Big Data: Análisis y predicción de datos en aerolíneas mediante MongoDB y MLlib

Ivan González Villanueva

Resumen– El proyecto busca analizar la correlación entre los diversos factores que afectan a la puntualidad de los vuelos en lo Estados Unidos. Se llevará a cabo la implementación de un sistema de análisis y predicción, que utiliza las tecnologías de procesamiento no tradicionales como Spark, permitiendo analizar grandes conjuntos de datos de manera eficiente. Mediante este análisis, se podrá crear un modelo de predicción de los retrasos en los vuelos. Para dicha tarea se utilizará un algoritmo de aprendizaje automático conocido como *GBT Regression*, entrenándolo a partir de datos históricos y climáticos. A partir del modelo de perdición obtenido, se buscarán patrones que permitirán mejorar la eficiencia de los vuelos, así como prever futuros fallos o problemas.

Palabras clave– Mineria de datos, Datos abiertos, Spark, Hadoop, Hive, HDFS, MondoDB, Aprendizaje automático, Aerolíneas, Meteorología.

Abstract– The project seeks to analyze the correlation among several factors which affect the puntuality of the flights in the USA. The implementation of an analysis and prediction system will be carried out. This system is using non-traditional processing technologias as Spark, which can analize large data sets in a very efficient way. Thanks to this analysis, a prediction model will be implemented to calculate future flight delays. An algoritm of automatic machine learning named GBT Regression will be used. Historical and climatic data will be used to train the model. With the resulting prediction model, some patterns will be searched in order to improved the flight efficiency as well as anticipate future problems.

Keywords– BigData, OpenData, Spark, Hadoop, Hive, HDFS, MondoDB, Machine learning, Airlines, Weather.

1 Introducción

La utilización del *big data* ha ayudado a los investigadores a realizar hallazgos que les habría costado años descubrir sin el uso de dicha tecnología. Un ejemplo relevante es el proyecto del genoma humano. Dicho estudio tiene como objetivo encontrar, secuenciar y elaborar mapas genéticos y físicos de gran resolución del ADN humano. Como resultado, se genera una gran cantidad de datos, del orden de 100 Gigabytes por persona, imposibilitando la utilización de técnicas de procesamiento de datos tradicionales.

Debido a éste y a otros proyectos que trabajaban con grandes volúmenes de datos, surgieron nuevas técnicas y tecnologías de procesamiento para poder continuar desempeñando el procesado de enormes cantidades de datos eficientemente.

Por tanto, entendemos por *big data* el conjunto de estrategias, tecnologías y sistemas para el almacenamiento, procesamiento, análisis y visualización de conjuntos de datos complejos.[6][7]

Cabe destacar la definición de las 3 ”V”del *big data*, introducida por Doug Laney en 2001, como el conjunto de técnicas y tecnologías para el tratamiento de datos, referentes a:

- **Volumen:** Disponibilidad de una gran cantidad de datos que necesitan ser procesados.
- **Velocidad:** Los flujos de datos continuos (*streaming data*) deben ser procesados en tiempo real, es decir, en tiempos muy cortos (del orden de segundos o milisegundos, dependiendo del problema concreto).
- **Variedad:** Más allá de los datos estructurados, aparece

• E-mail de contacto: ivan.gonzalez@e-campus.uab.cat
• Mención realizada: Tecnologías de la Información
• Trabajo tutorizado por: Jordi Casas Roma (Departamento de Ingeniería de la Información y de las Comunicaciones)
• Curso 2017/2018

la necesidad de trabajar con datos semi-estructurados y no estructurados (como por ejemplo, imágenes o documentos de texto).

Posteriormente, la compañía IBM introdujo una interacción adicional con la cuarta "V"[6]:

- **Veracidad:** Dar consistencia y seguridad a unos datos que muchas veces son incompletos o ambiguos.

De acuerdo con el *National Institute of Standards and Technology*[1] (NIST) existen tres tipos de escenarios que requieren el uso de *big data* [7]:

- **Tipo 1:** donde una estructura de datos no relacional es necesaria para el análisis de datos.
- **Tipo 2:** donde es necesario aplicar estrategias de escalabilidad horizontal para procesar y analizar de manera eficiente los datos.
- **Tipo 3:** donde es necesario procesar una estructura de datos no relacional mediante estrategias de escalabilidad horizontal para procesar y analizar de manera eficiente los datos.

2 Objetivos

Los retrasos y cancelaciones en los aeropuertos suponen un problema a la hora de planificar un viaje, pudiendo suponer una molestia para los pasajeros.

En este apartado se procederá a explicar los objetivos principales de este proyecto, los cuales son:

1. Implementar una arquitectura *big data* para almacenar y analizar grandes volúmenes de datos, utilizando herramientas de almacenamiento masivo y algoritmos de aprendizaje automático.

