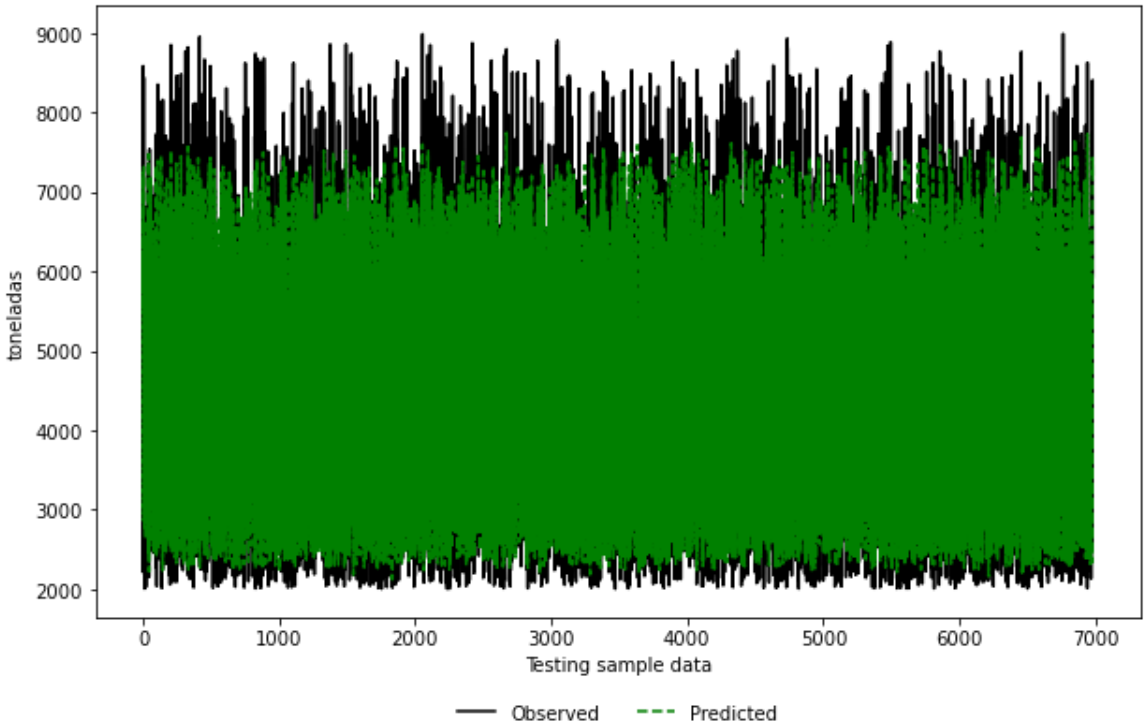
Y	1	2	3	4	5	6	7	8	9	10
Y_predict	5966.2	6352.6	5488.3	2629.5	6161.9	5751.8	5246.4	8178.8	5204.2	2357.2
Y_train	5951.9	6480.5	5470.1	2587.5	6265.8	5736.8	5189.0	8319.8	5175.7	2051.0

Nota: fuente elaboración propia.

Se graficó la productividad testeada vs la predicha, así como el error de predicción y los valores residuales en el DataSet de testeo.

