



Instituto Tecnológico y de Estudios Superiores de Monterrey
Campus Monterrey

Predinálisis

Equipo 5

Leonardo Laureles Olmedo, PM y Científico de datos

Mariana Rincón Flores, Analista de datos y Científico de datos

Carlos Mateos Pérez, Desarrollador web y Científico de datos

Daniel Núñez López, CDO y Científico de datos

Juan Manuel Cantú, Ingeniero de datos y Científico de datos

Profesora Ma. Angelina Alarcón Romero

Profesora Rubí Isela Gutiérrez López

1. Descripción del problema específico

1.1 Objetivo del Negocio: objetivos del negocio, beneficios esperados, metas y criterios de éxito.

Objetivo

- Armado de propuesta para determinación de un predictivo de Defecto en base a indicadores de comportamiento de cada equipo.

Beneficio de Negocio:

- Reducir el volumen de materiales con defectos causado por comportamiento del equipo
- Aumentar la productividad de la línea reduciendo los desvíos o reprocesos de materiales

1.2 Evaluación de situación actual

El reto consiste en diseñar una propuesta para la disminución de caídas y retención de materiales recubiertos para la empresa Ternium, que buscan transformar y disminuir la compra de material de acero con defectos para diferentes sectores como lo es la construcción (optimiza los costos de proyectos de construcción y brinda alternativas estéticas aptas para una amplia gama de diseños) automotriz (se buscan aceros avanzados con menor peso y mayor resistencia), linea blanca (aceros pesados para manufactura de piezas de menor peso y mayor resistencia), agroindustria (evita que la tierra sobre la que trabajan los equipos se comprima y se vea dañada, a largo plazo para producir alimentos), transporte (un 15% del acero fabricado en el mundo se destina a este sector) [5].

Se busca dar una solución a este problema ya que en promedio en la fabricación de acero, se tiene registrado que hay un gasto excesivo de material a la hora de hacer los cortes ya que no existe un procedimiento especial para los cortes, esto a nivel general. Existen diversas soluciones, que van desde la metodología del proyecto, como lo es la metodología Scrum, que busca regular un conjunto de tareas con el objetivo principal de trabajar de manera colaborativa, hasta la implementación de nuevas tecnologías como la informática, específicamente la rama de ciencia de datos, que es una herramienta sumamente llamativa para este tipo de empresas que buscan la optimización de sus recursos usando la inteligencia artificial como solución, por ejemplo, en la industria de la fabricación, la empresa no solo incrementa su productividad económica, sino también ayuda al negocio de mejora de sus productos optimizando los procesos y concretando nuevos clientes teniendo en cuenta que la empresa Ternium no se enfoca en un solo sector como la construcción.

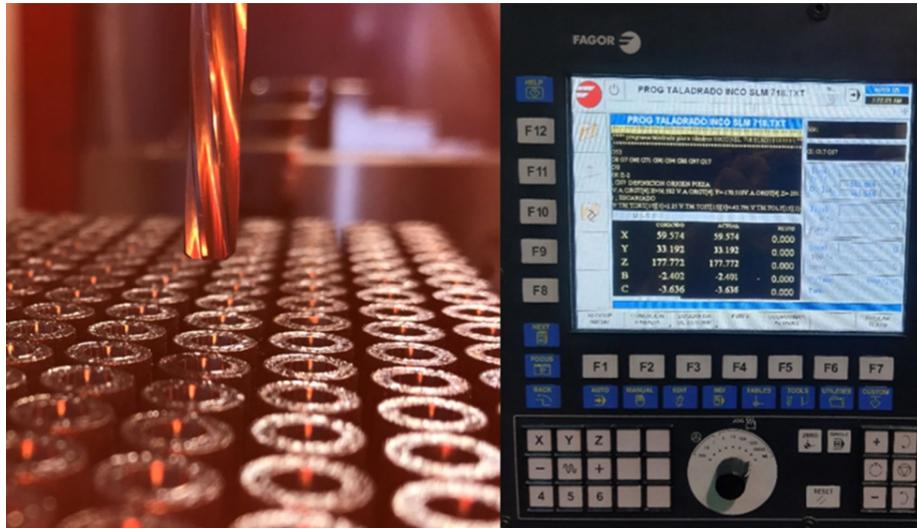


Figura 1. Ejemplo de cómo la programación ayuda en los procesos de corte de materiales.

1.3 Solución propuesta: Enfoque a utilizar para resolver el problema.

Para poder resolver esta problemática, se buscará utilizar algún algoritmo de Machine Learning con ayuda de la librería scikit-learn y el lenguaje de programación Python para poder determinar la causa de los defectos con mayor impacto, o en dado caso, saber qué factores son los que hacen que se obtenga el material final con algún defecto, así mismo, se brindarán recomendaciones a Ternium sobre cómo minimizar dichos problemas.

1.4 Hipótesis

Al utilizar técnicas de Machine Learning, se detectará si se encuentran defectos en la producción de productos de acero de Ternium, con esto, se podrá dar información de utilidad a la empresa para que reduzcan o mitiguen dichos defectos, por lo tanto, se verán beneficios económicos para Ternium y una mayor eficiencia y precisión al momento de realizar las órdenes de sus clientes a través de las distintas líneas de producción por las que pasa el acero.

1.5 Objetivo del proyecto: Propósito

1.5.1 Objetivo general

Tener un modelo matemático basado en Machine Learning que pueda identificar las causas de los defectos más significativos con base en los datos brindados por parte de la empresa para lograr líneas de producción más eficientes, mitigando los errores y dando solución a las posibles causantes de los mismos.

1.5.2 Objetivos específicos

- a) Analizar la historia que se tiene de defectos e indicadores de línea Recubiertos.
- b) Armado de una propuesta de solución considerando el comportamiento descubierto en el cruce de la información histórica y los patrones de detección de defectos. (Conocer la empresa socio formadora y los datos proporcionados por la misma).
- c) Establecer un lógica piloto aplicable a línea de Recubiertos como prueba de concepto en donde se valide la hipótesis de causa de generación de defecto. (Una vez identificados los defectos más significativos, y trabajando con un modelo matemático basado en machine learning, se busca generar nuevas estrategias de adaptación y mitigación para la empresa. Toma de decisiones a partir de un análisis de datos).
- d) Dar seguimiento al modelo e ir de la mano con la empresa para así lograr que las líneas de producción sean cada vez más eficientes. (Certificar la prueba de concepto tomando en cuenta producción actual (periodo 1 año de línea Recubiertos))

1.6 Justificación

La ciencia de datos es una rama muy moderna que a la vez incluye otras técnicas como Machine Learning, Deep Learning, Big Data, Artificial Intelligence, entre muchas otras, que dan oportunidad a las empresas de analizar sus datos de una manera más precisa y confiable al ser respaldadas por fundamentos matemáticos, estadísticos y programación. La empresa socio formadora, Ternium, busca un armado de propuesta para determinación de un predictivo de defectos en base a indicadores de comportamiento de cada equipo. Al brindar un dataset, entre otros documentos que son de utilidad, se podrá realizar un análisis basado en ciencia de datos que lleve a brindarles una solución.

1.6.1 Product failure detection for production lines using a data-driven model [3] (Daniel):

En este reporte, se menciona que para tener una línea de producción sana, es importante tener una baja tasa de fallos en los productos finales, así mismo, se menciona que se utiliza el uso de; ANOVA (Análisis de Varianza), métodos de selección de características, normalización de mínimos y máximos, técnica de imputación media, algoritmo de clasificación de bosques

aleatorios, técnica de muestreo de datos y optimización de parámetros de búsqueda de cuadrícula.

Uno de los beneficios que brinda la reducción de errores en líneas de producción es que se producen menos desechos y se ahorra energía eléctrica. Según el artículo, los datos fuente que se obtienen en la producción comúnmente tienen valores nulos debido al ambiente. Así mismo, se menciona que los datos no están balanceados ya que la mayoría de datos pertenecen a la producción que no tiene errores y que la mayoría, pertenece a la producción con errores y que esto requiere de metodologías adicionales.

En el artículo, primeramente se divide en dos, siendo una parte la que se usará como datos de aprendizaje para el modelo y datos de prueba. Para el preprocesamiento de datos, se ocupan los datos que se usarán para entrenar el modelo y aquí se ocupa el método de imputación media para reemplazar valores nulos, y el método de normalización de mínimo y máximo. Para la selección de características, se utiliza ANOVA, luego, para el muestreo se usan técnicas de sobremuestreo y submuestreo para mejorar el rendimiento de los algoritmos. Para la optimización de parámetros se utiliza la técnica de búsqueda de cuadrícula. Se menciona que se utilizaron cinco algoritmos de clasificación: Support Vector Classifier (SVC), Multilayer Perceptron (MLP), Random Forest (RF), Gradient Boosted Trees (GDBT) y, RUSBoostedTree (RUSBT).

El artículo menciona que las técnicas de sobremuestreo y submuestreo son efectivas al tener un problema de desbalance de clasificación. Por otro lado, la selección de características quita ruido de las características que menos se correlacionen así como mejorar la precisión de los algoritmos, por lo que la combinación de ambas técnicas mejoran los métodos utilizados para incrementar la precisión de algoritmos que detectan fallos en los productos.

1.6.2 Predictive maintenance system for production lines in manufacturing: A machine learning approach using IoT data in real-time [1] (Leo)

Este artículo, habla sobre la importancia de la ciencia de datos en la industria ya que permite crear nuevos modelos para la creación de ecosistemas lo cual permite flexibilizar el proceso productivo a través del intercambio de

los datos. El primer paso para entrar a esta nueva era es el aprovechamiento de la nube, lo cual permite subir una cantidad infinita de datos, y es que los datos permiten crear estrategias globales en las empresas; por ejemplo, se hace mención de cómo se puede “analizar la computación en la nube y el big data bajo la referencia de la arquitectura de implementar la industria 4.0”, la industria 4.0 se refiere a la cuarto revolución industrial, usando el enfoque de la tecnología en los negocios y cómo la industria se han obligado a competir.

Cloud computing es el término que se da al almacenamiento de los datos, este se define como el proceso de la producción y se ve afectado por los algoritmos ya que los sensores capturan una mayor cantidad de datos que una persona, lo cual permite la disminución del desperdicio de los materiales, haciendo beneficio a las empresas porque de este modo se reducen los costos y se crean nuevos para cuidar la parte de las finanzas. Por otra parte, el big data es el complemento del cloud computing; este permite analizar el volumen de datos almacenados generando un ecosistema de producción centrado en los datos, por ejemplo, cuando se habla de la “optimización”, se centra en crear respuestas óptimas a las problemáticas.

En el artículo se habla sobre el análisis de la correlación con los datos, que sirve para “medir la fuerza la relación lineal entre 2 variables y calcular su asociación” y es que en el ejemplo las variables altamente correlacionadas se derivaron de diferentes sensores correspondiendo en el subsistema de unidad de producción ya que permite identificar la causa y el efecto que tienen a la hora de la fabricación, además el aprendizaje automático, es una tarea esencial para el modelado de los datos, lo cual ayuda a crear estrategias para la reducción de errores para la predicción de información, por ejemplo el PCA, es una técnica de reducción de dimensiones que proyecta un conjunto de variables en un conjunto más pequeño de nuevas dimensiones artificiales para capturar las variaciones más grandes.

Concluyendo con el artículo, las estrategias tradicionales de mantenimiento de la producción implican el reemplazo de equipos que ocurren después de una falla o en periodos de tiempo, por la falta de información y no tener un análisis de los datos de cuanto es la vida promedio de la maquinaria por lo tanto no se prevén los costos adicionales debido a la sustitución anticipada de los equipos, por ende crear un algoritmo que pueda prevenir las

fallas de la maquinaria lograría alcanzar a prever o crear un ahorro y que así la empresa esté asesorada y pendiente de que probablemente haya un gasto extra y no sea algo que les llegue de sorpresa.

1.6.3 Data quality for data science, predictive analytics, and big data in supply chain management: An introduction to the problem and suggestions for research and applications [2] (Carlos)

En la actualidad, los expertos en cadenas de producción están inundados de datos, con nuevas maneras de pensar cómo los datos son producidos, organizados y analizados. Esto ha incentivado a las empresas a adoptar y perfeccionar funciones de análisis de datos para mejorar los procesos de las cadenas de producción. En este reporte, se realizan sugerencias sobre búsqueda y aplicaciones relacionadas al análisis de datos en las cadenas de producción.