2. Responder a un conjunto de preguntas relacionadas con los vuelos, con el objetivo de determinar características básicas del conjunto de datos, como pueden ser:

- Número de aeropuertos existentes en los EEUU.
- Número de vuelos que se realizan cada año.
- Aeropuertos con más afluencia de pasajeros.
- Meses con mayor cantidad de vuelos.
- Incremento de los vuelos realizados en los EEUU.
- Tendencia del número de retrasos.
- Factores de cancelación de vuelos.
- Vuelos con retraso sobre el resto.
- Aeropuertos con mayor número de retrasos.

3. Realizar un modelo que permita la predicción de retrasos en los vuelos. En este trabajo se buscarán patrones que permitan mejorar la eficiencia de un aeropuerto y así también prever futuros retrasos.

[1] NIST: https://www.nist.gov/

3 Metodología

En este proyecto se utiliza la metodología *Scrum* para planificar los requisitos a cumplir en cada iteración del desarrollo, tal y como se observa en la Figura 1.

Se añadirán nuevas funcionalidades a esta infraestructura, una vez esté probada y validada, cerciorándose de que todas las funciones requeridas estén correctamente aplicadas.

La fase de análisis se realizará con un desarrollo incremental, empezando con la extracción de información sencilla para ir aumentando progresivamente la dificultad y la complejidad, hasta obtener resultados más específicos.

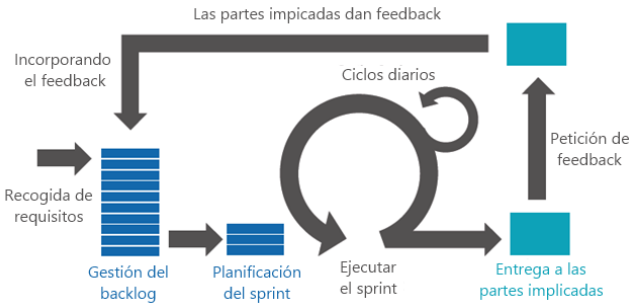

Fig. 1: Metodología *Scrum*. Fuente: https://www.ida.cl/

4 Estado del arte

Grandes empresas tecnológicas como Google, Amazon o Yahoo se encontraron con la dificultad de procesar la enorme cantidad de datos que se generan actualmente, del orden de un Zettabyte a día de hoy. Para continuar desempeñando sus tareas cotidianas con un buen grado de eficiencia, tuvieron que dejar de lado los sistemas de datos tradicionales y empezar a investigar soluciones *big data* para procesar los datos rápidamente. Por ello empezaron a utilizar sistemas distribuidos, para paralelizar el trabajo en distintos ordenadores y ser capaces de trabajar con una gran diversidad de datos [7].

Entre las diversas herramientas que existentes en el mercado cabe destacar *Apache Hadoop* y *Apache Spark*.

- **Apache Hadoop**[2]**:** es un framework de procesamiento distribuido de código abierto, que se utiliza para procesar de forma eficiente conjuntos de datos de gran tamaño. Consta de tres partes: Hadoop MapReduce, framework de procesamiento en paralelo; Hadoop YARN para la planificación de trabajos y gestión de recursos; y el sistema de ficheros distribuido y escalable de Hadoop (Hadoop Distributed File System, HDFS).
- **Apache Spark**[3]**:** es un framework de procesamiento distribuido de código abierto. Utiliza el almacenamiento en memoria y la ejecución optimizada para ofrecer un desempeño rápido, facilitando la creación de programas paralelos, escalables a cientos o miles de máquinas. Dispone de una amplia gama de librerías entre las que se encuentran librerías de aprendizaje automático (MLlib), soporte a operaciones SQL (Spark

[2] Apache Hadoop: http://hadoop.apache.org/
[3] Apache Spark: https://spark.apache.org/

SQL), procesamiento y análisis en tiempo real (Spark Streaming), motor de análisis de grafos (GraphX) y soporte para lenguaje estadístico R (SparkR) entre los más utilizados.

Actualmente Spark es conocido como un entorno de procesado en memoria más rápido y más fácil de usar que Hadoop. Hadoop utiliza MapReduce, que es ineficiente para procesos de algoritmos iterativos o consultas interactivas. Por ello, Spark fue diseñado para ser más rápido en estos casos, ofreciendo un nuevo enfoque de persistencia en memoria y un eficiente sistema de tolerancia a fallos. Sin embargo, ambas tecnologías están preparadas para poder coexistir, por ello Spark puede trabajar sobre HDFS de Hadoop.

En lo referente a librerías de aprendizaje automático, contamos con dos librerías: MLlib para Spark y Apache Mahout para Hadoop:

- **MLlib**[4]**:** es la librería de aprendizaje automático de Spark, librería escalable que proporciona una amplia gama de algoritmos mediante la utilización de un conjunto de elementos distribuidos (*resilient distributed dataset* RDD), sobre los que podemos acceder en paralelo. Incluye varias bibliotecas de estadística, optimización y álgebra lineal. [19]
- **Apache Mahout**[5]**:** proporciona algoritmos de aprendizaje automático escalables para Hadoop, muy utilizado para tareas de agrupamiento, clasificación y filtrado.