Figura 49

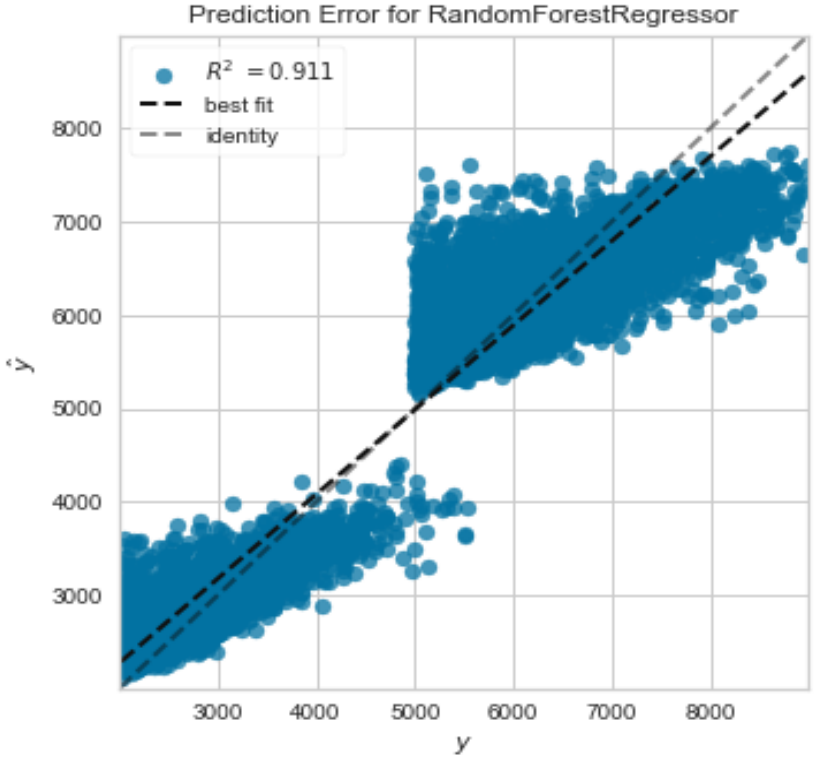
Productividad predicha vs asignada en el DataSet de testeo.



Nota: fuente elaboración propia.

Figura 50

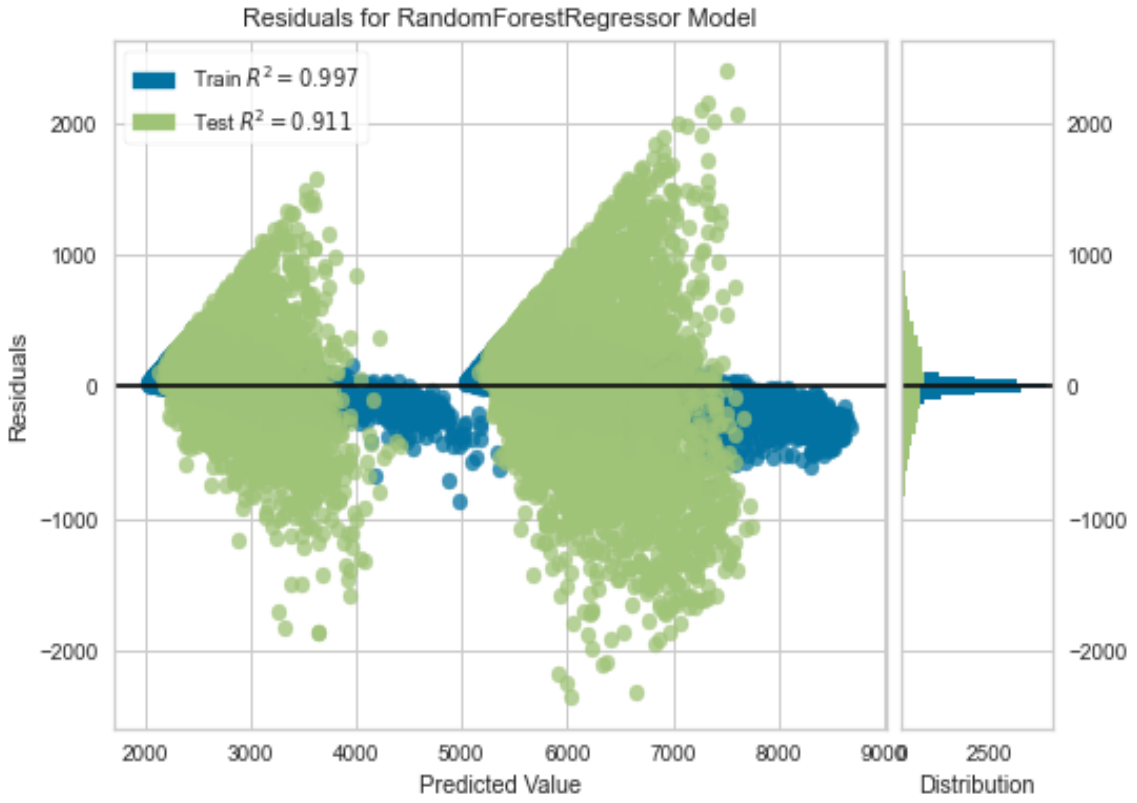
Error de predicción en el DataSet de testeo aplicando Random Forest Regressor.