La propuesta que se hace en el reporte incluye que considerar la calidad de los datos debería ser una práctica común en futuras búsquedas sobre data science, predictive analytics, y big data en líneas de producción. Como mínimo, la calidad de los datos debería ser conocida, medible, monitoreada y controlada. Idealmente, el objetivo de cualquier actividad de medición o monitoreo debería ser mejorar la calidad del proceso y del producto. Se cree que con tan sólo conocer, medir y monitorear la calidad de los datos de la cadena de producción sería inevitable mejorar la calidad de los datos. Además, establecer un esquema de monitoreo continuo permitirá controlar futuras adquisiciones de datos, con el objetivo de mejorar la calidad de los datos y por último, la toma de decisiones basadas en los datos.

La creciente importancia de los datos en las cadenas de producción debería liderar a que las empresas concienticen en la necesidad de tener datos de mayor calidad, ya que los resultados de la toma de decisiones basadas en datos de baja calidad podrían llegar a ser muy costosos. Los gerentes de las cadenas de producción deberían empezar a ver la calidad de los datos para tomar decisiones de la misma manera que ven la calidad de los productos que entrega la cadena de producción. Los gerentes que aprecian el valor de los productos de datos que son precisos, consistentes, completos y puntuales deberían de considerar el potencial de usar métodos de control para mejorar la

calidad de los datos de la misma manera que utilizan los métodos para mejorar la calidad de los productos manufacturados.

1.6.4 Advances and opportunities in machine learning for process data analytics [4] (Mariana)

En este artículo se abordan cinco temas principales sobre el avance y las oportunidades de la aplicación de machine learning para el análisis de datos de sistemas de fabricación por procesos, el cual se refiere a los métodos de producción que crean bienes a partir de la combinación de materiales crudos usando una fórmula o receta, este artículo se relaciona perfectamente al reto, ya que el socio formador con el que se está trabajando, Ternium, es una empresa productora de aceros planos a partir de un proceso industrial con materia prima.

El primer tema trata sobre el impulso actual de machine learning y artificial intelligence basado en la teoría del aprendizaje estadístico y los éxitos comerciales de las principales empresas de Big Data, como lo mencionan Qin y Chiang “La demostración de AlphaGo por parte de DeepMind de Google a principios de 2016 fue uno de los eventos más impactantes para el mundo.” (*Qin & Chiang, 2019*) ya que éste probó en más de una ocasión su eficiencia al ganar todos los juegos de la Asian Go Championship en 2016, gracias a la combinación de machine learning y técnicas de búsqueda de árbol en una base de datos masiva que contenía registros de tanto los jugadores humanos como de los jugadores máquina o digitales. El segundo punto fue una breve introducción sobre las características de los sistemas de fabricación por procesos y el desarrollo del análisis de datos en las últimas tres décadas. El tercer punto y el más importante habla sobre los tres atributos del análisis de datos de procesos para hacer que las técnicas de machine learning sean aplicables en las industrias de procesos, estos tres atributos incluyen, que sea compatible con los principios del modelo, que pueda adaptarse a incertidumbres que cambian con el tiempo y que genere soluciones interpretables. El cuarto punto habla sobre cómo el machine learning y otras técnicas modernas son una rama aplicable para el futuro de la ciencia de datos; y finalmente el quinto punto habla sobre la adopción de una cultura de ciencia de datos para las industrias en general.

Como conclusión, lo más relevante del artículo y su relación al reto, es que las operaciones y el control de procesos industriales tienen que lidiar con algunos defectos que tienen ciertos principios conocidos, por lo que es posible analizarlos con ciencia de datos y nuevas técnicas como lo son, machine learning, artificial intelligence o deep learning, para poder encontrar los estados que no son conocidos (información que pareciera no ser relevante u oculta dentro de las bases de datos) y por lo tanto, poder llevar a cabo una mejor toma de decisiones que reduzca los defectos y el impacto de los mismos.

1.6.5 CRISP-DM: Towards a Standard Process Model for Data Mining [6]

(Juan)

Este artículo habla de la importancia del método de orientación CRISP DM para la minería de datos. Esto es de relevancia para este proyecto, ya que se basa en la metodología CRISP-DM para el análisis de los datos de la empresa.

En este artículo se concluye que la metodología CRISP-DM es muy efectiva, de hecho, mostró ser todavía más efectiva de lo que se anticipa anteriormente, por lo que queda demostrado que este método puede ser de alta relevancia para la realización de nuestro análisis descriptivo y predictivo.

1.7 Mercado potencial

Todas aquellas empresas que utilizan líneas de recubrimiento y láminas de acero, para la creación de sus artículos, ya sean piezas automotrices, muebles de cocina como estufas y refrigeradores, contenedores industriales, entre otros. Además de las varillas, tuberías y otros productos de acero y hierro de la más alta calidad. El mercado es amplio en general ya que cuenta con 20 centros de producción y clientes alrededor de todo el continente americano.

1.8 Identificación de clientes/consumidor y usuarios.

- Proceso industrial de la empresa Ternium.
- Área de control de calidad, producción y mantenimiento de la empresa Ternium.
- Sector de la construcción, automotriz, línea blanca, agroindustria y transporte.

1.9 Plan de actividades del proyecto

El socio formador proporcionó una presentación, que incluye el contexto del proceso industrial con materia prima, las premisas incluyendo el modelo analítico que ve lo que puede pasar en recubiertos viendo la información de línea de masivos. De igual manera, la presentación incluye el objetivo, los alcances, los líderes involucrados, beneficio de negocio y las áreas impactadas del proyecto.

El socio formador proporcionó un data set cuya estructura de información incluye la información de LíneaA con datos como el peso, despunte, descole, norma, grado, espesor, velocidad, entre otros datos físicos, química del abasto, proveedor del material, la información de LíneaB que está relacionada con el diseño, al igual que la receta LíneaB, la línea DefectoPrincipal LíneaB incluye el objetivo, también se incluye la información del cliente final, información teórica de recubiertos, información de temperaturas, así como la información teórica de recubiertos.

2. Comprensión de los datos

2.1 Descripción del set de datos (dimensiones) y sus variables, tipo de dato y origen (quitar columnas que no tienen datos).

a) Dimensión del dataset

Se inició dimensionando el dataset, con el método df.shape se obtuvo que existen 23722 filas y 375 columnas, de igual manera, con el método df.size se obtuvo que el data frame contiene 8895750 elementos.

```
[ ] df.shape
```

```
(23722, 375)
```

```
[ ] df.size
```

```
8895750
```

b) Descripción de los datos

Se utilizó el método df.info(verbose = True) para obtener el nombre y tipo de dato por columna, se obtuvo que existen 74 columnas con tipo de dato flotante, 243 tipo entero y 58 tipo objeto.

```
[ ] df.info(verbose=True)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 23722 entries, 0 to 23721
Data columns (total 375 columns):
 #   Column           Dtype  
 --- 
 0   Material Entrada    object  
 1   Material Salida    object  
 2   Cliente            object  
 3   Peso Teorico       int64   
 4   Peso Báscula        int64   
 5   Peso                int64   
 6   Peso Báscula Entrada float64 
 7   Peso Mínimo         int64   
 8   Peso Máximo         int64   
 9   Descarte Est. Prep. float64 
 10  Despuente Entrada   float64 
 11  Descole Entrada    float64 
 12  Norma               object  
 13  Grado               object  
 14  Subnorma            float64 
 15  Forma Final          float64 
 16  Ancho Salida         float64 
 17  Ancho Mínimo         float64 
 18  Ancho Máximo         float64 
 19  Espesor Salida       float64 
 20  Espesor Mínimo       float64 
 21  Espesor Máximo       float64 
 22  Espesor Punta        float64 
 23  Espesor Cola         float64 
 24  Largo                float64 
 25  Calibre              float64
```

Para conocer la cantidad de valores nulos por columna en el dataset se utilizó el método `df.isna().sum()`, de igual manera, para conocer los posibles valores que se podrían encontrar por columna, se utilizó el método `df.apply (pd.value_counts)`.



```
df.isna().sum()
```

```
Material Entrada      0
Material Salida       0
Cliente                0
Peso Teorico          0
Peso Báscula          0
...
Temp10                 0
Temp11                 0
Temp12                 0
PCTJE_REDUCCION       0
TIPO_RECUBRIMIENTO   0
Length: 375, dtype: int64
```

```
[ ] df.apply(pd.value_counts)
```

	Material Entrada	Material Salida	Cliente	Peso Teorico	Peso Báscula	Peso Báscula Entrada	Peso Mínimo	Peso Máximo	Descarte Est. Prep.	Despuente Entrada	Descole Entrada	Norma	Grado	Subnorma	Forma Final	Ancho Salida	Ancho Mínimo	Ancho Máximo	Espesor Salida	Espesor Mínimo	Espesor
-8830.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	1.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
-4190.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
-4080.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
-3960.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
-3880.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
...	
Z 140	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
Z100	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
Z18	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
Z27	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
Z275	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

138533 rows x 375 columns

2.2 Exploración de datos

a) Medias estadísticas

Se utilizó el método `df.describe()` para mostrar las medias estadísticas ya que este método muestra datos como la cantidad de valores por columna, la media, desviación estándar, primer, segundo y tercer cuartil, así como el mínimo y máximo valor de cada columna. Para calcular la varianza se utilizó el método `df.var()`, para calcular la moda se utilizó `df.mode(numeric_only = True)` para mostrar sólo los valores numéricos.

```
[ ] df.describe()
```

	Peso Teorico	Peso Báscula	Peso	Peso Báscula Entrada	Peso Mínimo	Peso Máximo	Descarte Est. Prep.	Despuente Entrada	Descole Entrada	Subnorma	Forma Final	Ancho Salida	Ancho Mínimo	Ancho Máximo
count	23722.000000	23722.000000	23722.000000	23719.000000	23722.000000	23722.000000	6476.000000	23292.000000	23296.000000	0.0	0.0	23722.000000	23722.000000	23722.000000
mean	23683.633842	23761.047129	23776.155889	12637.289515	10707.916702	28517.566436	618.083539	4830.825348	3539.118776	NaN	NaN	1432.056444	1415.822542	1440.822542
std	3215.394066	3226.431778	3166.548831	12363.235631	1735.175738	2882.198350	598.483793	1675.853056	1362.442006	NaN	NaN	264.347739	264.228367	264.228367
min	0.000000	0.000000	132.000000	0.000000	10000.000000	18000.000000	1.000000	-830.000000	-515.000000	NaN	NaN	808.000000	786.000000	811.000000
25%	22479.000000	22550.000000	22560.000000	0.000000	10000.000000	28806.000000	367.000000	3550.000000	3229.000000	NaN	NaN	1276.000000	1265.000000	1290.000000
50%	25170.000000	25230.000000	25230.000000	16810.000000	10000.000000	30000.000000	447.000000	4770.000000	3272.000000	NaN	NaN	1452.000000	1440.000000	1465.000000
75%	25889.000000	25980.000000	25980.000000	25840.000000	10900.000000	30000.000000	709.000000	5960.000000	3675.000000	NaN	NaN	1630.000000	1614.000000	1639.000000
max	27639.000000	27790.000000	26740.000000	27010.000000	30000.000000	30000.000000	7869.000000	32767.000000	32767.000000	NaN	NaN	1860.000000	1848.000000	1873.000000

Varianza

```
[ ] df.var()
C:\Users\danie\AppData\Local\Temp\ipykernel_38584\1568254755.py:1: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in
df.var()
Peso Teórico      1.033876e+07
Peso Báscula     1.040986e+07
Peso             1.002703e+07
Peso Báscula Entrada 1.528496e+08
Peso Mínimo      3.010835e+06
...
Temp9            3.347840e+03
Temp10           7.389717e+02
Temp11           7.996487e+02
Temp12           1.035319e+03
PCTJE_REDUCCION 1.918821e+00
Length: 317, dtype: float64
```

Moda

```
[ ] df.mode(numeric_only=True)
```

	Peso Teórico	Peso Báscula	Peso Entrada	Peso Mínimo	Peso Máximo	Descarte Est. Prep.	Despuente Entrada	Descole Entrada	Subnorma	Forma Final	Ancho Salida	Ancho Mínimo	Ancho Máximo	Espesor Salida	Espesor Mínimo	Espesor Máximo	Espesor Punta	Espesor Cola	Largo	Calibre	
0	25440	26210	26210	0.0	10000	30000	404.0	3330.0	3278.0	NaN	NaN	1450.0	1820.0	1845.0	0.637	0.6183	0.6583	NaN	NaN	2824.0	NaN

Para las variables cualitativas se creó una copia del data frame original y seleccionando sólo los tipos de datos objeto excluyendo los flotantes y enteros para realizar la tabla de distribución de frecuencia con el método df1.apply(pd.value_counts) y para calcular la moda con el método df1.mode(numeric_only=False).