Para garantizar que los componentes de Spark operan íntegramente, Databricks, empresa fundada por los creadores de Spark, salvaguarda que el desarrollo de los componentes de Spark cumplan los estándares de calidad que empresas y entornos de producción requieren. Apache Mahout, en cierto modo, sufre una falta de control de calidad riguroso, con lo que no queda claro si los algoritmos están listos para su uso en entornos de producción a gran escala. [21]

Para almacenar los datos que se usaran en nuestra infraestructura, es necesario disponer de una base de datos. Actualmente existen gran cantidad de implementaciones de bases de datos, cada una de ellas optimizada para un tipo de datos o tareas concreto. Se diferencian en dos tipos:

- **SQL:** es una base de datos que se trata como un conjunto de tablas y se manipula de acuerdo con el modelo de datos relacional. Contiene un conjunto de objetos que se utilizan para almacenar y gestionar los datos, así como para acceder a los mismos. Las tablas, vistas, índices, funciones, activadores y paquetes son ejemplos de estos objetos.
- **NoSQL:** Los datos almacenados no requieren estructuras fijas como tablas, normalmente no soportan operaciones JOIN, ni garantizan completamente ACID (atomicidad, consistencia, aislamiento y durabilidad), y escalan horizontalmente. Están altamente optimizadas para las operaciones recuperar y agregar.

Finalmente se escoge una base de datos NoSQL como MongoDB, dado que cumple con todas las necesidades, siendo capaz de almacenar y gestionar un gran volumen de datos eficientemente.

[4]MLlib: http://spark.apache.org/mllib/

[5]Apache Mahout: https://mahout.apache.org/

5 PLANIFICACIÓN

Está dividida en cinco fases, cuyos periodos de realización están organizados en:

5.1. Fase 1 - Arquitectura

En la primera fase se definirá la arquitectura a implementar, con un entorno de almacenamiento y un entorno analítico en modo *standalone*, con el que poder ejecutar las herramientas en un único terminal.

Una implantación en producción supondría escalar los recursos y distribuirlos en diversos nodos, con el objetivo de garantizar un correcto funcionamiento y servicio.

5.2. Fase 2 - Recogida de datos

La segunda fase se centrará en buscar fuentes de datos *open data* para ser analizadas por nuestra infraestructura, con un volumen suficientemente elevado para que permita añadir cierta complejidad a los análisis.

Los datos empleados en este proyecto corresponden a los vuelos realizados en los EEUU, de los cuales se extraerán las primeras conclusiones. Posteriormente se procederá a ampliar la variedad del conjunto de datos, añadiendo datos meteorológicos históricos del aeropuerto durante los despegues y aterrizajes de cada vuelo.

Dentro de esta fase se incluirán las tareas de limpieza, normalización y almacenamiento comentadas anteriormente.

5.3. Fase 3 - Análisis preliminar

En la tercera fase se iniciará el análisis de la base de datos. Se empezará realizando consultas analíticas sobre todo el conjunto de datos. Para llevar a cabo dichas consultas, es indispensable tener almacenados la totalidad de los datos en la base de datos MongoDB.

5.4. Fase 4 - Análisis predictivo

En la cuarta fase se aplicará un modelo predictivo sobre el conjunto de datos con el objetivo de predecir el retraso de un avión en función de las condiciones meteorológicas.

Para enriquecer el análisis y mejorar la capacidad predictiva del modelo, se incluirá un nuevo conjunto de datos que dispone de las condiciones meteorológicas en el momento de despegue y aterrizaje de cada avión.

5.5. Fase 5 - Visualización de datos

La quinta fase se centra en una correcta visualización de los datos.

Se mostrará la información obtenida a través de Zeppelin[6], que es una herramienta que implementa el concepto *web notebook* que permite trabajar sobre una interfaz web en lugar de un terminal de comando. Facilita y agiliza la interacción con nuestro sistema *big data*. [1]

[6]Apache Zeppelin: https://zeppelin.apache.org/

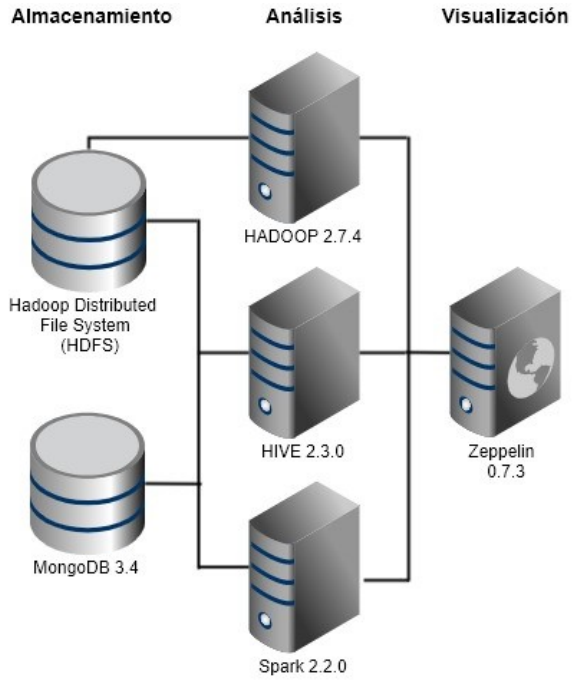

Fig. 2: Diagrama de arquitectura

Además, esta herramienta permite exportar los resultados de las consultas y análisis, guardándolos en formato de fichero.