Nota: fuente elaboración propia.

Figura 51

Valores residuales en el DataSet de testeo aplicando Random Forest Regressor.

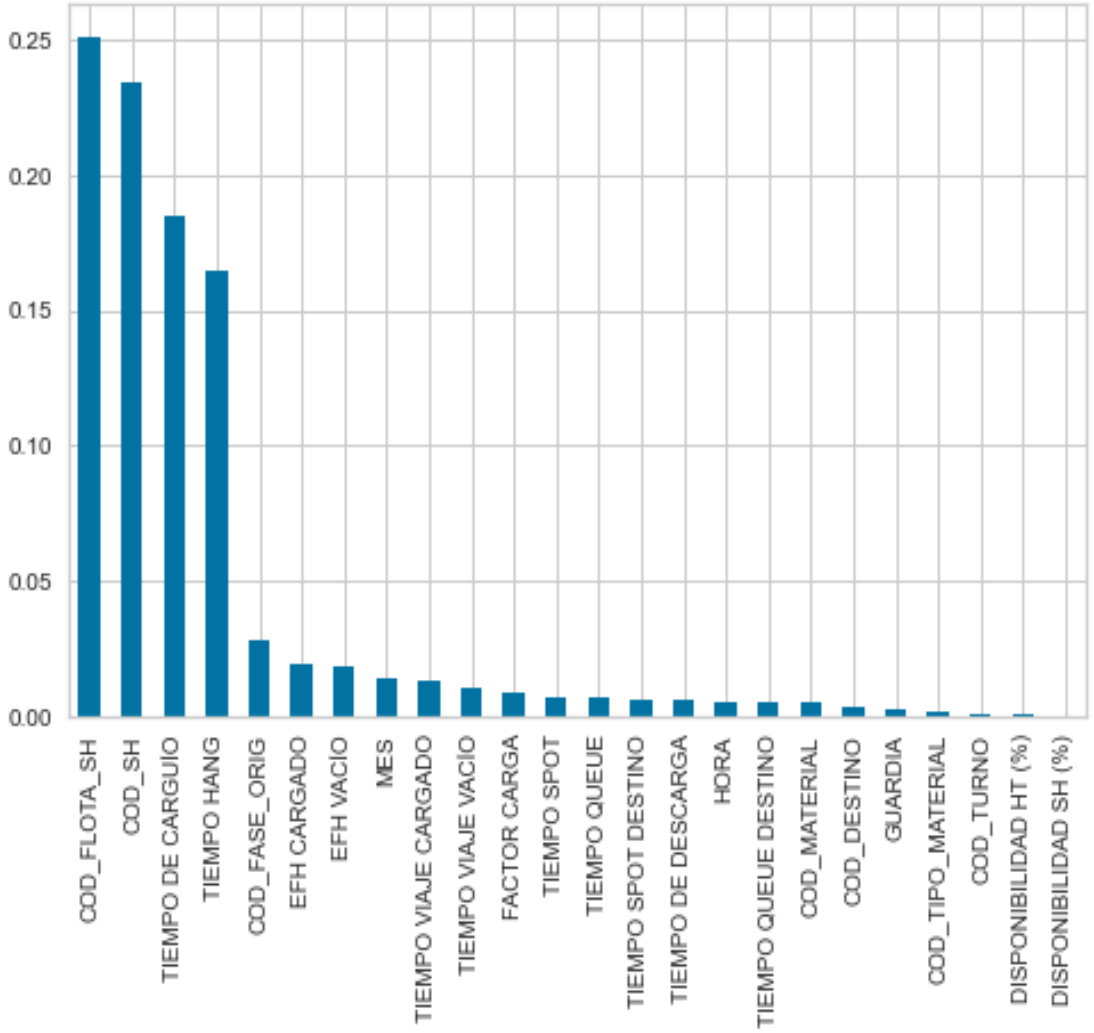


Nota: fuente elaboración propia.