Variables cualitativas

```
[ ] df1 = df.copy()
df1 = df1.select_dtypes(include=object, exclude=int and float)
```

Tabla de distribución de frecuencia

```
[ ] df1.apply(pd.value_counts)
```

	Material Entrada	Material Salida	Cliente	Norma	Grado	Código Acabado	Número Parte	Dictamen	Resolución	PRAM	Grado Acero Molino	Observaciones Material Entrada	Acabado Superficial	Empaque Cumple	Cara Garantizada	Cara Inspeccionar	Cara Programada	Figura
100_00023	NaN	NaN	NaN	NaN	18.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
100_00040	NaN	NaN	NaN	NaN	45.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
63101-T7AA-A000	NaN	NaN	NaN	NaN	NaN	NaN	37.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
66712-H8000	NaN	NaN	NaN	NaN	NaN	NaN	3.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
760224JJL0	NaN	NaN	NaN	NaN	NaN	NaN	16.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
...	
TRATAMIENTO L	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
PASIVADO	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
AUTOMOTRIZ	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
RECUB01	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
RECUB02	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

b) Exploración de los datos usando herramientas de visualización

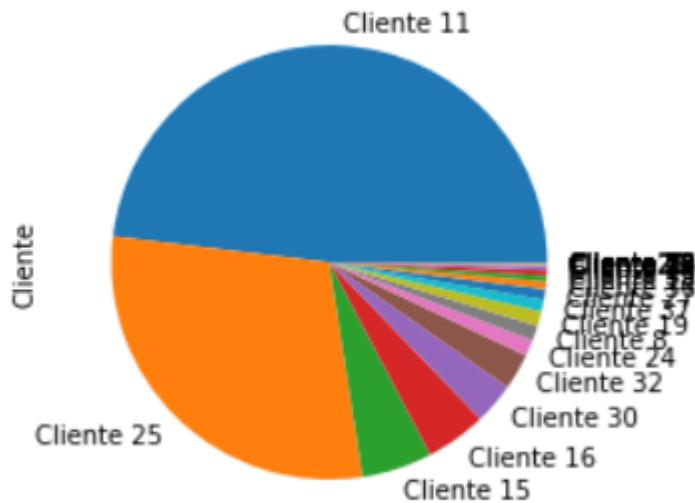
Se inició explorando las variables categóricas mediante diagramas de pastel, creando un nuevo data frame que sólo incluyera los tipos de datos objeto, este data frame tiene una dimensión de 17456 filas y 444 columnas.

Se realizó la exploración de datos con el método:

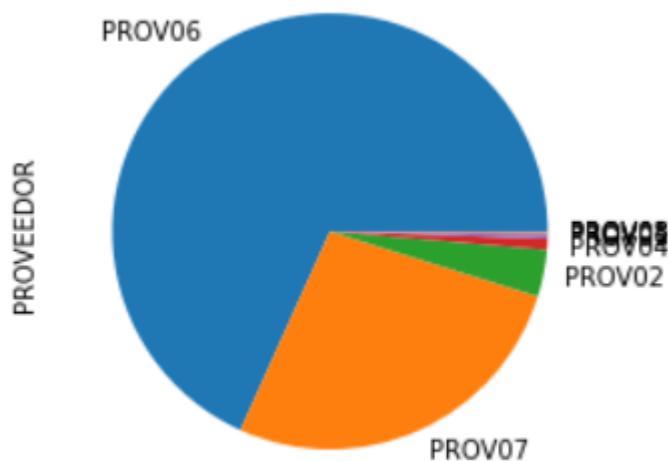
```
for column in newdf2:
```

```
    plt.figure()
    newdf2[column].value_counts().plot(kind='pie')
```

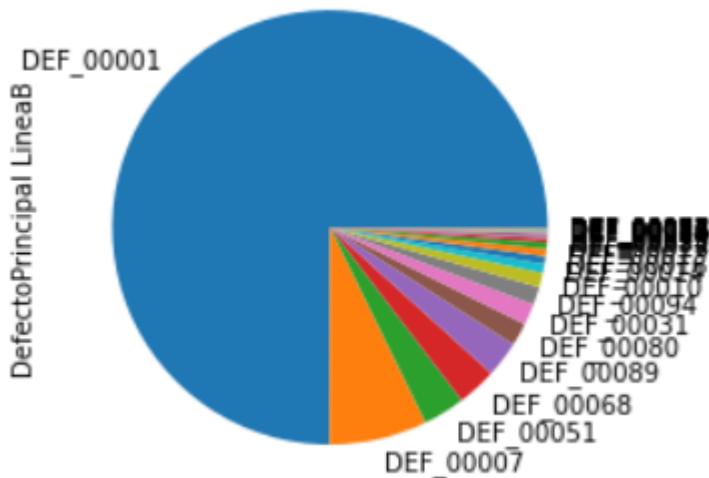
Algunas de las gráficas que se obtuvieron fueron:



La gráfica muestra la organización de número de datos por cliente, se observa que el Cliente 11 acapara casi la mitad de los datos totales, seguido del Cliente 25 que reúne poco más del 25% del total de los datos, el resto de los datos se concentra entre los demás clientes dividido en proporciones similares.



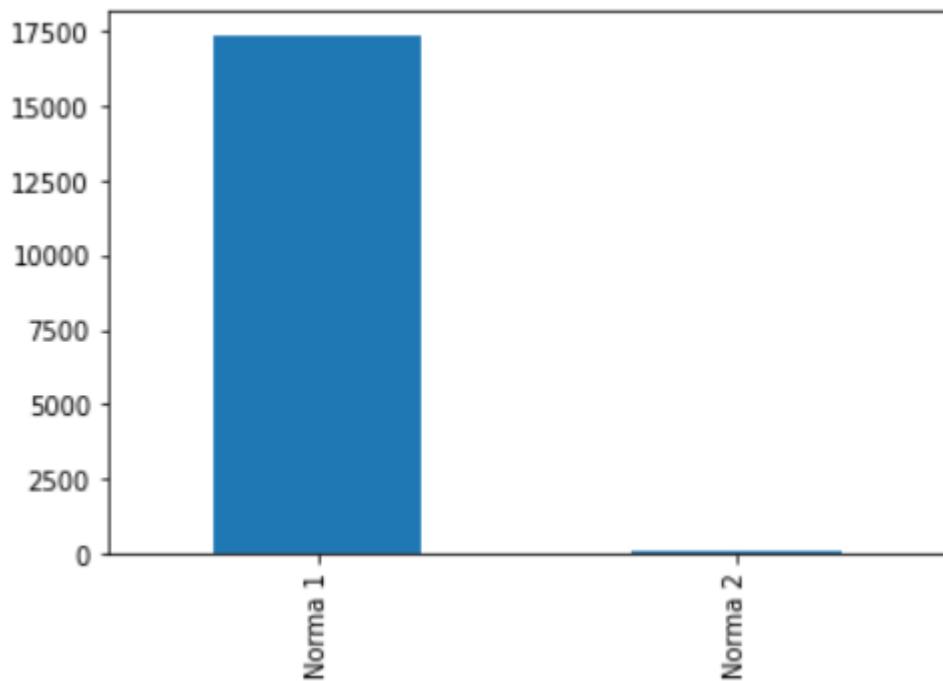
La gráfica muestra la concentración de los datos por proveedor, se observa que el PROV06 contiene una gran parte de los datos, casi el 75%, de igual manera se observa que el PROV07 concentra una cuarta parte de los mismos, el resto del porcentaje se divide entre los proveedores restantes.



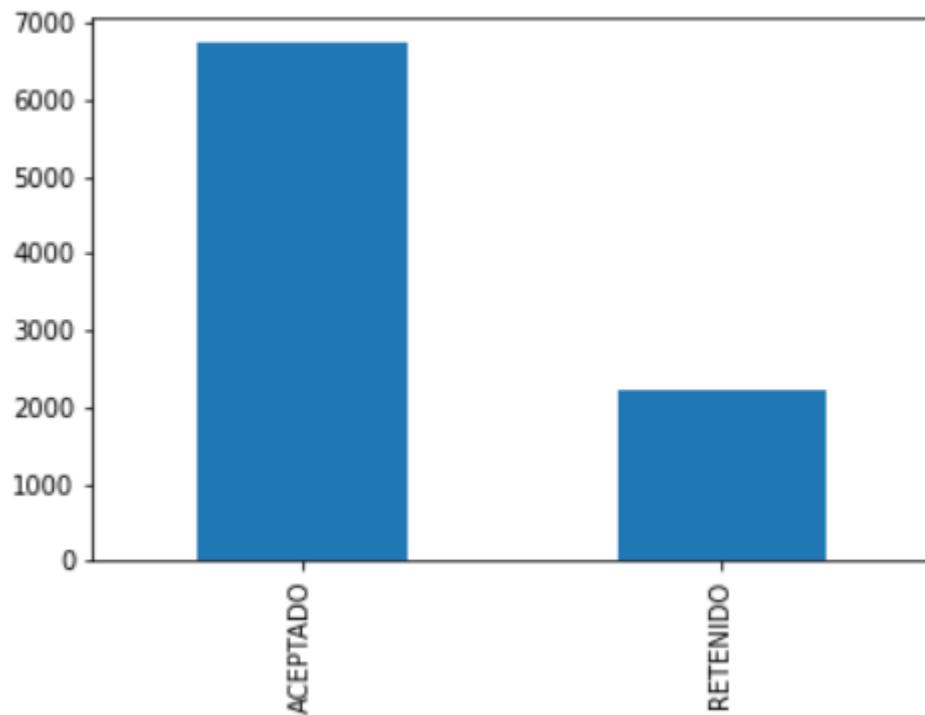
La gráfica muestra el DefectoPrincipal LineaB, se observa que el DEF_00001 concentra el 75% de los datos, el resto se dividen entre los defectos restantes.

De igual manera, se realizaron diagramas de barras para visualizar los datos, se realizaron con el método:

```
for column in newdf2:  
    plt.figure()  
    newdf2[column].value_counts().plot(kind='bar')
```



En el diagrama se observa que hay una tendencia hacia la Norma 1 ya que contiene aproximadamente 17500 datos, en contraste, la Norma 2 casi no se alcanza a distinguir a su baja cantidad de datos.



En el diagrama se observa que la mayoría de datos fueron aceptados, se puede ver que en el data frame hay aproximadamente 7000 datos aceptados y 2000 datos retenidos.

Para explorar las variables cuantitativas se realizaron análisis de correlación de los datos, análisis de distribución de los datos mediante histogramas, de igual manera se obtuvieron medidas de posición no-central mediante boxplots.