Para la generación de los gráficos, se empleará la herramienta Plotly[7], que es una herramienta online de código abierto que permite hacer gráficos sin codificación, ni instalaciones adicionales.

6 IMPLEMENTACIÓN

6.1. Fase 1 - Arquitectura

En la primera fase se implementará la infraestructura definida anteriormente.

El montaje es llevado a cabo en un terminal con las siguientes características: procesador Intel Core i7-4790 a 3,6 GHz, 16gb de RAM DDR3 a 1600 MHz y un disco solido de 256GB de almacenamiento.

Con el fin de disponer de mayor versatilidad y modelar un escenario de ejecución flexible, se utilizara una máquina virtual con Ubuntu[8] 16.04 LTS[9] de 60gb, 6gb de RAM y 8 procesadores asignados.

Como sistema de respaldo se irán realizando copias de seguridad a medida que se implementen modificaciones en dicha infraestructura, para salvaguardar los avances ante una posible corrupción de la máquina virtual.

La arquitectura del sistema implementado está dividida en tres partes, como se muestra en la (Figura 2):

- Almacenamiento: dispone de los datos a ser tratados en crudo, que serán transferidos al sistema de ficheros HDFS y a la base de datos NoSQL MongoDB.
- Análisis: aplica el módulo Hive para realizar los análisis estadísticos, así como Spark para crear un modelo predictivo, y Hadoop, el cuál integra MapReduce e incorpora HDFS.
- Visualización: compuesto por Zeppelin, que ofrece un *frontend* trabajando sobre Hadoop y Spark para crear un entorno más amigable en el que poder utilizar, explotar y visualizar los datos almacenados.

A continuación, se muestran los componentes que forman el ecosistema virtual, junto con las herramientas necesarias para cubrir los objetivos definidos.

- Apache Hadoop 2.7.4: Entorno de ejecución masivo de datos. Contiene tres módulos: Hadoop Distributed File System (HDFS), la infraestructura de programación MapReduce, y YARN para la planificación de trabajos y gestión de recursos.
- Apache Hive[10] 2.3.0: Permite realizar consultas SQL y convertirlas a un trabajo MapReduce para el sistema de ficheros de Hadoop.
- Apache Spark 2.2.0: Framework de análisis de datos, que permite: el procesamiento de *streaming*, *machine learning* (MLib), cálculo de grafos (GraphX) y análisis interactivos.
- Apache Sqoop[11] 1.4.6: Permite la transferencia de datos bidireccionalmente entre un sistema de ficheros HDFS de Hadoop y una base de datos externa de nuestra elección.
- Apache Zeppelin 0.7.3: Entorno web que permite el procesamiento, explotación y visualización de datos sobre Hadoop 2.7.4 y Spark 2.2.0.
- MongoDB[12] 3.4: Sistema de base de datos NoSQL orientado a documentos.

6.2. Fase 2 - Recogida de datos

En la segunda fase se realizará la obtención de datos de carácter público de vuelos de aerolíneas de los Estados Unidos [2]. Dicho conjunto de datos registrados contiene un periodo de 20 años, desde 1987 a 2008, con un total de 120 millones de registros.

Como se ha mencionado con anterioridad, la generación del modelo predictivo es enriquecido por datos de vuelos, junto con otros conjuntos de datos. En concreto, añadiendo información de las condiciones meteorológicas en el despegue y aterrizaje de los vuelos, tales Como: existencia de lluvia, niebla, viento y/o nieve, así como la temperatura, entre otros. En la Figura 3 se muestran los diferentes campos de los datos de condiciones meteorológicas comentados.

Agregar esta información permitirá generar un modelo mucho más fiable, puesto que las condiciones climatológicas afectan en gran manera al tiempo total de vuelo, así como son el origen de problemas que afecten a la realización de éste.