Aplicando “feature_importances_” de la librería pandas se obtuvo el grado de importancia de cada una de las variables que conforman el modelo, siendo las principales: el Cod_flota_SH, el Cod_SH, el tiempo de carguío y el tiempo hang.

Figura 52

Importancia de las variables en el modelo Random Forest Rgressor.



Nota: fuente elaboración propia.

En conclusión, se rechaza la hipótesis nula (H_0) y se acepta la hipótesis alternativa (H_A), la cual indica que la construcción de un modelo predictivo de productividad de palas aplicando Random Forest Regressor predice mejor que otros modelos.

Conclusiones

Para el modelo de machine learning se utilizó todas las variables presentes en el DataSet, incluyendo las variables cualitativas, las cuales fueron categorizadas antes del entrenamiento.

De los modelos de machine learning el modelo que mejor predice la productividad de las palas es el “Random Forest Regressor” con una precisión del 99.7% para el entrenamiento y 91.1% para el DataSet de testeo; mientras que el peor modelo de machine learning es “Dummy Regressor” con - 0.03% de precisión.

Los modelos clásicos de productividad de palas considera solo las toneladas movidas, el tiempo de carguío y el tiempo hang determinado en una pala; mientras que el modelo de Random Forest Regressor construido toma en cuenta las variables mencionadas y agrega el tiempo queue, el tiempo spot, el tiempo de viaje cargado, el tiempo queue en el destino, el tiempo spot en el destino, el tiempo de descarga, el tiempo de viaje vacío, el factor de carga, el código de flota SH, el código de SH, el código de fase origen, el código de fase destino, el EFH cargado, el EFH vacío, el mes, la hora, el material, el tipo de material, la guardia, la disponibilidad de los camiones y de las palas.

El modelo clásico no permite cuantificar la importancia o influencia de las variables en el modelo; mientras que el modelo construido determina que la importancia para cada variable como se muestra en la Tabla 9 siguiente:

Tabla 9

Importancia de las variables en el modelo Random Forest Regressor.

VARIABLE	IMPORTANCIA (%)
Cod_Flota_SH	24.9259%
Cod_SH	23.2086%
Tiempo de carguío	19.3239%
Tiempo hang	16.0613%
Cod_fase_origen	2.7542%
EFH Cargado	1.9929%
EFH Vacío	1.8462%
Mes	1.3963%
Tiempo viaje cargado	1.3190%
Tiempo viaje vacío	1.0892%
Factor de carga	0.8684%
Tiempo spot	0.7272%
Tiempo queue	0.7172%
Tiempo spot en el destino	0.6224%
Tiempo de descarga	0.5976%
Hora	0.5431%
Tiempo queue destino	0.5123%
Cod_material	0.5084%
Cod_destino	0.3555%
Guardia	0.2771%
Cod_tipo_material	0.1874%
Cod_turno	0.1111%
Disponibilidad HT (%)	0.0408%
Disponibilidad SH (%)	0.0140%

Nota: fuente elaboración propia.

El modelo predictivo de productividad de palas con Random Forest Regressor de Machine learning optimiza y mejora la producción en comparación a otros modelos clásicos y de machine learning.

Recomendaciones

Se debe aprovechar la librería “pycaret” de Python para determinar un ranking de precisón de los modelos de machine learning aplicados al DataSet que se desea modelar.

