Tecnológico de Monterrey

Campus Querétaro

Bloque:

Inteligencia artificial avanzada para la ciencia de datos II (Gpo 501)

Proyecto:

Cruise Prediction

Colaboradores:

Fermín Méndez García A01703366

Emiliano Vásquez Olea A01707035

Diego Emilio Barrera Hdz. A01366802

Karen Cebreros López A01704254

José Ángel García López A01275108

Fecha:

23/10/2023

Introducción

El presente documento integra los puntos y entregables principales para la segunda fase del modelo CRISP-DM: Data Understanding. En esta fase del entendimiento de los datos se abarca el proceso de adquisición, consulta, análisis y revisión de los datos, estas diferentes etapas se separan en cuatro secciones principales: Recolección de datos; descripción de los datos; exploración y planteamiento de hipótesis; y finalmente, el análisis de calidad de los datos. Esta fase del proyecto permite que el equipo esté seguro de que los objetivos pueden ser alcanzados con el conjunto de datos al que se tiene acceso, además de contar con la información relevante al pasar a otras etapas en el manejo de datos.

Recolección de datos

Información del dataset

El conjunto de datos utilizado dentro de este proyecto es proporcionado directamente por General Electric (GE) para alcanzar los objetivos de minería de datos. El dataset incluye información de vuelos, como será descrito más adelante, que fueron generados por una herramienta en GE para simular las condiciones de una aeronave en distintos momentos de vuelo. Estos datos simulados se asemejan a recorridos reales de aviones y cuentan con una clasificación correcta con respecto al estado de estabilidad, que es el componente que queremos describir.

Los datos en formato .csv son accesibles únicamente a través de los dispositivos proporcionados por la empresa, por lo que la carga y manejo de datos es realizado directamente en estos equipos de cómputo. Esta limitación fué uno de los bloqueos en el acceso a los datos por parte del equipo, debido a que fué necesario cubrir ciertos requisitos, descargas y obtención de credenciales para utilizar las herramientas en el equipo de computo. Este problema fué resuelto siguiendo la configuración inicial de los equipos a lo largo de un par de semanas.

Herramientas de carga de datos

Como ya se mencionó antes, se trabaja de lado de GE, quienes nos dieron la facilidad de proporcionarnos equipos y diferentes facilidades para la elección e instalación de software que se usará para la solución del reto.

Hardware a utilizar

Dispositivo	Modelo	Características
Laptop	Laptop Dell	**Sistema operativo**: Windows Enterprise **Procesador**: Intel Core I7 **GPU 0**: Intel(R) UHD Graphics **GPU 1**: NVIDIA T1200 Laptop GPU **Ram**: 32 GB **Almacenamiento**: 2T

Dentro del equipo de trabajo consideramos que tenemos una configuración de hardware adecuada, pero es importante analizar todo tipo de características para poder tener planes de contingencia que se enfrenten a cualquier situación o complicación.

Ventajas:

La RAM que tenemos es beneficiosa para manejar grandes conjuntos de datos, nos permitirá procesarlos con cierta facilidad. Por otro lado, el almacenamiento es suficiente para poder trabajar los 160 GB de datos y aún así poder crear dummies si la situación lo amerita. También, tenemos un procesador de alto rendimiento que nos ayudará para un buen procesamiento de datos. Por último, se cuenta con una GPU que nos puede ayudar para acelerar los entrenamientos de los modelos y así tener más tiempo para realizar diferentes alternativas.

Desventajas:

En nuestra computadora tenemos una GPU integrada, la cual nos puede marcar algunas limitaciones para el procesamiento intensivo de gráficos o visualización de datos. Por otro lado, es importante aceptar que Linux es un sistema operativo que preferentemente se utiliza para proyectos de ciencia de datos gracias a su flexibilidad y control sobre su entorno. Por último, se debe tener cuidado con la forma en la que procesamos y manejamos los datos, pues nuestra GPU dedicada puede quedarse corta al querer hacer estas actividades.

Software a utilizar

1. **WinPy (3.11.5):**
    a. ¿Por qué?

    Gracias a que la instalación de los programas es limitada debido a los permisos y diversas situaciones que han surgido para la descarga de los mismos, se optó por la instalación de WinPy, ya que, es una alternativa que nos ofrece diversos programas básicos que se pueden utilizar para la solución del proyecto, así como, Pandas, Jupyter y Pip.

    b. ¿Cuándo se usaría?

    Debido a la naturaleza del proyecto, se estará utilizando en todas las etapas empezando por “Data understanding”.

    c. Alternativas:
        i. Data Science Anaconda

2. **Data Science Anaconda (3.9.7):**
    a. ¿Por qué?

    Al igual que WinPy, es una alternativa que se planea utilizar debido a las limitaciones que tenemos, esto debido a que Anaconda es un gestor de paquetes completos que nos sirven para la solución del reto y, a diferencia de WinPy, este incluye PySpark.

    b. ¿Cuándo se usaría?

    Debido a la naturaleza del proyecto, se estará utilizando en todas las etapas empezando por “Data understanding”.

    c. Alternativas:
        i. WinPy

3. **Jupyter (7.0.4):**
    a. ¿Por qué?

    Debido a su entorno interactivo para desarrollar y colaborar en modelos de ciencia de datos. Facilita análisis, visualización de datos, y comunicación efectiva, es compatible con varios lenguajes y permite informes integrales.

    b. ¿Cuándo se usaría?

Al igual que las tecnologías pasadas, se estará usando desde la etapa de “Data understanding”.

c. Alternativas:
    i. Visual Studio, gracias a que Google Colab se usaría con internet.

4. Spark (3.5.0):

a. ¿Por qué?

El framework Apache Spark, junto con su librería para Python PySpark, son herramientas que permiten el análisis y procesamiento de datos. Spark brinda la posibilidad de manejar datos a gran escala.

Como es mencionado más adelante, en el apartado de Enfoque de Manejo de Datos, algunos de los beneficios que provee Spark para el manejo de datos no son accesibles por las restricciones de tecnología en el proyecto.

b. ¿Cuándo se usaría?

Se iniciará la etapa de data understanding.

c. Alternativa:
    i. Pandas

5. MongoDB (7.0.2):

a. ¿Por qué?

MongoDB es bueno en soluciones junto con Spark para almacenar datos no estructurados. Esto permite el análisis rápido del conjunto completo de datos gracias a su flexibilidad y escalabilidad, simplificando la integración con Spark.

b. ¿Cuándo se usaría?

Se iniciará la etapa de data understanding.

c. Alternativas:
    i. SQL
    ii. Leer los datos directos de los CSV sin almacenarlos

6. Pandas (2.0.0):

a. ¿Por qué?