De cara a obtener los datos meteorológicos, existen multitud de API (*Application Programming Interface*) de entidades públicas y privadas de donde obtenerlos. Pero, debido a la necesidad de que los datos meteorológicos y de los

[7] Plotly: https://plot.ly/
[8] Ubuntu: https://www.ubuntu.com/download
[9] LTS: https://wiki.ubuntu.com/LTS
[10] Apache Hive: https://hive.apache.org/
[11] Apache Sqoop: http://sqoop.apache.org/
[12] MongoDB: https://www.mongodb.com/es

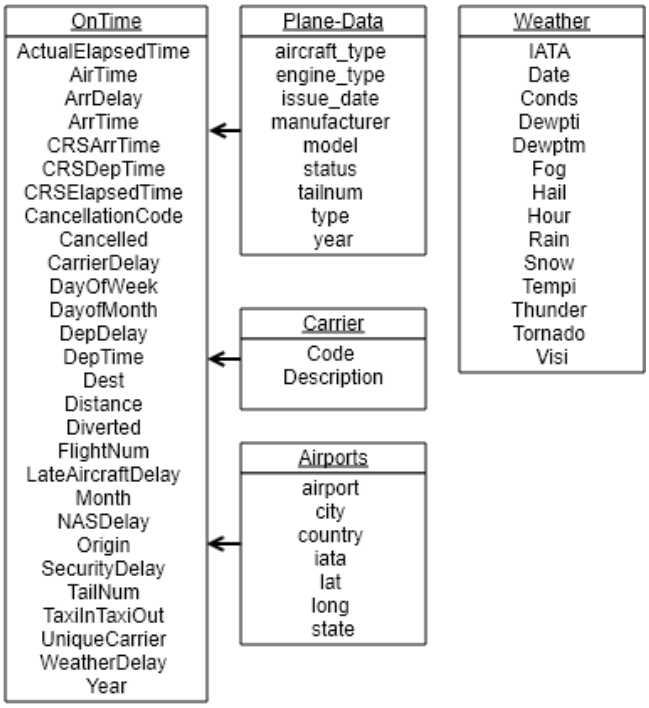

Fig. 3: Diagrama de representación de datos meteorológico y vuelos

vuelos sean del mismo periodo de tiempo, esto reduce el conjunto de opciones a únicamente dos API: la del *National Oceanic and Atmospheric Administration*[13] (NOAA) y la de Wunderground[14].

Finalmente se opta por la API privada Wunderground debido a la facilidad de integrar los datos en nuestro sistema, así como permitir incluir información adicional como las condiciones meteorológicas en modo categórico (lluvia, nieve, viento, etc). Sin embargo, Wunderground alberga restricciones de transferencia, ya que en su versión gratuita limita su uso a 500 peticiones por día y 10 peticiones por minuto.

Debido a las limitaciones de adquisición comentadas, es necesario acortar el periodo de tiempo de estudio a únicamente los datos del año 2004. Este año ha sido escogido debido al hecho que fue el primer año que se registró correctamente las cancelaciones en los vuelos. También se ha acotado el número de datos a sólo la información de un único aeropuerto: el de Oklajoma.

Una vez almacenados los datos de los vuelos, se procede a la creación de una nueva colección de datos que únicamente agrupe los datos vinculados al aeropuerto Oklahoma en 2004. Para asociar los datos del vuelo con los datos climáticos, se ha desarrollado una sencilla aplicación que realizará una lectura de los vuelos y buscará los datos climáticos en el momento del despegue y aterrizaje. Dicho nuevo conjunto de datos meteorológicos es almacenado en el sistema de ficheros Hadoop.

Finalmente se agrupan los datos aéreos y meteorológicos en una única tabla. La estructura de los campos se puede ver en la Tabla 8.

La limitación a la hora de adquirir datos de carácter meteorológico nos limita el volumen de datos. No obstante, supone una ventaja considerable a la hora de trabajar con algoritmos de aprendizaje de MLlib en Spark. Con este subconjunto de los datos, los tiempos de entrenamiento del modelo son inferiores a 1 hora para todo el conjunto. Esto permite reducir los tiempos y experimentar con distintas configuraciones de entrenamiento.

6.3. Fase 3 - Análisis estadísticos

En la tercera fase se utilizarán los datos adquiridos para la realización de estudios estadísticos que permitan generar conclusiones efectivas.

Para la realización de las consultas se incide directamente sobre MongoDB utilizando las herramientas MongoDB Compas[15] y NoSQLBooster[16].

El conjunto de datos usado para el análisis estadístico consta de 123.534.969 vuelos, repartidos en 3.376 aeropuertos en los EEUU, entre octubre de 1987 y diciembre de 2008.

Los diez aeropuertos con mayor afluencia son los que muestra la Tabla 1.

#	Aeropuertos	# vuelos
1	Chicago-O'Hare International	6.597.442
2	William B Hartsfield-Atlanta Intl	6.100.953
3	Dallas-Fort Worth International	5.710.980
4	Los Angeles International	4.089.012
5	Phoenix Sky Harbor International	3.491.077
6	Denver Intl	3.319.905
7	Detroit Metropolitan-Wayne County	2.979.158
8	George Bush Intercontinental	2.884.518
9	Minneapolis-St Paul Intl	2.754.997
10	San Francisco International	2.733.910

TABLA 1: AEROPUERTOS MÁS TRANSITADOS

Se inicia el estudio con un análisis de todo el conjunto de vuelos almacenados, entre 1988 a 2008, viendo que el número de vuelos es especialmente superior en los meses de invierno y verano, tal como muestra la Figura 4.