Los boxplots se realizaron con el método:

```
for column in newdf:
```

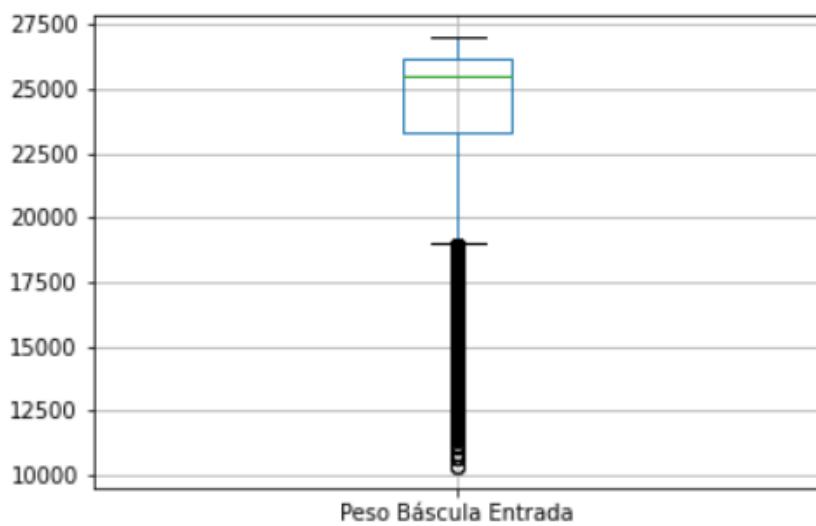
```
    plt.figure()  
    newdf.boxplot([column])
```



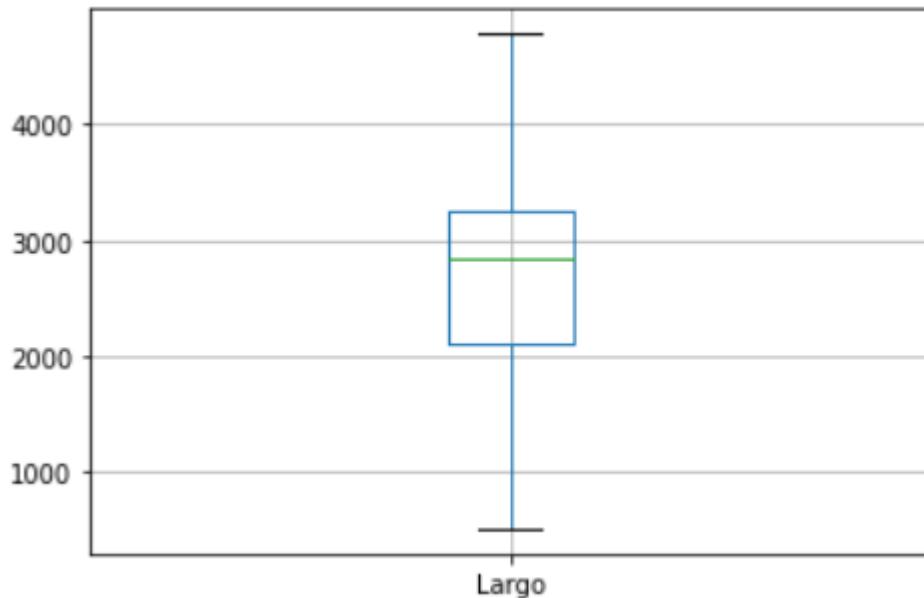
```
for column in newdf:  
... plt.figure()  
... newdf.boxplot([column])
```



C:\Users\danie\AppData\Local\Temp/ipykernel_38584/2260657119.py:2:
plt.figure()



En el boxplot para el Peso Báscula Entrada se observa que el promedio de los valores se encuentra entre 23000 y 26000, sin embargo, fuera del diagrama hay muchos valores atípicos.



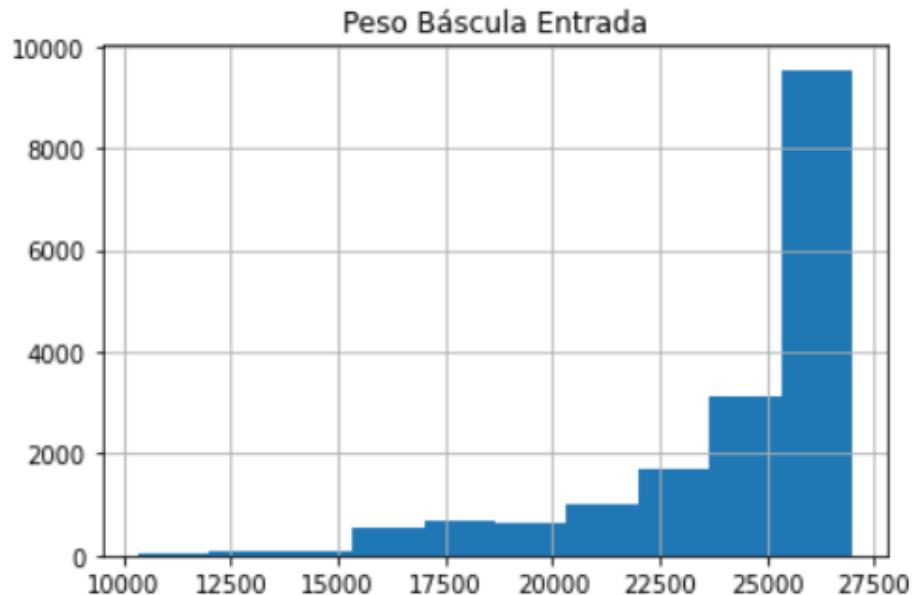
En el boxplot para el Largo, se observa que no hay valores atípicos, todos los valores se encuentran dentro del rango, el promedio se encuentra entre 2100 y 3300.

Para realizar los histogramas se utilizó el método:

```
for column in newdf:  
    plt.figure()  
    newdf.hist([column])
```

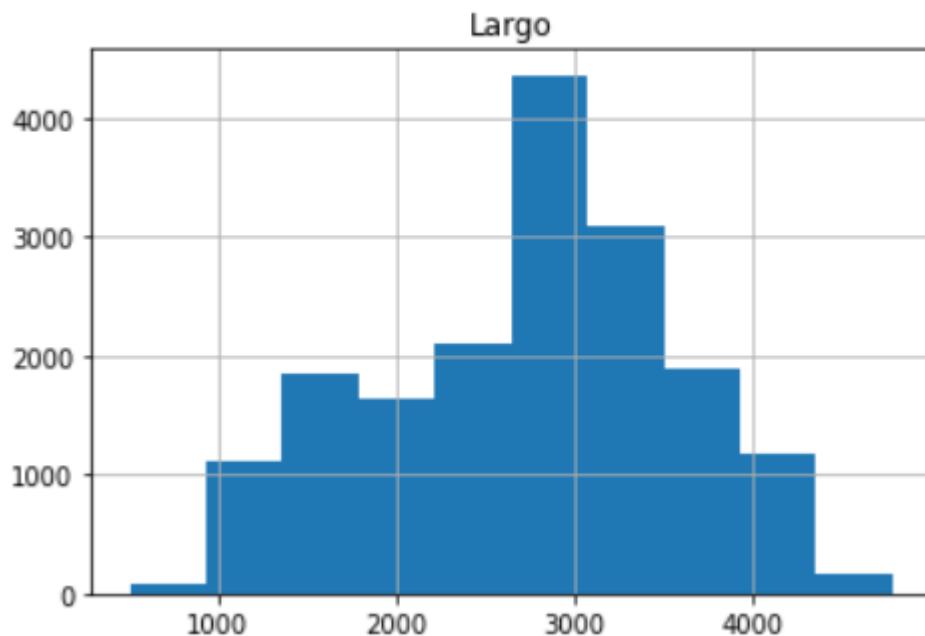
```
▶ for column in newdf:  
    plt.figure()  
    newdf.hist([column])
```

```
C:\Users\danie\AppData\Local\Temp\ipykernel_38584/3315212  
plt.figure()  
<Figure size 432x288 with 0 Axes>
```



En el histograma para el Peso Báscula Entrada se puede observar que la distribución de los datos radica entre 23000 y 27000, teniendo este último valor la mayor cantidad con datos con más de 9000.

<Figure size 432x288 with 0 Axes>



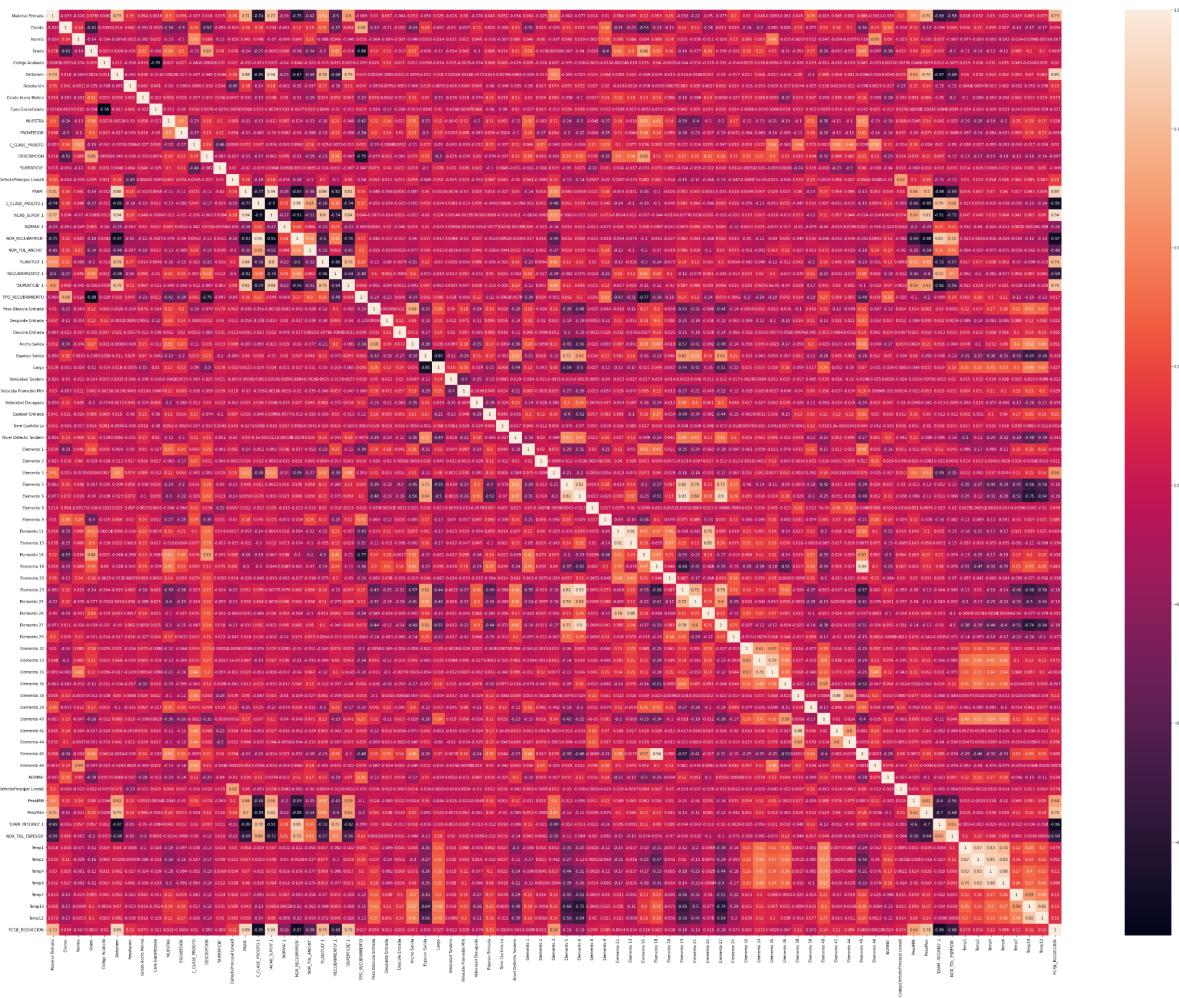
En el histograma para el Largo se observa que los datos están distribuidos de manera que no hay valores atípicos, la media podría encontrarse entre 2700, lo que es un valor central tomando en cuenta los valores mínimos y máximos.

Se observa que existen tanto formas simétricas como asimétricas dependiendo de la columna.

El análisis de correlación se realizó con el método newdf.corr(). De igual manera, se realizó un mapa de calor con el método sns.heatmap(finaldf2.corr()) para mostrar el análisis de una manera más visual.

	Peso Báscula Entrada	Despuente Entrada	Descole Entrada	Ancho Salida	Espesor Salida	Largo	Velocidad Tandem	Velocidad Promedio Mrh	Velocidad Decapado	Peso Entrada	Ancho Entrada	Espesor Entrada	Ancho Set Trimmer	Torre Cuchilla Lo	Torre Cuchilla Lm	Nivel Defecto Tandem
Peso Báscula Entrada	1.000000	0.000704	0.032447	0.675246	-0.333957	0.293018	0.050288	0.197942	-0.188352	0.976291	0.675123	0.161967	0.675258	0.027719	0.030170	-0.292054
Despuente Entrada	0.000704	1.000000	0.118481	0.091032	-0.182238	0.122706	0.012325	0.079405	-0.109115	0.001891	0.092164	0.096189	0.092401	0.056842	0.061187	-0.139366
Descole Entrada	0.032447	0.118481	1.000000	0.072937	-0.173539	0.136392	0.020398	0.061969	-0.066155	0.035670	0.074019	0.081824	0.074123	0.016564	0.015013	-0.107497
Ancho Salida	0.675246	0.091032	0.072937	1.000000	-0.381695	0.035354	0.009728	0.204037	-0.345504	0.698255	0.999880	0.170924	0.999761	-0.005407	-0.000983	-0.355604
Espesor Salida	-0.333957	-0.182238	-0.173539	-0.381695	1.000000	-0.847795	-0.120029	-0.319242	0.150843	-0.330228	-0.383907	0.173296	-0.383862	-0.052097	-0.054453	0.524115
...
Temp9	0.321704	0.087612	0.105938	0.406584	-0.431599	0.303380	0.025395	0.164375	-0.171484	0.329877	0.408224	0.159884	0.408254	0.039848	0.034859	-0.284508
Temp10	0.425546	0.121242	0.145164	0.523913	-0.636693	0.477455	0.018967	0.249811	-0.261315	0.436750	0.526562	0.247227	0.526595	0.008314	0.010708	-0.475234
Temp11	0.429219	0.121209	0.140430	0.522728	-0.637067	0.478226	0.018772	0.252414	-0.255456	0.439092	0.525289	0.233148	0.525295	0.004012	0.006653	-0.458235
Temp12	0.373069	0.093443	0.141765	0.457559	-0.562297	0.418174	0.008936	0.224632	-0.211922	0.381294	0.459586	0.191770	0.459614	-0.010546	-0.007999	-0.391526
PCTJE_REDUCCION	0.058724	0.022005	0.055481	0.051040	-0.034609	0.026748	-0.022341	0.031460	-0.078773	0.063667	0.051930	0.247758	0.051847	-0.002433	-0.009569	-0.041355

80 rows × 80 columns



2.3 Calidad de datos

Se sustituyeron los valores nulos de los datos numéricos por el promedio de la columna y los valores nulos de los datos categóricos con la moda de la columna. Se eliminó una columna que no es de utilidad. Se retomó el paso de cambiar los datos categóricos a numéricos y se concatenaron ambos data frames, categóricos y numéricos, sólo que el categórico ya ya transformado a numérico. Se revisaron las filas y columnas y se realizó la matriz de correlaciones del data frame concatenado. Se eliminaron las columnas con una correlación mayor al 95%, ya que al estar tan correlacionadas no proporcionan utilidad. Se obtuvo el data frame final con las columnas útiles basado en la matriz de correlaciones. Finalmente, se revisó que todos los datos fueran efectivamente numéricos.