Pandas serviría para un proyecto de 160 GB de datos. Ofrece facilidad de uso y herramientas para análisis y manipulación de datos, siendo útil para

tareas de preprocesamiento y exploración de datos; acciones que serán importantes realizar para la solución del proyecto.

b. ¿Cuándo se usará?

   Se inicia su uso en la etapa de data understanding.

c. Alternativas
   i. Spark

Enfoque de manejo de datos

Como se describe más adelante, el conjunto de datos a utilizar en este proyecto está distribuido en una serie de archivos, sumando a un total de más de 160 GB de información. Tomando en cuenta esto, el equipo de trabajo ha decidido no tomar un enfoque de Big Data para el manejo de datos en el proyecto, principalmente debido a que esta información no presenta un formato “complejo”. Al tener un formato tabular, junto con tipos de datos limitados a valores numéricos y binarios, no será necesario utilizar herramientas o métodos más sofisticados para su manejo. Además, las restricciones de tecnologías utilizables dentro de los equipos de cómputo no permiten el uso de técnicas como el cómputo distribuido con la herramienta Spark, que formaría parte de un enfoque en Big Data.

A pesar de estas restricciones, de igual manera se tiene contemplado utilizar Spark junto con un sistema de bases de datos no relacionales, siempre y cuando las restricciones de instalación de tecnologías en los equipos de cómputo lo permitan. Estas herramientas además de ser opciones para el manejo de datos, apoyan al factor académico del desarrollo de este proyecto, ya que se aplican tecnologías aptas para el trabajo con Big Data.

Descripción de los datos

Propiedades de los datos

Los registros a los que se tiene acceso son datos tabulados, es decir, cada atributo o variable es organizado en un formato de columnas, con cada instancia o registro individual representado por una fila. Los datos son almacenados en una serie de archivos en formato CSV, con una cantidad total de 2718 documentos diferentes, donde cada uno de ellos representa la simulación de un viaje realizado por una aeronave.

NOTA: Las cifras de almacenamiento y cantidad de archivos que son mencionadas en este documento hacen referencia a la última consulta realizada por el equipo. Es posible que algunos detalles o cantidad de datos sufran ciertos cambios en actualizaciones más recientes, los cuales serán tomados en cuenta para los siguientes avances.

Cada archivo CSV está compuesto por 37 columnas distintas, que indican diferentes condiciones en un instante del vuelo. La mayoría de estos componentes se encuentran

anonimizados, por lo que se cuenta con los valores pero no con una descripción de qué es lo que significan para la aeronave, esto es una de las restricciones sobre el acceso a la información que tiene el equipo. Los dos atributos que si son etiquetados es el instante de tiempo (“time”) en segundos, y el valor binario “stableCruise_boolean”. La descripción o nombre de atributo de cada columna sí cuenta con el tipo de dato de la columna dentro de su mismo nombre, por ejemplo, el nombre de columna col_14_float, indica que en la columna son almacenados valores de tipo flotante.

Una última característica importante del conjunto de datos es la redundancia, ya que para cada una de las variables (excepto el tiempo), están registradas dos columnas aparentemente idénticas con los datos. Esto representa que se tomaron dos registros para un mismo atributo como método para manejar cualquier posible fallo o inconsistencia en la captura de datos. Se ve reflejada esta redundancia en los datos tabulados debido a que columnas adyacentes, por pares, cuentan con la misma, o muy similar, información (por ejemplo, col18_boolean es igual a la columna col19_boolean).

Viabilidad de los datos

Para comprobar la viabilidad de los datos se plantea realizar un script en python que nos permita clasificar los datos en útiles y no útiles, ya que en palabras de socio formador, hay archivos que no reportan estabilidad en todo el transcurso del la mision de vuelo, por lo que en primera instancia se realiza un filtro que reconoce si hay o no presencia de “1” en la columna objetivo “stableCruise_boolean”, esto debido a que si esta lleno de ceros, podría afectar al modelo.

En adición a lo anterior es necesario comparar ambas columnas de los reportes para detectar anomalías entre las lecturas para saber si son significativas y con eso saber si hay que aplicar alguna técnica que las elimine ya que sería un desperdicio de recursos usar información repetida.

Exploración y planteamiento de hipótesis

Consulta inicial

En primera instancia el dataset que se nos proporcionó consta de 169 GB de información los cuales vienen distribuidos en más de 2700 archivos .csv. Estos archivos reportan vuelos diferentes, dichos reportes son un conjunto de información arrojada por el avión durante toda la misión. Sin embargo, por cuestiones de privacidad no se nos revela el significado de cada una de las columnas, aun con eso y basado en un diagnóstico inicial, pudimos concluir que el reporte posee redundancia ya que cada columna posee un duplicado el cual funge como un respaldo. Con todo esto, algunas de las columnas que pudimos identificar basado en mera observación son la de tiempo la cual es la primera de todas y la única que no presenta un respaldo ya que es única en todos los archivos y reporta en segundos. También identificamos la columna que reporta la altitud y lo que parece ser latitud y longitud, estos últimos son mera especulación ya hasta el momento solo hemos podido observar algunos ejemplos de csv.

NOTA: Según el calendario, el 6 de octubre teníamos pronosticado tener acceso a los datos y poder manipularlos. Hasta el 23 de Octubre el equipo solo ha tenido disponible los datos en dos sesiones. El socio formador no pudo acudir por temas médicos y queda por confirmar si las gráficas generadas pueden ser plasmadas en este reporte. Por ese motivo, cada gráfica se va a mantener con los datos de prueba y será actualizada cuando tengamos el permiso del socio formador.

Las siguientes secciones son el plan de trabajo, así como scripts que simulan los datos de acuerdo a la información que pudimos recabar estas semanas. Además, como equipo, tomamos la decisión de reducir la cantidad de tecnologías para aminorar el riesgo de que no se puedan instalar. El análisis que se muestra a continuación está escrito con Python y pandas principalmente.

Link de repositorio: https://github.com/FerminMendez/DataAnalitics_Reto2

Sobre los datos de prueba:
Hemos confirmado que la estructura de los datos es la misma para los datos que nos proporciona la empresa.