Se recomienda hacer la limpieza de los outliers antes de generar el modelo de Random Forest Regressor, para obtener un mayor grado de precisión, se debe tener en cuenta el criterio de operación para seleccionar los datos correctos y no perder información importante, esta limpieza no debe exceder del 5% del total de los datos.

Antes de iniciar cada modelo de machine learning, se debe buscar los mejores hiperparámetros del modelo, que se desea desarrollar con ayuda del método de “Validación cruzada”, ya que permitirá realizar mejor predicción.

A mayor cantidad de datos en el DataSet, el modelo de predicción será mejor, si a pesar de esto, la predicción del modelo es baja, se debe eliminar las variables de menor importancia, ya que estas afectan la interacción de los árboles de decisión.

Para corroborar la predicción del modelo se debe hacer pruebas con la data de entrenamiento y también identificar el nivel de predicción de este modelo con los datos en el testeo, ya que, si la data de entrenamiento alcanza un alto grado de predicción, pero en el testeo es bajo, la causa principal es porque se genera overfitting, es decir, se acostumbró al entrenamiento, pero su grado de predicción realmente no es alto. Por ello se recomienda evaluar también la predicción del modelo con la data de testeo, la cual también debe ser buena por encima del 80%.

Para entrenar su modelo en Random Forest Regressor, se recomienda dividir el DataSet en 25% para testeo y 75% para entrenamiento.

Referencias bibliográficas

Alí Soofastaei. (2020). Data Analytics Applied to the Mining Industry. Australia.

Alpaydin E. (2010). “Introduction to Machine Learning” (2da Edición). Cambridge, Estados Unidos: Massachusetts Institute of Technology

Andreas C. Mueller, Sarah Guido (2017), Libro “Introduction to Machine Learning with Python”. EEUU

Breiman, L., J. H. Friedman, R. A. Olshen, and C. J. Stone. (1984). Classification and Regression Trees. https://www.crcpress.com/Classification-and-Regression-Trees/Breiman-Friedman-Stone-Olshen/p/book/9780412048418.

Breiman, Leo (1996). Bagging predictors 24 (2). Machine Learning. pp. 123-140

Breiman, L. (2001). Random Forest. En L. Breiman. Robert E. Schapire.

Chong Chong Qi. (2020). Big Data Management in the Mining Industry. http://ijmmm.ustb.edu.cn/en/article/doi/10.1007/s12613-019-1937-z

Cornejo Castro, Samuel Sebastián. (2017). *Optimización - simulación de carguío y acarreo en tajo Abierto utilizando NSGAII y programación lineal entera.* [Tesis de pregrado, Pontificia Universidad Católica del Perú]. Repositorio Institucional PUCP. https://repositorio.pucp.edu.pe/index/

Gil Matías (2012). La Industria Minera en búsqueda de la eficiencia con Big Data. https://www.americaeconomia.com/analisis-opinion/la-industria-minera-en-busqueda-de-la-eficiencia-con-big-data

Gonzales, Andrés (2014). ¿Qué es Machine Learning? http://cleverdata.io/quees-machine-learning-big-data/

Hultstrom, K. (2013). Image based wheel detection using random forest classification. Centre for Mathematical Sciences.

Instituto SAS (2016). Guía del usuario del software de análisis estadístico (SAS) versión 9.4. Cary, Carolina del Norte: SAS Institute, Inc

Joaquín Amat Rodrigo (2020). Libro “Random Forest con Python”.

Mauricio Quiquia, Gerardo William. (2015). *Mejoramiento continuo en la gestión del ciclo de acarreo de camiones en minería a tajo abierto en Antamina, Cerro Verde, Toquepala, Cuajone, Yanacocha, Alto Chicama, Las Bambas, Cerro Corona, Antapacay y Pucamarca.* [Tesis de Maestría, Universidad Nacional de Ingeniería]. Repositorio Institucional UNI. https://cybertesis.uni.edu.pe/

Max Kuhn, Kjell Johnson (2013). Libro “Applied Predictive Modeling”.