Sustituimos los NaN de los datos numéricos por el promedio de la columna y los NaN de los datos categóricos con la moda de la columna

```
[ ] for column in newdf:  
    newdf[column] = newdf[column].fillna(newdf[column].mean())  
  
for column in newdf2:  
    newdf2[column] = newdf2[column].fillna(newdf2[column].mode())  
  
newdf2.isnull().sum().sum()  
  
8500
```

Quitamos una columna que no es de utilidad

```
[ ] newdf2 = newdf2.drop(["DESCRIPCION_1"], axis = 1)
```

Retomamos el paso de cambiar los datos categóricos a numéricos y concatenamos ambos dataframes, categóricos y numéricos, solo que el categórico ya se vuelve numérico.

```
[ ] dfcat = newdf2.copy()  
  
from sklearn.preprocessing import LabelEncoder  
  
for column in list(dfcat.columns):  
    dfcat[column] = dfcat[column].astype(str)  
  
Le = LabelEncoder()  
for column in list(dfcat.columns):  
    dfcat[column] = Le.fit_transform(dfcat[column])  
  
finaldf = pd.concat([dfcat, newdf], axis = 1)
```

Revisamos las filas y columnas del dataframe concatenado

```
[ ] finaldf.shape  
  
(17456, 123)
```

Realizamos la matriz de correlaciones del dataframe concatenado

```
[ ] corr = finaldf.corr().abs()  
  
▶ upper_tri = corr.where(np.triu(np.ones(corr.shape),k=1).astype(np.bool))  
C:\Users\danie\AppData\Local\Temp\ipykernel_38584/3688950637.py:1: DeprecationWarning: `np.bool` is a deprecated alias for the  
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations  
upper_tri = corr.where(np.triu(np.ones(corr.shape),k=1).astype(np.bool))
```

Quitamos las columnas con una correlación mayor al 95%, ya que al estar tan correlacionadas, no proporcionan utilidad

```
[ ] to_drop = [column for column in upper_tri.columns if any(upper_tri[column] > 0.95)]
```

Obtenemos el dataframe final con las columnas útiles basándonos en la matriz de corelaciones

```
[ ] finaldf2 = finaldf.drop(to_drop, axis=1)
```

Revisamos que todos los datos sean numéricos

```
[ ] newdf.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 17456 entries, 0 to 23721
Data columns (total 80 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   Peso Báscula Entrada    17456 non-null   float64
 1   Despuente Entrada      17456 non-null   float64
 2   Descole Entrada        17456 non-null   float64
 3   Ancho Salida          17456 non-null   float64
 4   Espesor Salida        17456 non-null   float64
 5   Largo                 17456 non-null   float64
 6   Velocidad Tandem      17456 non-null   float64
 7   Velocida Promedio Mrh  17456 non-null   float64
 8   Velocidad Decapado     17456 non-null   int64  
 9   Peso Entrada          17456 non-null   int64  
 10  Ancho Entrada         17456 non-null   float64
 11  Espesor Entrada       17456 non-null   float64
 12  Ancho Set Trimmer     17456 non-null   float64
 13  Torre Cuchilla Lo     17456 non-null   int64  
 14  Torre Cuchilla Lm     17456 non-null   int64  
 15  Nivel Defecto Tandem  17456 non-null   int64  
 16  Nivel Defecto Decapado 17456 non-null   int64  
 17  Elemento 1            17456 non-null   float64
 18  Elemento 2            17456 non-null   float64
 19  Elemento 3            17456 non-null   float64
 20  Elemento 5            17456 non-null   float64
 21  Elemento 6            17456 non-null   float64
 22  Elemento 7            17456 non-null   float64
 23  Elemento 8            17456 non-null   float64
 24  Elemento 9            17456 non-null   float64
 25  Elemento 10           17456 non-null   float64
 26  Elemento 11           17456 non-null   float64
 27  Elemento 12           17456 non-null   float64
 28  Elemento 13           17456 non-null   float64
 29  Elemento 15           17456 non-null   float64
 30  Elemento 16           17456 non-null   float64
 31  Elemento 17           17456 non-null   float64
```

3. Preparación de los datos

3.1 Selección de datos

Para la preparación de los datos se utilizarán los datos en donde no se tengan repeticiones en las columnas, por ejemplo, existen 4 columnas con el "Peso" pero sólo una fue relevante, analizando a fondo la base de datos, se observó que hay varias cosas similares,

más adelante se limpiará esto. También hay columnas que sirven de guías para simplificar la información, pero luego de utilizarlas, se vuelven columnas innecesarias, así mismo, se tomará en cuenta la información del socio formador sobre los defectos relevantes y no relevantes. El trabajo estará enfocado en la columna de los defectos, ya que se busca reducir la cantidad de los defectos más relevantes.

3.2 Limpieza de datos

Para realizar la limpieza de datos se comienza seleccionando los tipos de datos entero y flotante, excluyendo los tipo objeto, esto se hace con el método df.select_dtypes(include=int and float, exclude=object).

Se inicia con la eliminación de duplicados con el método df.drop_duplicates(), para la corrección de valores erróneos no hay valores mencionados de este tipo, se procede al manejo de valores faltantes o no relevantes y verificación de calidad de los mismos.

Se empieza con la eliminación de los datos dónde sólo hubiera una opción en la columna.

```
nunique = df.nunique()
cols_to_drop = nunique[nunique == 1].index
df3 = df.copy()
df3 = df3.drop(cols_to_drop, axis=1)
df3
```

	Material Entrada	Material Salida	Cliente	Peso Teórico	Peso Báscula	Peso Entrada	Peso Mínimo	Peso Máximo	Descarte Est. Prep.	Despuente Entrada	Descole Entrada	Norma	Grado	Subnorma	Forma Final	Ancho Salida	
0	MatEnt_00001MAC00	MatSal_00001PDT00	Cliente 30	26209	26130	26130	26530.0	10000	30000	NaN	6710.0	3279.0	Norma 1	TNA_00047	NaN	NaN	1370.0
1	MatEnt_00002MAC00	MatSal_00002PDT00	Cliente 27	25500	25610	25610	0.0	10000	30000	NaN	5810.0	4841.0	Norma 1	TNG_00031	NaN	NaN	1455.0
2	MatEnt_00003MAC00	MatSal_00003PDT00	Cliente 11	26079	26010	26010	26540.0	10000	30000	NaN	5960.0	4853.0	Norma 1	TNG_00031	NaN	NaN	1500.0
3	MatEnt_00004MAC00	MatSal_00004PDT00	Cliente 16	23889	23910	23910	24530.0	10000	30000	NaN	5960.0	4865.0	Norma 1	TNA_00033	NaN	NaN	1580.0

Se prosigue con la eliminación de las columnas con el 49% de los valores nulos.

```
round(len(df3.index) * 0.51)
12098

df3 = df3.dropna(axis=1, thresh = 12098)

df3.describe(include=object)
```

	Material Entrada	Material Salida	Cliente	Norma	Grado	Código Acabado	Dictamen	Resolución	PRAM	Grado Acer Molino	Cara Garantizada	MUESTRA	PROVEEDOR
count	23722	23722	23722	23722	23722	23722	14942	23722	23722	23722	23722	23722	23722
unique	23689	23722	33	2	51	3	2	4	20177	35	3	3361	8
top	MatEnt_09340MAC00	MatSal_00001PDT00	Cliente 11	Norma 1	TNG_00027	MATE	ACEPTADO	SOBRE ORDEN	Pram_1727370A0	GRA_AC27	SUP	ACEZX1105AA03899	PROV06
freq	2	1	10626	23561	4084	17743	9633	15207	2	5475	17872	32	16899

Para la eliminación de filas se empieza por sustituir los valores vacíos de la columna Peso Báscula Entrada por los valores de la columna Peso, posteriormente se eliminan las filas en dónde los datos de ciertas columnas no entran en el rango establecido, se eliminan las columnas en dónde los defectos no fueran relevantes y finalmente se borran las columnas en dónde se establece el mínimo y máximo para ciertas categorías.

Sustituimos los valores vacíos de la columna Peso Báscula Entrada, por los valores de la columna Peso

```
[ ] df4 = df3.copy()
df4["Peso Báscula Entrada"] = np.where(df4['Peso Báscula Entrada'] == 0, df4['Peso'], df4['Peso Báscula Entrada'])
```

Borramos las filas en donde los datos de ciertas columnas no entraran en el rango establecido

```
[ ] df4 = df4.drop(df4[(df4["Peso Báscula Entrada"] < df4["Peso Mínimo"]) | (df4["Peso Báscula Entrada"] > df4["Peso Máximo"])].index)
df4 = df4.drop(df4[(df4["Ancho Salida"] < df4["Ancho Mínimo"]) | (df4["Ancho Salida"] > df4["Ancho Máximo"])].index)
df4 = df4.drop(df4[(df4["Espesor Salida"] < df4["Espesor Mínimo"]) | (df4["Espesor Salida"] > df4["Espesor Máximo"])].index)
df4 = df4.drop(df4[(df4["Velocida Promedio Mrh"] == 0)].index)
df4 = df4.drop(df4[(df4["Velocida Promedio Mrh"] == np.nan)].index)
df4 = df4.drop(df4[(df4["Velocidad Decapado"] == 0)].index)
df4 = df4.drop(df4[(df4["Velocidad Decapado"] == np.nan)].index)
df4 = df4.drop(df4[(df4["Velocidad Tandem"] == 0)].index)
df4 = df4.drop(df4[(df4["Velocidad Tandem"] == np.nan)].index)
```

Borramos las columnas en donde los defectos no fueran relevantes

```
[ ] relevantes = ["DEF_00001", "DEF_00007", "DEF_00051", "DEF_00068", "DEF_00089", "DEF_00080", "DEF_00031", "DEF_00094", "DEF_00010", "DEF_00014"]
df4 = df4[df4["DefectoPrincipal LineaB"].isin(relevantes)]
```

Borramos las columnas en donde se establece el máximo y mínimo para ciertas categorías

```
[ ] df5 = df4.copy()
del df5["Ancho Mínimo"]
del df5["Ancho Máximo"]
del df5["Peso Mínimo"]
del df5["Peso Máximo"]
del df5["Espesor Mínimo"]
del df5["Espesor Máximo"]
del df5["Peso Teórico"]
del df5["Peso Báscula"]
```

Para el manejo de datos categóricos se utilizó el método unique(), se eliminaron columnas y se creó un nuevo data set sin las columnas eliminadas. Se realizó de la siguiente manera:

	Norma	Dictamen	Torre Cuchilla Lo	Torre Cuchilla Lm	Nivel Defecto Tandem	Nivel Defecto Decapado	C_CLASE_PROTO	'DIAM. INTERNO'	'EXTREMOS'	'NOR_TOL_ANCHO'	'NOR_TOL_ESPESOR'	'PLANITUD'	'SUPERFICIE'	'USO_GRAL'	'ACAB_SUPER'.1
0	Norma 1	ACEPTADO	1	1	5	6	M-P-CALIENTE-BOB-SKP	762	ESTANDAR DE LAMINACION	NTA_001	NTE_002	CONTROLADA	SEMI-EXPUESTA	GALV. AUTOMOTRIZ	0
3	Norma 1	ACEPTADO	1	1	5	6	M-P-CALIENTE-BOB-SKP	762	ESTANDAR DE LAMINACION	NTA_001	NTE_002	CONTROLADA	SEMI-EXPUESTA	GALV. AUTOMOTRIZ	0
5	Norma 1	RETENIDO	1	1	5	6	M-P-CALIENTE-BOB-SKP	762	ESTANDAR DE LAMINACION	NTA_001	NTE_002	CONTROLADA	SEMI-EXPUESTA	GALV. AUTOMOTRIZ	0
6	Norma 1	ACEPTADO	1	1	5	6	M-P-CALIENTE-BOB-SKP	762	ESTANDAR DE LAMINACION	NTA_001	NTE_002	CONTROLADA	SEMI-EXPUESTA	GALV. AUTOMOTRIZ	0
7	Norma 1	ACEPTADO	1	1	5	6	M-P-CALIENTE-BOB-SKP	762	ESTANDAR DE LAMINACION	NTA_001	NTE_002	CONTROLADA	SEMI-EXPUESTA	GALV. AUTOMOTRIZ	0
...
23717	Norma 1	RETENIDO	2	2	3	4	M-P-CALIENTE-BOB-SKP	762	ESTANDAR DE LAMINACION	NTA_001	NTE_002	CONTROLADA	SEMI-EXPUESTA	GALV. AUTOMOTRIZ	0
23718	Norma 1	ACEPTADO	2	2	3	4	M-P-CALIENTE-BOB-SKP	762	ESTANDAR DE LAMINACION	NTA_001	NTE_002	CONTROLADA	SEMI-EXPUESTA	GALV. AUTOMOTRIZ	0
23719	Norma 1	ACEPTADO	2	2	3	4	M-P-CALIENTE-BOB-SKP	762	ESTANDAR DE LAMINACION	NTA_001	NTE_002	CONTROLADA	SEMI-EXPUESTA	GALV. AUTOMOTRIZ	0

3.3 Transformación de datos

En cuanto a la transformación de datos no se consideró necesario discretizar los datos o normalizarlos o construir atributos hasta este momento porque al hacerlo, se podría quitar información que pudiera ser relevante para el modelo a seleccionar, pero esto podría cambiar al intentar aplicar modelos al data frame.

La reestructuración de los datos se realizó al momento de la limpieza de datos, el data frame resultante fue el reformato de los datos.

df5	Material Entrada	Material Salida	Cliente	Báscula Entrada	Peso Entrada	Despuente Entrada	Descole Entrada	Norma	Grado	Ancho Salida	Espesor Salida	Largo	Código Acabado	Velocidad Tandem	Velocidad Promedio Mrh	Velocidad Decapado	Dictamen	Peso Entrada	Resolución
0	MatEnt_00001MAC00	MatSal_00001PDT00	Cliente 30	26530.0	6710.0	3279.0	Norma 1	TNA_00047	1370.0	0.588	4203.0	MATE TNG	317.0	87.1617	81	ACEPTADO	26560	CON OBSERVACIONES	
3	MatEnt_00004MAC00	MatSal_00004PDT00	Cliente 16	24530.0	5960.0	4865.0	Norma 1	TNA_00033	1580.0	0.692	2837.0	MATE TNG	257.0	83.0547	132	ACEPTADO	24630	CON OBSERVACIONES	
5	MatEnt_00006MAC00	MatSal_00006PDT00	Cliente 11	26100.0	6130.0	4869.0	Norma 1	TNG_00042	1745.0	0.688	2763.0	MATE TNG	326.0	NaN	108	RETENIDO	26100	CALIDAD DE PLANTA	
6	MatEnt_00007MAC00	MatSal_00007PDT00	Cliente 11	25000.0	7310.0	4860.0	Norma 1	TNG_00031	1835.0	0.587	2760.0	MATE	97.0	100.0000	80	ACEPTADO	24990	SOBRE ORDEN	
7	MatEnt_00008MAC00	MatSal_00008PDT00	Cliente 11	25830.0	5860.0	4857.0	Norma 1	TNG_00031	1835.0	0.587	3039.0	MATE	794.0	48.7928	28	ACEPTADO	25870	CON OBSERVACIONES	
...	
23717	MatEnt_23718MAC00	MatSal_23718PDT00	Cliente 11	23590.0	6480.0	3275.0	Norma 1	TNG_00027	1448.0	0.889	2311.0	MATE	203.0	NaN	92	RETENIDO	23690	CALIDAD DE PLANTA	
23718	MatEnt_23719MAC00	MatSal_23719PDT00	Cliente 11	23700.0	4690.0	3264.0	Norma 1	TNG_00027	1454.0	0.889	2331.0	MATE	202.0	84.9851	115	ACEPTADO	23700	SOBRE ORDEN	

Una vez que se tiene un entendimiento de los datos, ahora lo que sigue es filtrarlos para conseguir aquellos datos que puedan ayudar a cumplir con el objetivo planteado.

- Se eliminaron los datos duplicados en las filas.
- Se eliminaron las columnas en donde todas las filas tienen el mismo valor.
- Se cambiaron las variables categóricas a numéricas.
- Se redujo el tamaño de datos de 8,895,750 a 3,155,026

En la segunda etapa del reto, se realizó la comprensión y preparación de los datos de acuerdo a la metodología CRISP DM.

Business understanding

La empresa Ternium proporcionó una presentación para poder comprender a más detalle los objetivos del proyecto, la presentación incluía el contexto del proceso industrial

con materia prima, también se incluyen las premisas, en las cuáles se menciona que se busca predecir los defectos probables con los que sale el material en línea de Recubiertos, tanto el contexto como las premisas incluyen gráficos que facilitan su entendimiento. Finalmente, se incluyó el objetivo de la empresa, sus alcances esperados, el beneficio del negocio y las áreas impactadas. La información proporcionada, permitió conocer a la empresa, saber el contexto de lo que quieren y esperan para poder trabajar en un proyecto adecuado a sus objetivos.

Data understanding

Esta etapa de la metodología CRISP se enfoca en el entendimiento de los datos e incluye las actividades que permiten familiarizarse con los datos, en el cuál se incluyó la descripción del problema específico, idea central del proyecto, hipótesis, justificación, objetivos general y específicos, justificación, en la que cada miembro del equipo citó un artículo científico relacionado al problema, se incluyó una descripción general del mercado potencial, además de identificar clientes/consumidores y usuarios, con el dataset que proporcionó la empresa se realizó una descripción de las fuentes de información, de igual manera se incluyó el impacto social principal y hacia los Objetivos de Desarrollo Sostenible, finalmente se incluyó una línea del tiempo describiendo las actividades a realizar en cada semana del proyecto.

Data preparation

Se realizó la comprensión de los datos del negocio, que incluye la dimensión del dataset, la descripción de los datos, exploración de los datos, verificar la calidad de los datos, también se llevó a cabo la preparación de los datos en dónde se selecciona el conjunto de datos a utilizar, la limpieza, transformación y en caso de ser necesario, la reestructuración de datos.

4. Aplicación de Técnicas de Modelación

4.1 Variables predictoras y variables target

Se definirá como la variable “target” la variable de los códigos de los defectos, llamada *CodigoDefectoPrincipal LineaB*. Esto, debido a que el proyecto se enfoca en buscar predecir si un rollo va a tener un defecto con base en su procesamiento. Asimismo, se definen como algunas variables predictoras las variables de temperatura, de despunte, descole, su espesor, su porcentaje de reducción, entre otras para buscar predecir la variable target.

4.2 Explicación breve de los diferentes modelos de aprendizaje e hiperparámetros

En este proyecto, se enfocará en utilizar un modelo de clasificación. Esto, debido al formato de la variable dependiente, por lo que es más conveniente el utilizar un modelo de clasificación sobre uno de regresión e igualmente, de acuerdo a nuestra revisión de artículos científicos, los modelos de clasificación funcionan mejor que los modelos de clustering..

Los modelos de clasificación que fueron utilizados para realizar el análisis fueron el modelo Linear svc, Decision Tree Classifier y Random Forest Classifier. Para los modelos de clustering se usaron K Means y Spectral Clustering. Esto, con el propósito de buscar el modelo que pueda aportar mayormente a la predicción de los defectos que puedan surgir en los rollos.

Cuando se habla del modelo Linear svc, se hace referencia al modelo “Support Vector Classifier” o también conocido como “Support Vector Machine”. Este modelo busca categorizar los datos que sean proporcionados en un hiperplano para que, con base en estos mismos, se pueda crear una predicción.

Los clasificadores de K Means y de Spectral Clustering son parecidos en cuanto a la forma en la cual categorizan y dividen los datos en agrupaciones, sin embargo, el clasificador “Spectral Clustering” se enfoca más en la descomposición espectral de los datos, mientras que el clasificador “K Means” se enfoca más en hacer un número k de agrupaciones.

Los árboles de decisión o “Decision Tree Classifier” tienen un mayor enfoque en buscar clasificar los datos con base en reglas de decisión con respuestas binarias. Ahora bien, si se juntan varios clasificadores de árboles de decisión y se obtiene su promedio, se puede llegar a un clasificador llamado “Bosques Aleatorios” o “Random Forest Classifier”. Este mismo clasificador se utiliza debido a que puede ayudar a mejorar la precisión del modelaje.

En cuanto a los hiper parámetros, para los algoritmos de clasificación, se cambiaron los tamaños de prueba del modelo y en algunos casos el número de iteraciones, de igual forma en los modelos de clustering, en los que se debe realizar PCA o Principal Component Analysis, y aquí se cambiaron la cantidad de variables a reducir.

4.3 Explicación de la metodología para el entrenamiento y prueba.

Para realizar el entrenamiento y prueba de los modelos, se utilizó la librería sklearn, el cuál es un conjunto de rutinas escritas en Python para hacer análisis predictivo, está basada en NumPy, SciPy y matplotlib.

Se utilizó el método X_train, X_test, y_train, y_test, además de train_test_split, a continuación se mostrarán los métodos que se utilizaron para cada modelo.

```
[ ] from sklearn.model_selection import train_test_split
from sklearn import svm
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report, precision_score
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.linear_model import LogisticRegression
```

- LinearSVC - Classification

```
[ ] X = finaldf2.drop("CodigoDefectoPrincipal LineaB",axis=1)
y = finaldf2["CodigoDefectoPrincipal LineaB"]

[ ] X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
```

Score de validación cruzada

```
[ ] from sklearn import datasets, linear_model
from sklearn.model_selection import cross_val_score
lasso = linear_model.Lasso()
print(cross_val_score(lasso, X_train, y_train, cv=3))

[0.40555951 0.4363866 0.42618772]
```

```
[ ] dt = svm.LinearSVC()
dt.fit(X_train, y_train)
y_pred = dt.predict(X_test)
```

- DecisionTreeClassifier - Classification

```
[ ] dt = DecisionTreeClassifier()
dt.fit(X_train, y_train)
y_pred = dt.predict(X_test)
```

- RandomForestClassifier - Classification

```
[ ] dt = RandomForestClassifier()
dt.fit(X_train, y_train)
y_pred = dt.predict(X_test)
```

- [LinearDiscriminationAnalysis](#)

```
[ ] dt = LinearDiscriminantAnalysis(n_components=2)
dt.fit(X_train, y_train)
y_pred = dt.predict(X_test)
```

- [LogisticRegression](#)

```
dt = LogisticRegression(max_iter=999999)
dt.fit(X_train, y_train)
y_pred = dt.predict(X_test)
```

4.4 Explicación breve de las métricas de evaluación.

Para obtener las métricas de evaluación de los modelos se utiliza la librería `sklearn.metrics`, que implementa varias funciones de pérdida, puntuación y utilidad para medir el rendimiento de la clasificación. Algunas métricas pueden requerir estimaciones de probabilidad de la clase positiva, valores de confianza o valores de decisiones binarias.

La función `accuracy_score()` calcula la precisión, así como la fracción o la cuenta de predicciones correctas. La función `confusion_matrix()` evalúa la precisión de la clasificación calculando la matriz de confusión con cada línea correspondiente a la clase True. La función `classification_report` crea un reporte de texto mostrando las métricas de la clasificación principal.

Igualmente se ocupa el análisis de sensitividad y especificidad en los que se evalúa la calidad de los modelos y se identificó que el modelo escogido sí tenía una buena puntuación en este aspecto.