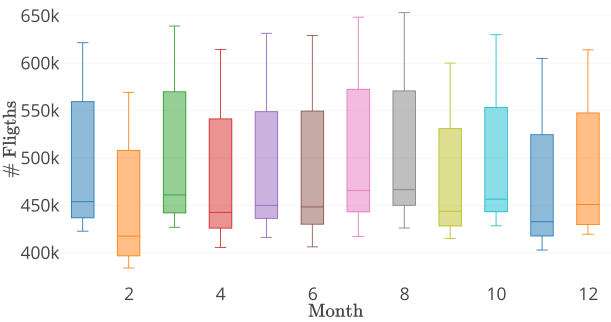

Fig. 4: Diagrama de distribución del número de vuelos totales por meses entre 1988 y 2008

En la Figura 5 se puede observar con mayor detalle el contraste de los vuelos en función del mes. El mes de febrero es el de menor tránsito, mientras que julio y agosto son

[13]NOAA: https://www.ncdc.noaa.gov/
[14]Wunderground: https://www.wunderground.com/
[15]Mongodb Compas: https://www.mongodb.com/products/compass
[16]NoSQLBooster: https://nosqlbooster.com/

los 2 meses del año con mayor afluencia de tránsito aéreo, con 1 millón de vuelos adicionales.

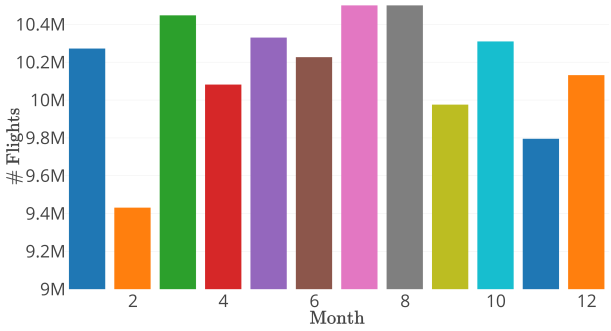

Fig. 5: Diagrama de número de vuelos totales por mes entre 1988 y 2008

Analizando los vuelos en intervalos de 4 años, se puede apreciar (Figura 6) un crecimiento del número de vuelos, salvo en el año 1992 donde se aprecia un claro cambio de tendencia, pudiendo ser una causa el hecho que durante aquel año la tasa desempleo de los EEU era muy elevada, concretamente siendo del 7,5 % de la población activa total [16].

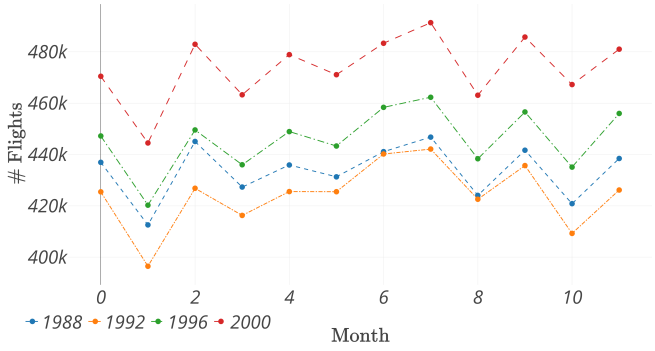

Fig. 6: Diagrama de incremento de vuelos anual

Después de analizar los datos, se descubre que sólo se tienen datos de las cancelaciones a partir del 2004. Por ello, los siguientes gráficos hacen referencia al período 2004-2008.

Analizando la Figura 7 se aprecia un ligero aumento en el número de cancelaciones totales. La figura muestra un desglose según cuatro motivos principales: compañía de transportes (*Carrier*), fenómenos meteorológicos (*Weather*), retrasos de *National Air System Delay* (NAS) y motivos de seguridad (*Security*).

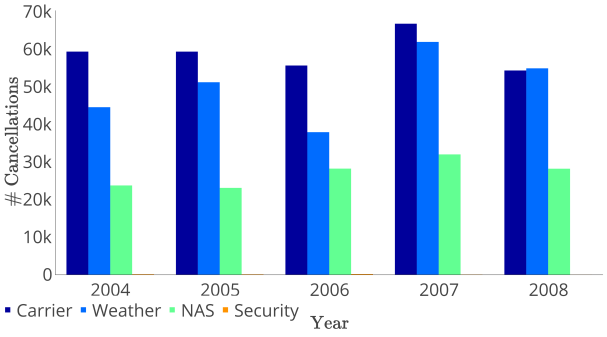

Fig. 7: Diagrama de cancelaciones de vuelos, agrupadas según el motivo.