Número de archivos dimensiones:Hemos desarrollado un script para configurar los datos de prueba. Se realizaron las pruebas con 500 archivos que tienen entre 1600 y 2400 columnas. Está configurado para que sean alrededor del 20% de las columnas dadas.

```
TEST_PATH = os.path.join(BASE_PATH, 'test_data')
NUM_TEST_FILES = 500
SIZE_TEST_FILES = 2000
```

Tomando como referencia uno de los archivos definimos las siguientes columnas:

HEADER = ['time', 'col1_boolean', 'col2_boolean', 'col3_boolean', 'col4_boolean', 'col5_float', 'col6_float', 'col7_float', 'col8_float', 'col9_float', 'col10_float', 'col11_float', 'col12_float', 'col13_float', 'col14_float', 'col15_float', 'col16_float', 'col17_float', 'col18_float', 'col19_float', 'col20_float', 'col21_float', 'col22_float', 'col23_float', 'col24_float', 'col25_float', 'col26_float', 'col27_float', 'col28_float', 'bool_stable_cruise', ', 'col31_integer', 'col32_integer', 'col33_integer', 'col34_integer', 'col35_integer', 'col36_integer']

Descripción	Número de columnas	Tipo de dato	Forma en que se llena el dato de prueba
Tiempo	1	Float	El segundo n corresponde a la fila n
Atributos ocultos	4	Boolean	Booleano aleatorio por registro.
Columnas desconocidas	24	Float	Flotante aleatorio por cada registro, se genera con media de 5 y desviación estándar de 1.
Stable cruise	2	Boolean	Por cada archivo hay un 50% de posibilidades que solo existan 0's en esa columna. El otro 50% toma un booleano aleatorio en cada uno de sus registros.
Columnas desconocidas	6	Integer	Entero aleatorio por cada registro, se genera con media de 0 y desviación estándar de 1.

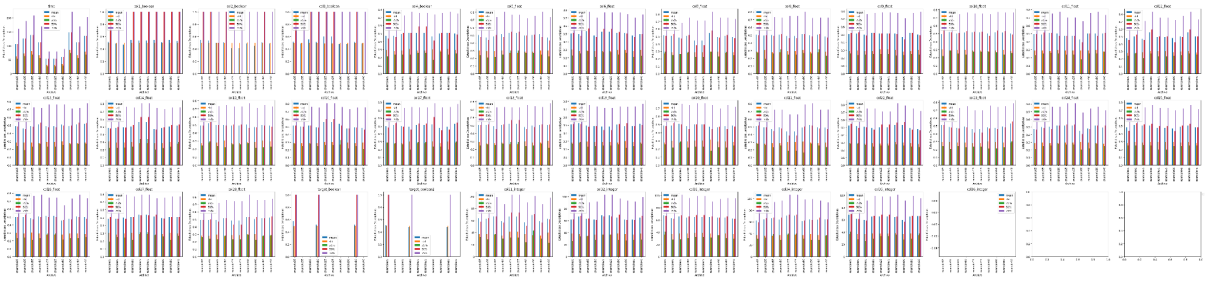
Estadística descriptiva

Para obtener información a partir de la estadística descriptiva, se creó un script que utiliza los datos de prueba mencionados anteriormente para hacer un data frame multi-indexado con la estadísticas de cada archivo. El objetivo de esta información es poder sacar las medidas de tendencia central de cada columna de cada archivo, que ayudarán a tener un mayor entendimiento de los datos.

		count	mean	std	min	25%	50%	75%	max
file	variable								
example0	time	248.0	123.500000	71.735626	0.000000	61.750000	123.500000	185.250000	247.000000
	col1_boolean	248.0	0.556452	0.497808	0.000000	0.000000	1.000000	1.000000	1.000000
	col2_boolean	248.0	0.451613	0.498660	0.000000	0.000000	0.000000	1.000000	1.000000
	col3_boolean	248.0	0.500000	0.501011	0.000000	0.000000	0.500000	1.000000	1.000000
	col4_boolean	248.0	0.511640	0.281637	0.000497	0.260108	0.524429	0.753178	0.995118
...	...	...	...	...	...	...	...	...	...
example90	col32_integer	296.0	61.804054	37.236424	0.000000	27.000000	61.000000	93.000000	127.000000
	col33_integer	296.0	64.003378	35.460601	1.000000	35.000000	63.500000	94.000000	127.000000
	col34_integer	296.0	64.287162	38.722857	0.000000	28.750000	63.500000	99.250000	127.000000
	col35_integer	296.0	59.692568	37.112062	0.000000	27.750000	57.000000	91.000000	127.000000
	col36_integer	296.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000

370 rows × 8 columns

En la imagen de arriba podemos ver el data frame creado de la estadística descriptiva con ciertos archivos (dummies) seleccionados.



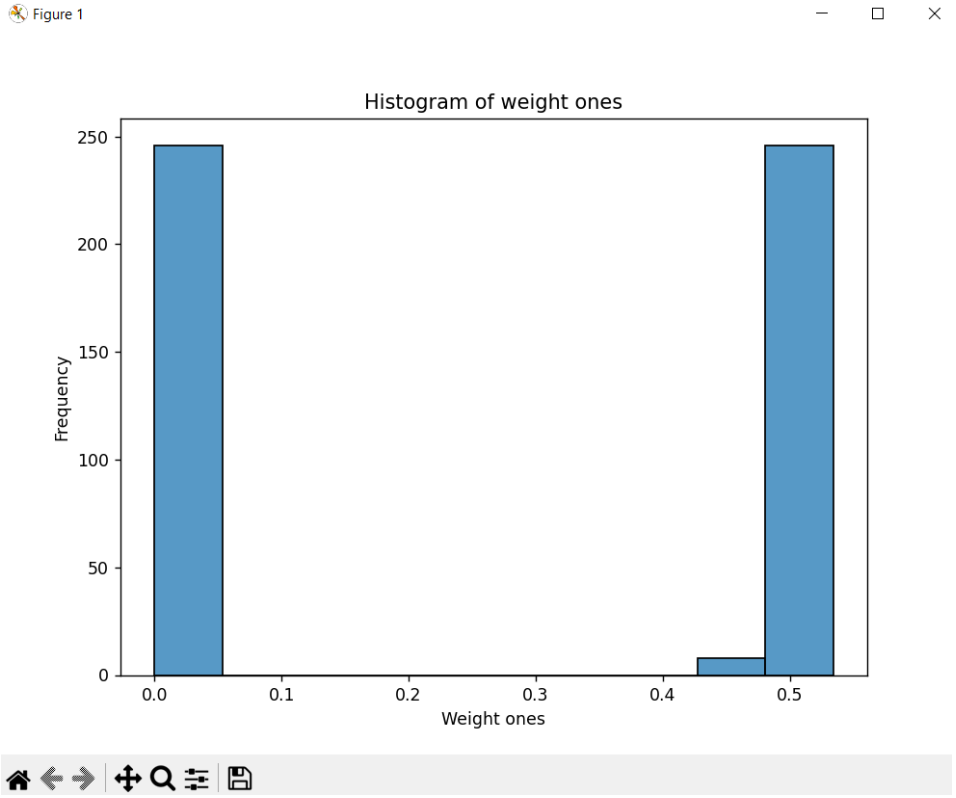
En las gráficas anteriores podemos observar aproximadamente 37 gráficas que contienen parte de la estadística descriptiva de cada columna.
Lo que se busca comparar en estas gráficas es básicamente el comportamiento de los vuelos por tamaño; pues dependiendo del tiempo y la media de este, se puede saber si es corto, mediano o largo y a partir de ahí, observar datos específicos como su desviación estándar y comportamiento en cuanto al valor típico y los percentiles.