4.5 Generación de los modelos y su evaluación a través de las diferentes métricas

A continuación, se presentarán los 5 modelos de predicción que se explicaron anteriormente aplicados a cada data frame. Esto, con el propósito de buscar el modelo que mejor se ajuste a los datos y que pueda dar con mayor precisión la predicción de los defectos de los rollos.

1) Modelo SVM

- LinearSVC - Classification

```
[ ] 1 X = finaldf2.drop("CodigoDefectoPrincipal LineaB",axis=1)
2 y = finaldf2["CodigoDefectoPrincipal LineaB"]
3 y.unique()

array([  0,   22,    2,   25,     7, 1171,   111,   674,    19, 1069,     8,
       101,   593,   665,   414,   627,   224,    62, 1346,   319,   700,   94,
        4,   14, 1345,   308,   115,   705, 1096,    36,    98,    40, 1284,
      576,   304])

[ ] 1 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)

[ ] 1 dt = svm.LinearSVC()
2 dt.fit(X_train, y_train)
3 y_pred = dt.predict(X_test)
4 y_pred

/usr/local/lib/python3.7/dist-packages/sklearn/svm/_base.py:1208: ConvergenceWarning:
ConvergenceWarning,
array([  0, 308,    0, ..., 308, 308, 308])
```

2) Modelo DecisionTreeClassifier

- DecisionTreeClassifier - Classification

```
[ ] 1 dt = DecisionTreeClassifier()
2 dt.fit(X_train, y_train)
3 y_pred = dt.predict(X_test)

[ ] 1 np.set_printoptions(threshold=sys.maxsize)
2
3 y_pred_df = pd.DataFrame(y_pred)
4
5 pred_df = y_pred_df.rename(columns={0: "Defectos"})
6
7 pred_df.head()
```

	Defectos	edit
0	0	
1	25	
2	0	
3	2	
4	627	

3) Modelo RandomForestClassifier

- RandomForestClassifier - Classification

```
[ ] 1 dt = RandomForestClassifier()
2 dt.fit(X_train, y_train)
3 y_pred = dt.predict(X_test)
4 y_pred

array([308,    0,    0, ...,   36,    0,    2])
```

4) Modelo K Means Clustering

```
[ ] #Load Data
pca = PCA(35)

finaldf3 = finaldf2.copy()

#Transform the data
finaldf3 = pca.fit_transform(X)

#Initialize the class object
kmeans = KMeans(n_clusters=35)
#kmeans = DBSCAN(eps = 2)

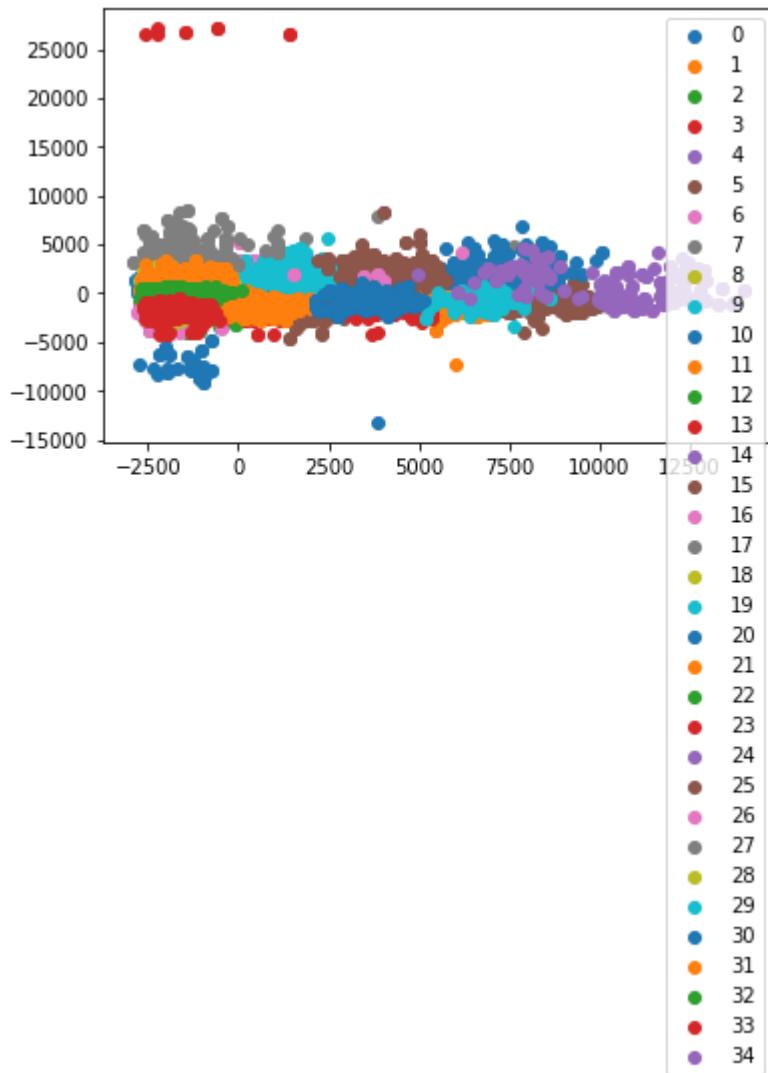
#predict the labels of clusters.
label = kmeans.fit_predict(finaldf3)

true_labels = finaldf2["CodigoDefectoPrincipal LineaB"]

#getting unique labels
u_labels = np.unique(label)

#plotting the results:
for i in u_labels:
    plt.scatter(finaldf3[label == i , 0] , finaldf3[label == i , 1] , label = i)

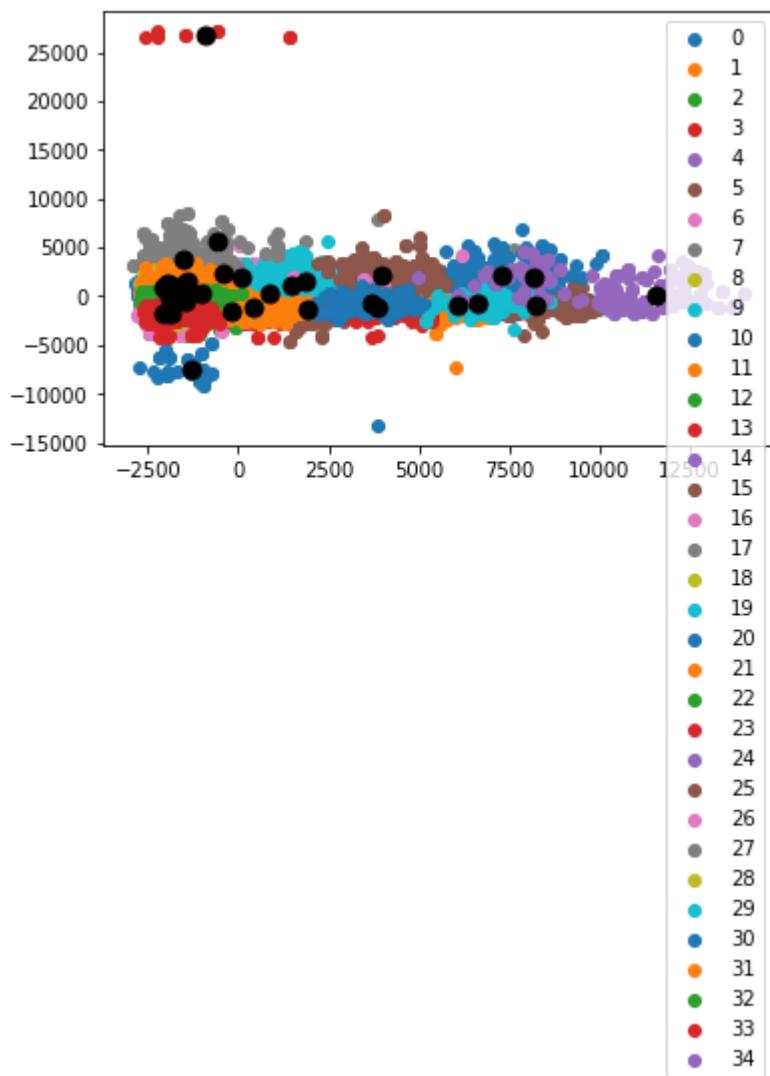
plt.legend()
plt.show()
```



```
centroids = kmeans.cluster_centers_
u_labels = np.unique(label)

#plotting the results:

for i in u_labels:
    plt.scatter(finaldf3[label == i , 0] , finaldf3[label == i , 1] , label = i)
    plt.scatter(centroids[:,0] , centroids[:,1] , s = 80, color = "black")
plt.legend()
plt.show()
```



5) Modelo Spectral Clustering

```

#Load Data
pca = PCA(35)

finaldf3 = finaldf2.copy()

#Transform the data
finaldf3 = pca.fit_transform(X)

#Initialize the class object
sc = SpectralClustering(n_clusters=35)

#predict the labels of clusters.
label = sc.fit_predict(finaldf3)

true_labels = finaldf2["CodigoDefectoPrincipal LineaB"]

#getting unique labels
u_labels = np.unique(label)

#plotting the results:
for i in u_labels:
    plt.scatter(finaldf3[label == i , 0] , finaldf3[label == i , 1] , label = i)

plt.legend()
plt.show()

```

4.6 Ajuste de modelos a través de hiperparámetros.

Con los cinco modelos, se usaron diferentes parámetros como cambiar el tamaño de la muestra, y los clusters, y es que cuánto más pequeño o más grande sea el número en ambos casos, la evaluación será diferente, por ejemplo, en el caso de clustering, permite encontrar grupos de objetos similares, es decir qué objetos están más relacionados entre sí con objetos de otros grupos; en el caso de la muestra, a la hora de tomar el grupo de test (prueba) y train (entrenamiento), será diferente para ambos casos ya que toma el porcentaje de la muestra total.

4.7 Evaluación y selección de modelo(s) de acuerdo a las métricas.

Hiperparámetros utilizados en los modelos.

A continuación se presentan los tests de precisión, sensibilidad y especificidad de todos los modelos de clasificación previamente descritos que fueron utilizados para evaluar el de mayor utilidad para predecir los defectos:

Modelo	SVM	Decision Tree Classifier	Random Forest Classifier	KMeans	Spectral Clustering
	Test size 0.2	Test size 0.2	Test size 0.2	35 KMeans	3 Spectral Clusterings
Accuracy Test	78.6655	97.5658	94.9026	2.1081	73.9688
Test de precisión	78.6655	97.5658	94.9026	2.1081	73.9688
Sensibilidad	0.9996	1.0	1.0	0.7151	1.0
Especificidad	0.0	1.0	0.9959	NaN	0.0

4.7.1 Imagen de árbol de decisión



En este árbol de decisión, se presentan todas las reglas de decisión con las que el algoritmo se basa para tomar una decisión y crear de esa forma una predicción. En este caso, debido a que existen muchas variables en el data frame, el árbol de decisión contiene una gran cantidad de nodos y de ramas.

Ahora bien, el árbol de decisión empieza con un nodo inicial, el cual ramifica con base en el valor de una columna. En este caso, se ramifica con base en la columna

“C_CLASE_PRODTO.1”. Esto genera otros dos nodos, los cuales también se ramifican con base en los valores de ciertas columnas y así sucesivamente.