Las cancelaciones debidas a motivos de seguridad representan una pequeña porción dentro del total, y prácticamente no se pueden ver en la Figura 7. Así, la Figura 8 muestra los datos referentes únicamente a las cancelaciones de vuelos por incidentes de seguridad. Se observa una tendencia a la baja, que pueden ser debido a una mejora en los protocolos aéreos.

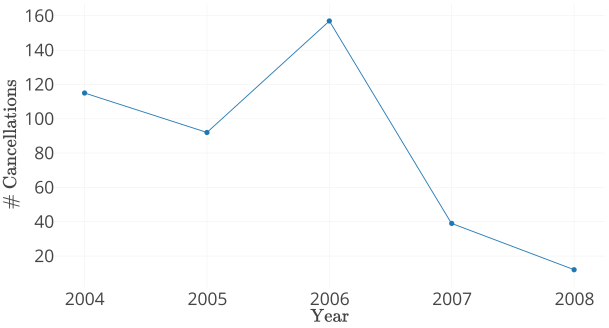

Fig. 8: Diagrama de cancelaciones de vuelos debido a problemas de seguridad.

En relación a las cancelaciones según el mes, la Figura 9 permite ver que existe un claro aumento durante los periodos invernales, debido a la lluvia y la nieve.

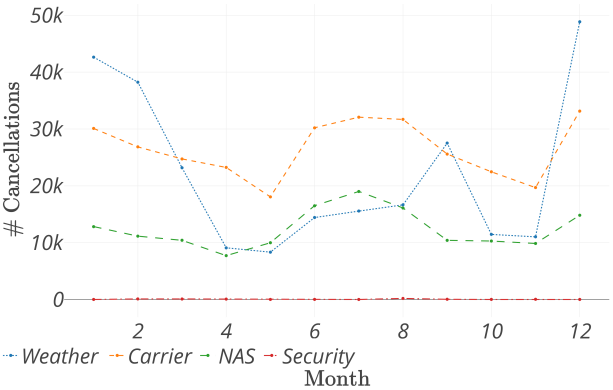

Fig. 9: Diagrama de número de cancelaciones por mes, agrupadas según el motivo.

Respecto los retrasos de los vuelos, es interesante ver en

la Tabla 2 que el 42,24 % son retrasos menores a 10 minutos, mientras que sólo el 10,69 % son superiores a 1 hora.

Franjas	Porcentajes
+60min	10.69 %
20min - 60min	25.05 %
10min - 20min	22.03 %
5min - 10min	17.45 %
0min - 5min	24.79 %

TABLA 2: TABLA DE RETRASOS DE LOS VUELOS.

Finalmente, en la Figura 10 se aprecia cierta correlación entre tiempos de retraso y meses.

Los retrasos de 0-5 minutos son los más comunes, con un 24.79 % del total, cuya cantidad se mantiene estable a lo largo de la temporada.

Los retrasos de 5-10 minutos son el 17.45 % del total y también su número es estable a lo largo del año.

Los retrasos de 10-20 minutos son el 22.03 % del total muestran una cierta estabilidad a lo largo de todo el periodo, con una recesión en el mes de septiembre.

Los retrasos superiores a los 60 y de 20-60 minutos presentan un 10.69 % y 25.05 % respectivamente, y presentan un claro incremento en la temporada de verano (junio, julio y agosto) y en la temporada de invierno (noviembre y diciembre). No obstante, los retrasos de 20-60 minutos son los más frecuentes, frente a los de larga duración que son menos frecuentes.

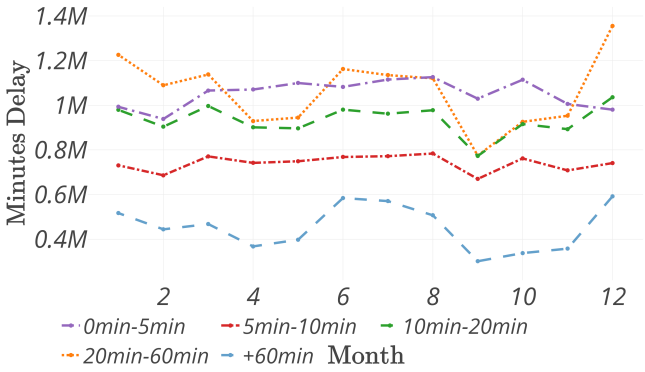

Fig. 10: Diagrama de retrasos.

6.4. Fase 4 - Análisis predictivo

En la cuarta fase se utiliza el algoritmo de aprendizaje automático de la librería MLlib incorporada en Spark. Se realizará un análisis predictivo de los retrasos aéreos sobre el conjunto de datos, añadiendo también sus respectivas condiciones meteorológicas.

Como ya se ha comentado, la adquisición de datos meteorológicos presenta restricciones, por lo cual se debe acotar el conjunto de datos. Se opta por escoger los vuelos procedentes del aeropuerto de Oklajoma, referentes al año 2004. La cantidad de datos total es de 23.000 registros.