Gráficos relevantes

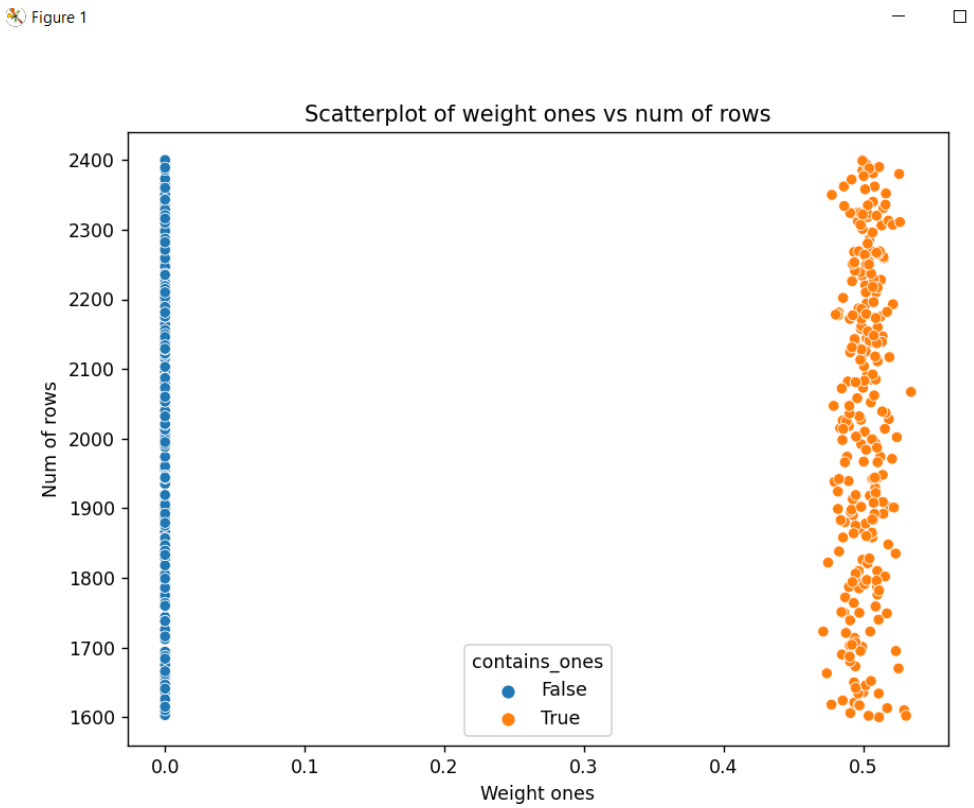
Como hemos comentado anteriormente, en esta sección solo mostramos el tipo de gráfica con los datos de prueba. Pero no representan datos reales. Aunque las interpretaciones sí son correctas.

A continuación enumeramos los gráficos que obtuvimos de los datos de prueba generados. Así como las hipótesis que tenemos y qué vamos a hacer con la información.

Histograma de unos ponderados.
Por cada archivo muestra el porcentaje de unos que contiene respecto al total de registros. Los “unos” buscados corresponden a la variable que queremos predecir. Stable_cruise
(GRÁFICA CON DATOS DE PRUEBA)



En esta gráfica comparamos el número de columnas, con la cantidad de unos y si contiene o no algún ejemplo de “stable_cruise”.



Preguntas a las que queremos dar respuesta.

¿Cuál es la proporción de archivos que sirven como ejemplo para predecir “Stable_cruise”? ¿Es conveniente tomar todos los archivos? ¿La cantidad de archivos que no contienen ningún 1 en Stable Cruise puede entorpecer el entrenamiento del modelo? ¿Hay alguna relación entre el número de columnas y la posibilidad de tener algún ejemplo de 1 en stable cruise?