4.7.2 Tabla de comparación de resultados para cada modelo

Tipo de Modelo	Juan Cantú	Mariana Rincón	Carlos Mateos	Leonardo Laureles	Daniel Núñez
Linear SVC	Tamaño de prueba: 30% Accuracy Test: 68.39% Test de precisión: 68.39% Sensitividad: 1.0 Especificidad: 0.0	Tamaño de prueba: 40% Accuracy Test: 77.01% Test de precisión: 77.01% Sensitividad: 0.9996 Especificidad: 0.0	Tamaño de prueba: 70% Accuracy Test: 58.68% Test de precisión: 58.68% Sensitividad: 0.7056 Especificidad: 0.856	Tamaño de prueba: 60% Accuracy Test: 76.828% Test de precisión: 76.828% Sensitividad: 0.9998 Especificidad: 0.0	Tamaño de prueba: 98.7% Accuracy Test: 24.87% Test de precisión: 24.87% Sensitividad: 1.0 Especificidad: 0.0
Decision Tree Classifier	Tamaño de prueba: 40% Accuracy Test: 97.479% Test de precisión: 97.479% Sensitividad: 1.0 Especificidad: 1.0	Tamaño de prueba: 12.3% Accuracy Test: 96.92% Test de precisión: 96.92% Sensitividad: 1.0 Especificidad: 1.0	Tamaño de prueba: 50% Accuracy Test: 97.204% Test de precisión: 97.204% Sensitividad: 1.0 Especificidad: 1.0	Tamaño de prueba: 30% Accuracy Test: 97.613% Test de precisión: 97.613% Sensitividad: 1.0 Especificidad: 1.0	Tamaño de prueba: 68.91% Accuracy Test: 96.83% Test de precisión: 96.83% Sensitividad: 1.0 Especificidad: 1.0
Random Forest Classifier	Tamaño de prueba: 50% Accuracy Test: 93.81%	Tamaño de prueba: 22.5% Accuracy Test: 94.93%	Tamaño de prueba: 70% Accuracy Test: 93.77%	Tamaño de prueba: 90% Accuracy Test: 90.357%	Tamaño de prueba: 68.91% Accuracy Test: 93.35%

	Test de precisión: 93.81% Sensitividad: 1.0 Especificidad: 1.0	Test de precisión: 94.93% Sensitividad: 1.0 Especificidad: 0.9966	Test de precisión: 93.77% Sensitividad: 1.0 Especificidad: 0.9898	Test de precisión: 90.357% Sensitividad: 1.0 Especificidad: 0.918	Test de precisión: 93.35% Sensitividad: 1.0 Especificidad: 0.97
K-Means	K Means: 35 Accuracy Test: 3.253% Test de precisión: 3.253% Sensitividad: 0.463 Especificidad: NaN	K Means: 50 Accuracy Test: 3.253% Test de precisión: 3.253% Sensitividad: 0.463 Especificidad: NaN	K Means: 75 Accuracy Test: 1.91% Test de precisión: 1.91% Sensitividad: 0.4188 Especificidad: NaN	K Means: 10 Accuracy Test: 6.456% Test de precisión: 6.456% Sensitividad: 0.304 Especificidad: NaN	K Means: 65 Accuracy Test: 2.325% Test de precisión: 2.325% Sensitividad: 0.468 Especificidad: NaN
Spectral Clustering	Spectral Clusterings: 3 Accuracy Test: 0.10% Test de precisión: 0.10% Sensitividad: 0.00389 Especificidad: 1.0	Spectral Clusterings: 50 Accuracy Test: 2.589% Test de precisión: 2.589% Sensitividad: 0.9977 Especificidad: NaN	Spectral Clusterings: 90 Accuracy Test: 0.0057% Test de precisión: 0.0057% Sensitividad: 0.333 Especificidad: NaN	Spectral Clusterings: 15 Accuracy Test: 0.01% Test de precisión: 0.01% Sensitividad: 0.0909 Especificidad: 1.0	Spectral Clusterings: 65 Accuracy Test: 0.022% Test de precisión: 0.022% Sensitividad: 0.25 Especificidad: NaN

Dentro de los modelos de clasificación, se decidió que el mejor modelo fue el de Decision Tree Classifier. Este modelo es bien conocido dentro de Machine Learning por su capacidad de capturar conocimiento descriptivo para la toma de decisiones a partir de los

datos suministrados. Para este modelo se utilizó el parámetro 'criterion:gini', el cual mide la calidad del split de las variables independientes de la dependiente. Su Accuracy score es de 97.57 con un modelo entrenado con el 80% de los datos y probado con el 20% restante, además de una especificidad y sensitividad de 1. Se graficó el modelo con ayuda de la librería de sklearn import tree, donde un nodo representa una instancia, los resultados de la prueba son representados por una rama y el nodo de la hoja personifica la etiqueta de clase. Cabe mencionar, que los modelos de Clustering, no sirven ya que arrojan un score de menos del 5% de precisión.

5. Evaluación

5.1 Evaluación de resultados: Entender e interpretar los resultados obtenidos, su impacto y utilidad, considerando los criterios de éxito del negocio.

Al haber realizado la comparación de los modelos, se entiende que los modelos más efectivos para el análisis predictivo son los modelos de árboles de decisión. Esto, debido a que tanto el modelo de “Decision Tree Classifier” como el de “Random Forest Classifier” mostraron tener un mayor porcentaje en el test de precisión, lo cual significa que son modelos altamente precisos. Se interpreta que se puede obtener con base en el análisis que los modelos de “clustering” son los menos efectivos para predecir los defectos, ya que mostraron tener un menor porcentaje en el test de precisión. Finalmente, se concluye que el modelo elegido (Decision Tree Classifier) no sólo es el mejor modelo de los previamente mencionados, sino que también será de gran utilidad al momento de hacer la predicción, ya que el porcentaje del test de precisión fue muy cercano al 100% para todas las pruebas (entre 96% y 98%).

Impacto y utilidad: Si la empresa Ternium implementa la solución propuesta, se vería beneficiada no solo con un impacto económico sino también de calidad, al reducir los defectos en sus productos, así como elevar su reputación como empresa en el mercado a nivel internacional. Cabe mencionar, que si se le da un seguimiento al proyecto, se le podrían hacer más mejoras a la solución planteada,

5.2 Revisión del proceso: Sumarizar todo el proceso, principales problemas, posibles mejoras.

Resumen del proceso: Se elaboró un análisis de datos basado en la metodología CRISP-DM, primero se buscó entender al negocio, sus necesidades y objetivos, así como los datos proporcionados por la misma; se limpiaron y prepararon los datos para después poder

ser modelados con el modelo de clasificación “Decision Tree” y finalmente se realizó la predicción.

Principales problemas:

- Poca comprensión de las variables en la base de datos.
- Tiempo vs carga de trabajo.
- Cantidad de datos, dimensión del dataset.
- Problemas en el preprocesamiento, limpieza de datos, su integración, transformación y reducción.
- La parte de limpieza de datos es la más tardada y problemática, ya que conforme se avanza con el proyecto, nuevos aspectos que no se consideraron antes surgen, por lo que hay que volver a revisarlos.

Possibles mejoras:

- Organizar de una manera más sencilla toda la información para que pueda ser comprendida por los terceros (Ejecutivos en Ternium interesados en ver nuestro proceso y código).
- Mostrar un diccionario claro y conciso de todas las columnas para que se pueda trabajar más fácilmente con los datos.

5.3 Impacto social principal

Las actividades empresariales y de producción que permitan reducir las complicaciones en los productos finales permite que haya un impacto social tanto negativo como positivo. No obstante, realizar las predicciones antes de cada fase de producción permite que se empleen recursos proporcionales a los rollos finales a producir, ya que al identificar y predecir los posibles defectos que puedan tener los rollos, la producción general es menor. Por lo tanto, se gasta una menor cantidad de recursos, generando así un impacto positivo no solo en la empresa en cuanto a reducir costos, sino también en la extracción de recursos y en las personas que compran los rollos sin defectos.

5.4 Impacto hacia los Objetivos de Desarrollo Sostenible.

El poder predecir los posibles defectos que hayan en los rollos que se producen en Ternium impacta principalmente en estos ODS:

8 TRABAJO DECENTE Y CRECIMIENTO ECONÓMICO



Objetivo 8: Promover el crecimiento económico inclusivo y sostenible, el empleo y el trabajo decente para todos.

Este proyecto se vincula con el objetivo 8,

ya que al tener un mejor manejo de recursos,

Se promueve el crecimiento económico de la empresa.

9 INDUSTRIA, INNOVACIÓN E INFRAESTRUCTURA



Objetivo 9: Construir infraestructuras resilientes, promover la industrialización sostenible y fomentar la innovación.

Este proyecto se vincula con el objetivo 9,

ya que se están buscando formas

de que la empresa pueda tener un uso más eficiente de sus

recursos para promover una industria más sustentable e inteligente.

12 PRODUCCIÓN Y CONSUMO RESPONSABLES



Objetivo 12: Garantizar modalidades de consumo y producción sostenibles.

Este proyecto se vincula con el objetivo 12, ya que se están buscando

formas de que la empresa produzca sus rollos de

una manera óptima y responsable prediciendo e identificando errores.

6. Despliegue

6.1 Descripción del prototipo funcional

Después de haber elegido el modelo de predicción, el cuál fue el modelo “Decision Tree Classifier”, se procede a crear el prototipo de la aplicación web para crear predicciones de los defectos.

Esta aplicación consiste en subir un archivo csv que cargue las variables predictoras y luego, con base en ellas, crear una predicción para cada fila. Esto se observa en la siguiente imagen:

Modelo de clasificación para predicción de defectos

Arrastra un archivo o [selecciona uno](#)

Nombre del archivo:
testdf2.csv

Index	Cliente	Norma	Brado	Código	Aclarado	Brado	Acero	Molino	Cara	Garanitizada	NUESTRA	PROVEEDOR	C_CLASE_PRODUCTO	SUPERFICIE	PRAN	C_CLASE_PRODUCTO_1	TIPO_RECUBRIMIENTO	Peso	Báscula Entrada	Despuete Entrada	Entrada	Tanque	Velocidad	Promedio	Min	Velocidad	Ocupap
0	0	0	39		1		19		2	2564	6	0	1	3	1	3	1	0	25369	6590	3277	2492	273	76.082617982934			
1	0	0	39		1		19		2	2598	6	0	1	3	1	3	1	0	24838	6590	3272	2482	218	76.082617982934			

Predicciones:

	0	627
0	627	
1	627	

Como se puede apreciar en la imagen, se carga un archivo csv con el nombre testdf2, el cual contiene todas las variables que serán utilizadas para crear la predicción. Asimismo, se aprecia como en la parte de abajo existe una tabla llamada “Predicciones”. Esta tabla hace referencia a la predicción de defectos por cada fila.

7. Recomendaciones

7.1 Recomendaciones al negocio

Que la empresa tenga un enfoque más ecológico, implementación de las ODS como la número 7 (Energía asequible y no contaminante), 11 (Comunidades sostenibles), 13 (Acción por el clima) y 17 (Alianzas para lograr los objetivos).

Asimismo, se recomienda poner especial énfasis en los defectos con código 2 y 627, que hacen referencia a los defectos DEF_00007 y DEF_00051 respectivamente, ya que en el algoritmo de predicción son los que más predominan. Esto se puede apreciar de mejor manera en la siguiente imagen:

```
[ ] 1 print(pred_df.groupby("Defectos").size())

  Defectos
  0        2593
  2        262
  4         1
  8         63
  19        18
  22         3
  25        39
  36         1
  40         1
  94        87
  101       10
  111       28
  115         6
  224       12
  304       10
  308       72
  414         1
  576         2
  593         6
  627      123
  665         1
  674       13
  705       11
  1069        9
  1096       44
  1171       23
  1345       52
  1346         1
  dtype: int64
```

7.2 Recomendaciones técnicas

Se recomienda crear un diccionario con la descripción de cada variable para una mejor comprensión de los datos y así facilitar la interpretación de los mismos, una vez que ya se tiene el modelo entrenado.

8. Siguientes pasos

Ya se ha creado un modelo de predicción para los defectos en la empresa. Debido al tiempo, este modelo no ha podido ser perfeccionado. Sin embargo, el siguiente paso a tomar será buscar crear un algoritmo que busque la forma de manipular las variables de tal manera que el modelo de predicción proyecte en su mayoría o totalidad, los rollos que no tengan ningún defecto.

9. Fuentes bibliográficas en formato APA.

- [1] Ayvaz, S., & Alpay, K. (2021). Predictive maintenance system for production lines in manufacturing: A machine learning approach using IoT data in real-time. *Expert Systems with Applications*, 173, 114598. <https://doi.org/10.1016/j.eswa.2021.114598>
- [2] Hazen, B. T., Boone, C. A., Ezell, J. D., & Jones-Farmer, L. A. (2014). Data quality for data science, predictive analytics, and big data in supply chain management: An introduction to the problem and suggestions for research and applications. *International Journal of Production Economics*, 154, 72-80.
- [3] Kang, Z., Catal, C., & Tekinerdogan, B. (2022). Product failure detection for production lines using a data-driven model. *Expert Systems with Applications*, 202, 117398. <https://doi.org/10.1016/j.eswa.2022.117398>
- [4] Qin, S. J., & Chiang, L. H. (2019). Advances and opportunities in machine learning for process data analytics. *Computers & Chemical Engineering*, 126, 465-473. <https://doi.org/10.1016/j.compchemeng.2019.04.003>
- [5] Ternium. (s. f.). *Ternium*. Recuperado 27 de mayo de 2022, de <https://mx.ternium.com/es>
- [6] Wirth, R., & Hipp, J. (2000, April). CRISP-DM: Towards a standard process model for data mining. In *Proceedings of the 4th international conference on the practical applications of knowledge discovery and data mining* (Vol. 1, pp. 29-40).