Prokopenko, J. (1989). Libro “La gestión de la productividad”.

Anexos

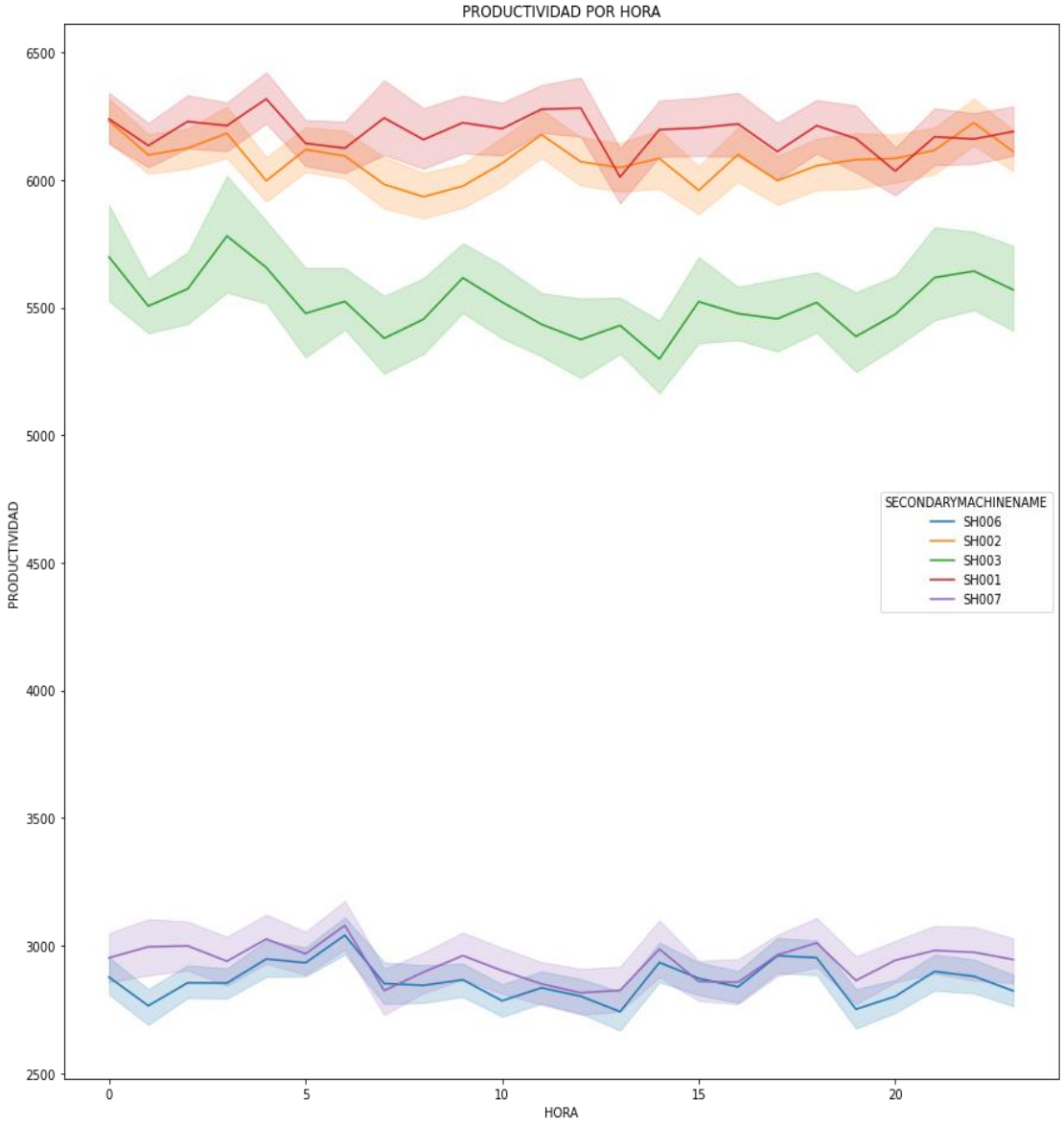
Lista de Anexos

Pág.