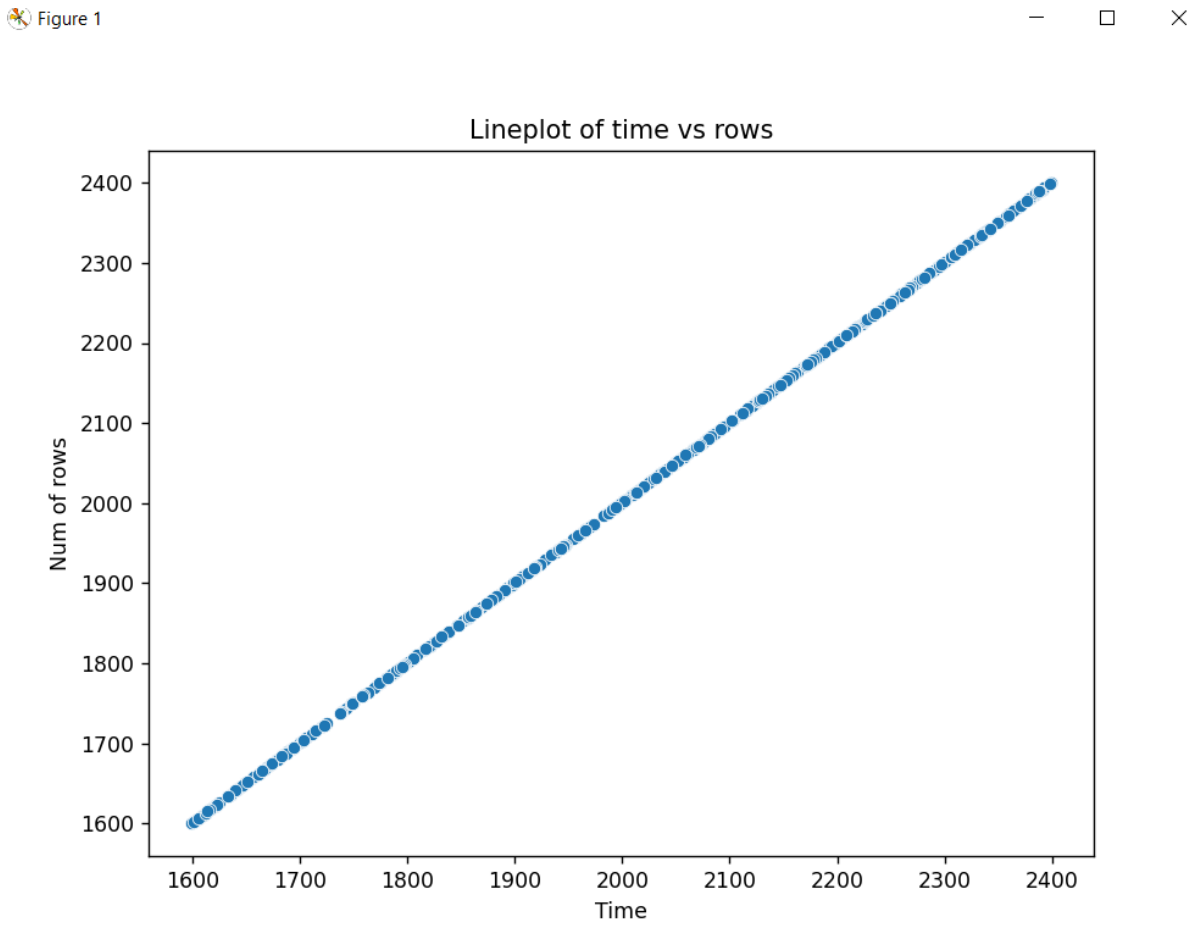
Hipótesis

- Los archivos pequeños no tendrán unos. Con esto podríamos establecer un tamaño mínimo de los archivos y reducir el dataset.
- Así mismo, creemos que la cantidad de archivos con 1 estará alrededor de los 800 ejemplares, suficiente para hacer la predicción únicamente con esos archivos.

Resultados

- Encontramos que casi 2000 archivos NO contienen ningún ejemplo de Stable Cruise. Esto deja una proporcion 7:20 casi que es casi equivalente a 1:3. Es decir, por cada archivo con 1 hay 3 archivos sin un solo 1.
- Encontramos que hay una gran cantidad de archivos con 200,000 columnas tal que sus variaciones de porcentaje de unos varía desde 0.05% hasta casi el 2%.

Line plot de tiempo final comparado con el número de columnas



Preguntas a las que queremos dar respuesta.

¿La frecuencia con la que se toma un dato es constante en todos los archivos?

¿Existen archivos del mismo tamaño que representan dos vuelos con tiempos muy distintos?

Hipótesis

- Creemos que la distribución del número de columnas con el tiempo será una relación lineal.

- Archivos del mismo tamaño representan vuelos del mismo tiempo

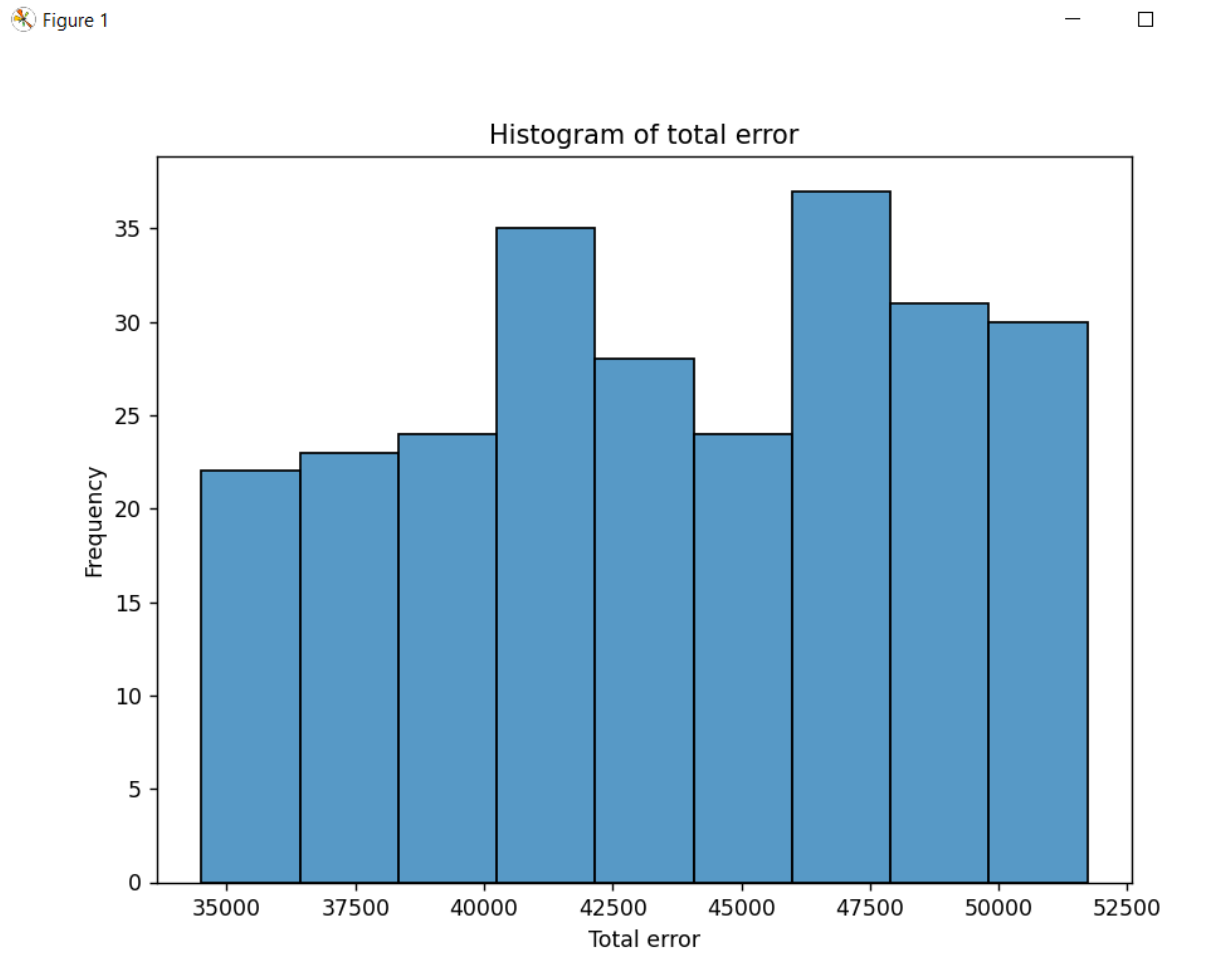
Resultados.

- El time step es consistente entre el numerosos de columnas y cada registro.

Gráficas sobre la redundancia de datos

Sabemos que las columnas por pares son obtenidas por dos dispositivos corriendo el mismo código en las mismas condiciones. A esto se le llama redundancia y es común en la industria aeronáutica para reducir los riesgos.

En este caso: bool_col1, bool_col2 ó int_col5, int_col6 son pares de columnas redundantes. Recuperamos la diferencia entre cada par redundante y lo graficamos.



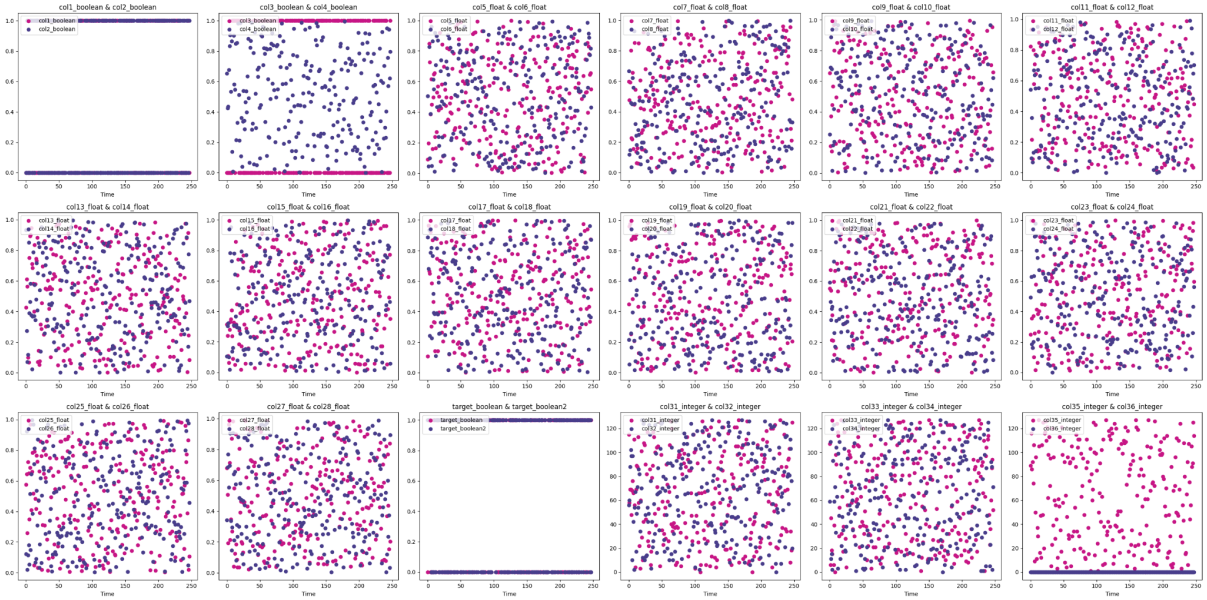
Preguntas a las que queremos dar respuesta.

¿Existen diferencias significativas entre los datos registrados con redundancia?

Hipótesis

- No existe diferencia entre los datos con redundancia, por esto podemos eliminar una columna y reducir el tamaño del dataset a la mitad.

Gráfica para la visualización de la redundancia de datos por cada par de columnas.
Como se mencionó en el punto anterior, por cada par de columnas hay cierta redundancia en los datos que son obtenidos por dos dispositivos corriendo el mismo código en las mismas condiciones.
En este caso graficamos cada par de columnas por separado de un archivo con datos dummies, meramente para poder visualizar la redundancia de éstas, además de poder obtenerla de forma numérica.



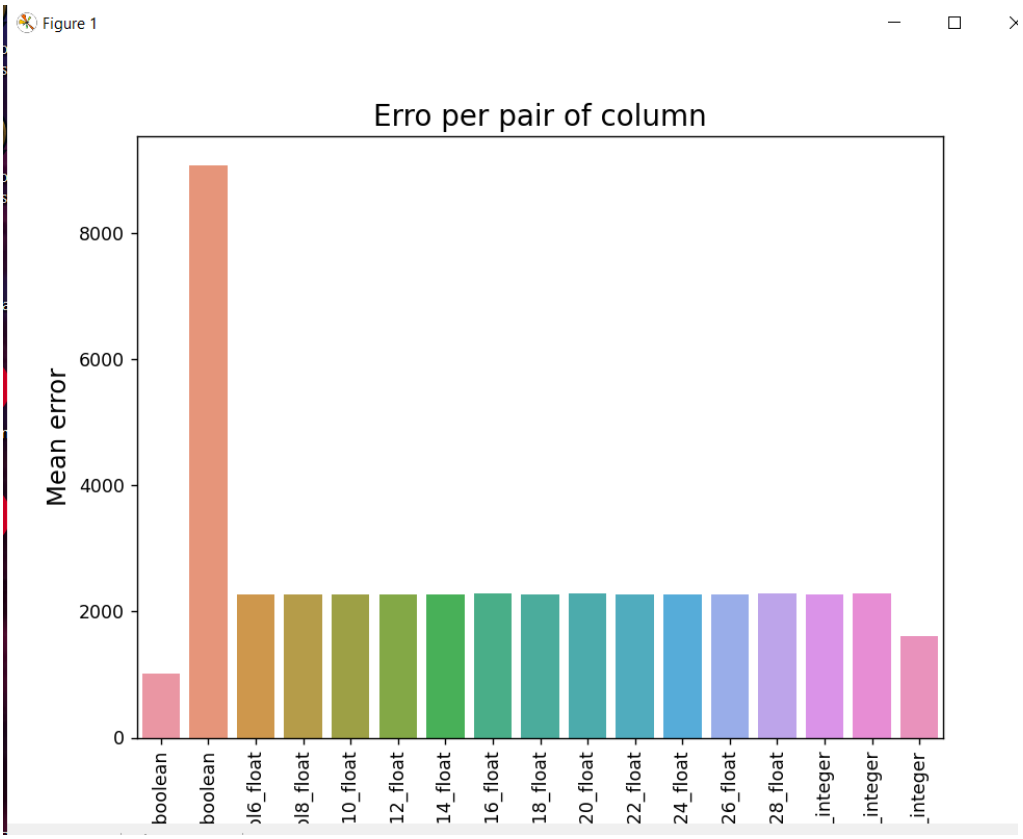
Cabe aclarar que como para esto utilizamos datos dummies, estas gráficas aún no están representando lo que queremos de los datos reales.