Anexo Nº 1. Evolución de la productividad de las palas por hora para cada flota.

Figura A1

Productividad (t/h) de las palas.

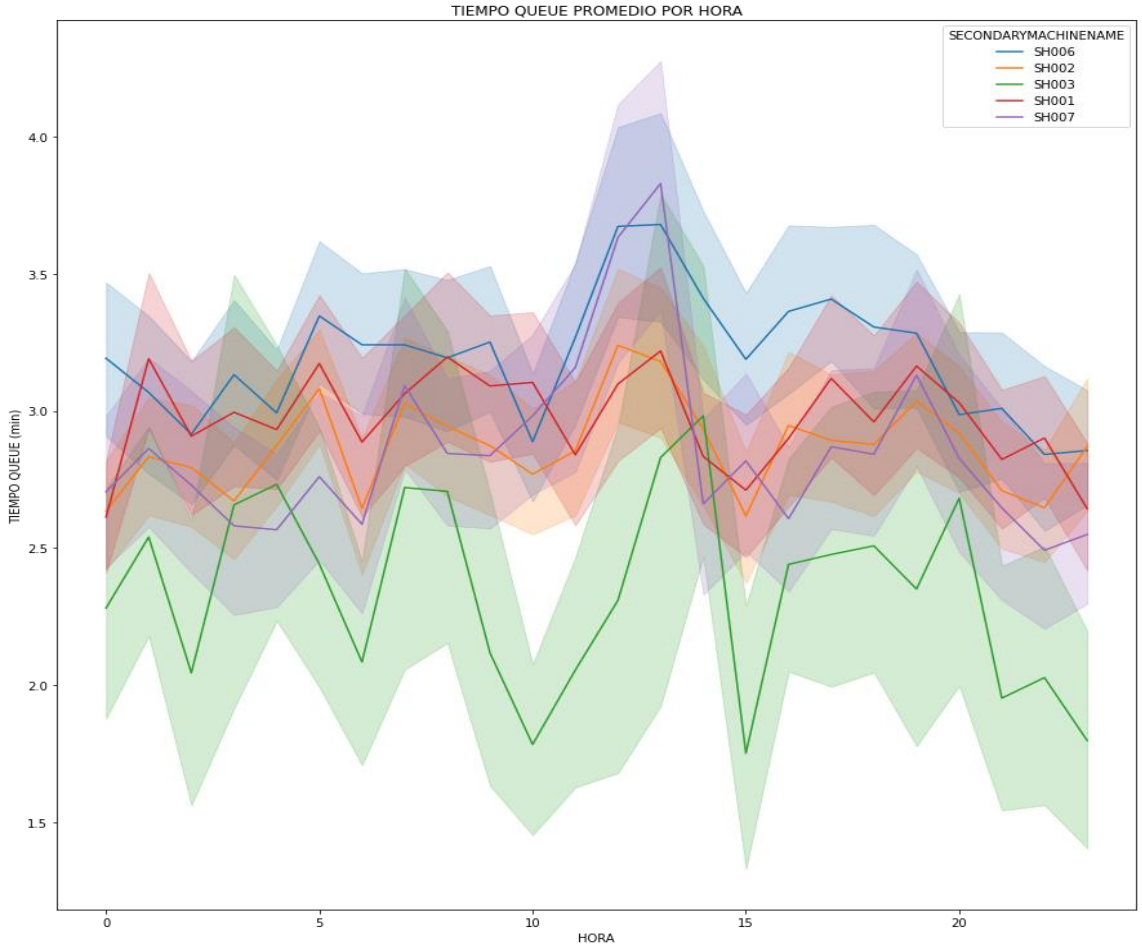


Nota: fuente elaboración propia.

Anexo Nº 2. Evolución del tiempo queue (min) en las palas por hora.

Figura A2

Tiempo queue (min) en las palas.



Nota: fuente elaboración propia.