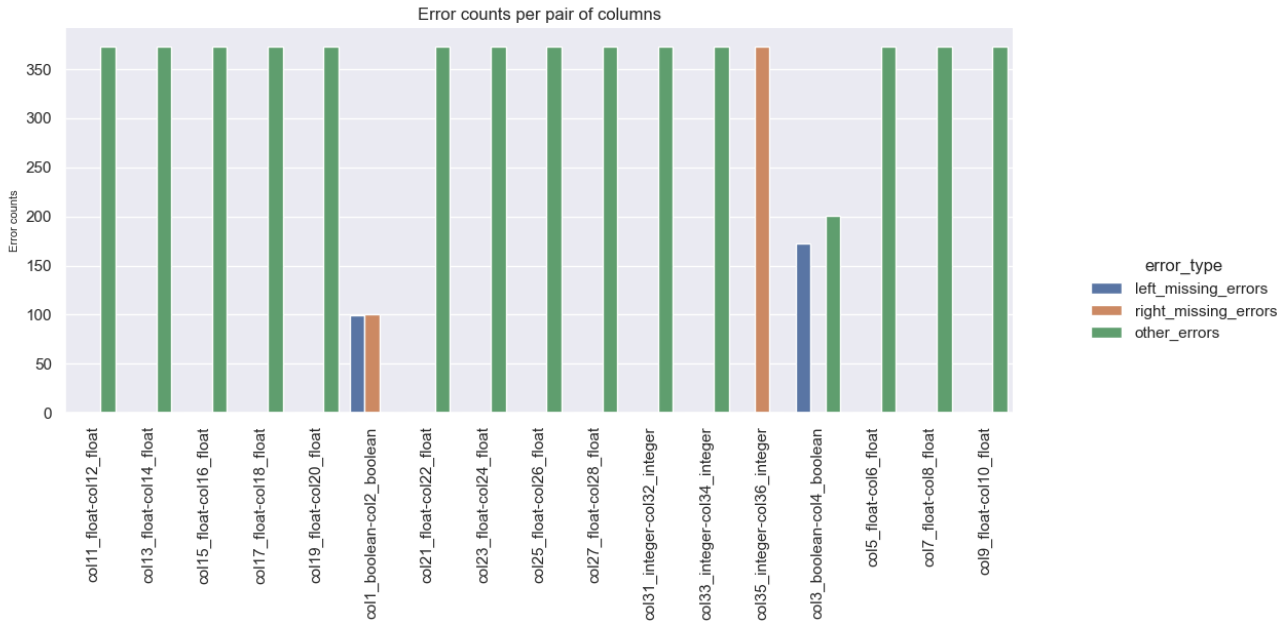
Preguntas a las que queremos dar respuesta.
¿Cómo se visualizan las diferencias de los datos registrados con redundancia por cada par de columnas?

Hipótesis

- No existe diferencia entre los datos con redundancia, por esto podemos eliminar una columna y reducir el tamaño del dataset a la mitad.

Gráficas sobre la redundancia de datos por columna





Preguntas a las que queremos dar respuesta.
¿Qué columnas presentan más errores entre los valores redundantes?
¿Qué comportamiento se presenta en los errores de cada variable? ¿Se deben a registros faltantes o medidas ligeramente desviadas?

Hipótesis

- Solo en caso de que existan errores significativos tendrá sentido indagar qué columnas presentan errores.

Conjuntos de prueba y entrenamiento de los datos

Para llevar a cabo la evaluación de los modelos de aprendizaje de máquina que serán generados en etapas posteriores, se utilizará el método K-Fold Cross Validation. De forma específica, se utilizarán 10 Folds en este proceso (10-Fold Cross Validation), debido a que es un calor típico utilizado para este parámetro, sin embargo, puede variar la cantidad a partir de la cantidad de datos y el equipo de cómputo al que se tiene acceso. Esto quiere decir que el conjunto de datos será dividido en 10 subconjuntos del mismo tamaño, cada uno de estos grupos seguirá un proceso que involucra separarlo del conjunto total, llevar a cabo el entrenamiento del modelo con el resto de los datos y posteriormente probar el desempeño del modelo con el conjunto separado inicialmente.

Este proceso se lleva a cabo con cada uno de estos Folds o subconjuntos, almacenando información sobre el desempeño del modelo al utilizar grupos variados para validación y entrenamiento. Esta técnica permite evaluar de forma confiable los diferentes modelos que se generen, permitiendo ajustar parámetros o comparar el desempeño con otros modelos.

Existen diferentes herramientas que facilitan a implementación de la técnica de K-Fold Cross Validation:

- En caso de utilizar PySpark, se puede utilizar la clase “CrossValidator”, importada directamente del módulo “pyspark.ml.tuning”, donde se pueden especificar el número de Folds, el modelo o estimador y parámetros a tomar en cuenta.
- Por otro lado, la librería scikit-learn en Python también permite implementar este proceso con la clase “cross_validate”, desde “sklearn.model_selection”. De igual forma se definen parámetros como el número de Folds y el modelo.

Una alternativa para separar los conjuntos de entrenamiento, prueba y validación del dataset es realizar una sola división del conjunto de datos general. Primero se toman los valores para el conjunto de prueba y entrenamiento, una posibilidad es tomar un 85% de los datos para entrenamiento y el otro 15% para las pruebas. Después, del conjunto de entrenamiento se puede separar otro subconjunto de 15% para la validación del modelo, terminando con una división del 70% para entrenamiento, 15% para validación y 15% para las pruebas.

Búsqueda de patrones

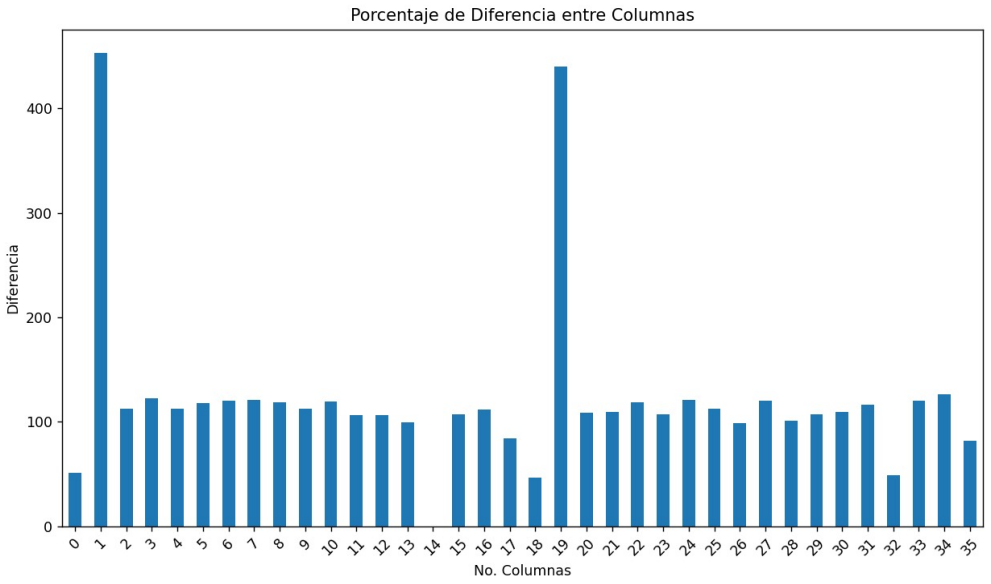
Ya tenemos disponible un dataframe con la estadistica descriptiva de cada columna de cada archivo. Además podemos clasificar los archivos por tamaño y cantidad de unos. Con vamos a tomar muestras de cada tipo de archivo y graficar la media y desviación estándar para encontrar patrones de acuerdo a la columna y al tipo de archivo que corresponda. Además graficaremos cada columna a lo largo del tiempo para identificar el comportamiento de cada columna

Inconsistencias

Para evaluar las inconsistencias dentro de los datos que se nos entregaron, decidimos realizar un análisis archivo por archivo para evaluar la continuidad y consistencia de los mismos, para esto, creamos una función que itera sobre cada par de columnas en todos los archivos dentro de un fichero establecido.

El código itera a través de las columnas del DataFrame brindado, en pares. Luego, calcula la diferencia absoluta entre estas dos columnas y determina el porcentaje de diferencia, que representa cuánto difieren estas columnas en relación al total de filas en el DataFrame. Los porcentajes de diferencia se almacenan en el diccionario de diferencias utilizando una clave que identifica el par de columnas.

Finalmente, se crea un nuevo DataFrame llamado resultados_df a partir del diccionario diferencias. Este nuevo DataFrame tiene dos columnas: una que muestra las claves que representan los pares y otra que muestra los porcentajes de diferencia entre ellas. El resultado de la función es este nuevo DataFrame, que proporciona un resumen de las discrepancias entre los pares de columnas en el DataFrame de entrada.



Tras aplicar el código descrito anteriormente en la carpeta donde se alojan los datos brindados, se descubrió que hay una cantidad exagerada de irregularidades en las columnas 1 y 29 mientras que las demás presenta un total que ronda las 100 diferencias exceptuando casos como en la 14, 18 y 32 donde es sumamente bajo el porcentaje, derivado de este análisis, es claro que la mayoría no aparenta ser sumamente representativa, sin embargo aquellas que sí deben ser tratadas ya que es probable que afecte de forma negativa una vez en la fase donde se prueban los modelos.

Redundancia

Uno de los principales objetivos de analizar la redundancia en los pares de columnas de datos es decidir la forma en que se integrarían ambos registros o incluso si es conveniente eliminar ciertas columnas para las etapas de preparación y modelado. Para esto tomamos en cuenta los errores, que hacen referencia a la diferencia entre el par de columnas que corresponde a un mismo dato, ya que, como fué mencionado anteriormente, cada registro cuenta con la toma de dos orígenes distintos.

Primero, se utilizan los gráficos de la media del error y conteo de errores para identificar si esta desviación entre los registros es realmente significativa, en caso de ser pocos errores, o cambios mínimos que no afecten en el registro de la variable de estudio (booleano de Stable Cruise), se puede tomar la decisión eliminar una de las columnas en cada par.

Además de esto, para el conteo de errores se realiza una división entre tres tipos de errores distintos para analizar el comportamiento de la desviación entre columnas, los errores debido a que una u otra de las columnas tenga un valor faltante (o valor de 0) y los errores por una desviación real entre los registros. Esta división permite revisar, para cada par de columnas, de qué forma están distribuidos los errores entre sus datos. En caso de que la mayoría de los errores sean causados por la falta de registros o anomalías en una de las columnas, es posible llegar a la decisión de únicamente mantener una columna del par, corrigiendo posteriormente el ruido o comportamientos inusuales. Por otro lado, si un mayor

porcentaje de los errores totales proviene de desviaciones entre los valores de un par de columnas, es necesario analizar el comportamiento de estas columnas de forma específica para decidir su manejo.

Análisis de calidad

Verificación de calidad

Respecto a la calidad de los datos, es seguro decir que son confiables, ya que se obtuvieron directamente del socio formador, mismo que expresó que no son completamente reales, ya que son datos sintéticos basados en misiones reales usados para el *testing* de sistemas. De igual forma y derivado el análisis inicial se encontraron inconsistencias en los archivos, mismos que concluyeron en un reformulado de datos por parte del socio formador, ya que algunos archivos dentro del dataset divergían con respecto a la cantidad de columnas reportados por lo que el socio se encargó de resolver dicho problema. Esta situación es retomada en el apartado de **Problemas y Soluciones**.

Con respecto a los errores dentro de los datos es preciso decir que las inconsistencias deben ser bajas y/o nulas, ya que un sistema aeronáutico debe ser virtualmente inmune al ruido o variaciones por cuestiones de seguridad, aun con eso, como se mencionó anteriormente, los datos presentan un duplicado de cada columna el cuál podemos usar para comparara y reparar anomalías que se puedan presentar sin tener que recurrir a técnicas como reemplazar, interpolar o eliminar en el peor de los casos.

En términos de completitud de datos, es difícil afirmar que existe una falta de datos en los archivos, como se mencionó anteriormente los sistemas aeronáuticos son sumamente preciso por lo que la probabilidad es baja, aun con eso consideramos pertinente realizar una revisión de registros atípicos y/o faltantes una vez que podamos manipular los datos libremente.

Retomando el tema de la redundancia, es claro que para el preprocesamiento de datos es de suma importancia ya que nos permite reparar y detectar anomalías. Sin embargo al momento de realizar el modelo no resulta indispensable incluirlo, ya que tal y como su nombre lo indica sería redundante, en este caso se tiene planeado comparar ambos reportes y analizar si existe una desviación significativa, y si vale la pena conservar el reporte redundante o no.

Problemas y soluciones

Una inconsistencia que fué detectada en las etapas iniciales de la consulta de datos fue una cantidad distinta de columnas o atributos en los archivos CSV. Al tener algunos archivos con ciertos atributos adicionales o faltantes con respecto a otros registros, sería necesario realizar algún tipo de estandarización para ajustar un modelo correctamente. Para este caso una solución formulada fué la siguiente:

- Calcular el número de archivos que presentan una cantidad de atributos distinta, o la distribución de estas cantidades en el dataset completo.
- A partir de esta información, junto con los resultados de la exploración de datos, decidir entre un filtro de archivos con inconsistencias o ajustar las columnas que son utilizadas para el análisis.
- En caso de eliminar atributos de un archivo CSV, es necesario asegurar que todas las columnas que formen parte del dataset final contengan los mismos registros (sean el mismo atributo). Esto quiere decir que se debe realizar una revisión para asegurar que las columnas sean consistentes.

A pesar de esto, el socio-formador por parte de General Electric de igual forma se percató de esta inconsistencia, actualizando el conjunto de datos que sería trabajado en el proyecto. Posterior a esta actualización ya no es necesario manejar esta situación.

Notas y adaptaciones de CRISP-DM

Para la fase de Data Understanding, el equipo de trabajo decidió sustituir las secciones de Hipótesis Inicial y Descubrimientos por apartados específicos dentro de otras secciones del documento. De igual forma se definen hipótesis, así como un resumen del análisis, pero esta información es integrada dentro de las secciones de **Consulta inicial** y **Búsqueda de patrones**.