Para la implementación del modelo predictivo se usará un algoritmo supervisado. Los métodos supervisados requieren de un conjunto de datos etiquetado para su entrenamiento. El conjunto de entrenamiento es el historial de vuelos mezclado con los datos meteorológicos en el momento del despegue y de la llegada del avión a su destino.

El objetivo será crear un modelo que nos permita predecir el retraso a partir de los datos básicos de un vuelo y de la predicción meteorológica asociada a los aeropuertos de salida y llegada de dicho vuelo.

Los algoritmos de aprendizaje supervisado, se dividen en:

- Algoritmos de clasificación, indicados cuando el atributo objetivo es categórico.
- Algoritmos de regresión, indicados cuando el atributo objetivo es numérico.

Se escogerá un algoritmo de regresión puesto que permite conocer una previsión numérica del retraso de un vuelo.

Dentro del gran conjunto de modelos supervisados, se escoge la rama basada en los árboles de decisión. Un árbol de decisión (Figura 11) es una secuencia de condiciones que son interrogadas con respecto a los datos de entrada, tomando una decisión parcial que lleva hacia una rama u otra, repitiendo este proceso hasta llegar a una hoja donde se toma una decisión final. Dicha hoja final es utilizada como representante o clase de dicha región.

La profundidad máxima de un árbol de decisión es el máximo número de condiciones que es necesario resolver para llegar del nodo raíz a una hoja terminal. [10]

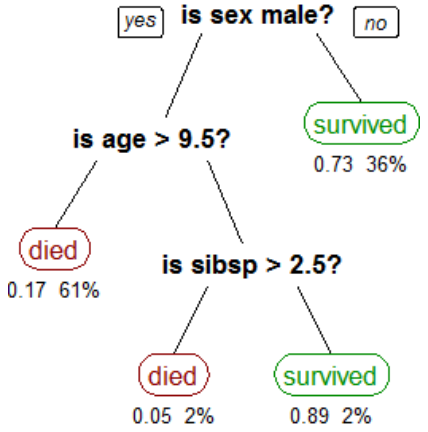

Fig. 11: Decision Tree. Fuente: https://www.medium.com/

Dentro de los diversos tipos de árboles de decisión, se ha elegido uno de tipo *Gradient-boosted tree regresion* (*GBT Regression*), incluido en la librería MLlib[9]. GBT es una técnica basada en conjuntos de árboles de decisión (véase Figura 12). Los GBT entrenan árboles de decisión de forma iterativa para minimizar una función de pérdida. Se entiende como función de pérdida la relación entre un muestreo y un número real que representa el coste asociado con el evento. GBT es válido para la realización de algoritmos tanto de clasificación como de regresión.

Para llevar a cabo el proceso de modelado hay que realizar entrenamientos y evaluaciones en conjuntos de datos de pruebas y métricas de precisión pertinentes [8].

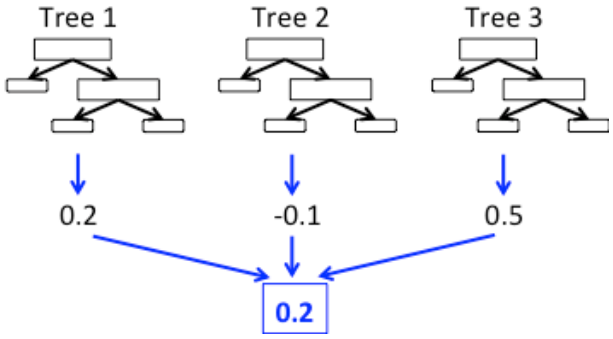

Fig. 12: GBT regression. Fuente: https://databricks.com/

Para entrenar el modelo se divide el conjunto de datos en dos grupos: un 70 % del total de datos se usará para el entrenamiento, mientras que el 30 % restante será usado para pruebas.

En la etapa de entrenamiento se crearán múltiples modelos y se elegirá el mejor de ellos, siendo el criterio de elección la minimización del valor error cuadrático medio (*Root mean squared error* RMSE). Este valor es una medida de precisión para comparar los errores de previsión de un grupo de muestreo de diferentes modelos.

7 Discusión de los Resultados

El primer experimento ha sido realizado utilizando el modelo de regresión *GBT Regresion*. La configuración aplicada consta de un conjunto de cien árboles y diez árboles de profundidad máxima.

En dicho conjunto de datos se han utilizado únicamente los datos de las aerolíneas, sin los datos meteorológicos. En dicho análisis se han descartado los campos no categóricos (cadena de texto), dado que GBT trabaja únicamente con valores numéricos. Después del entrenamiento pertinente hemos obtenido un RSME de 26,6217.

El segundo experimento, se ha aplicado con los mismo parámetros y configuración, realizado una transformación de nuestro conjunto de datos. En dicha transformación se ha transformado los campos categóricos a valores numéricos. Por ejemplo se ha convertido el campo de retrasos (numérico en minutos) a un conjunto de franjas horarias, correspondiente a la Tabla 2.

Dicha modificación a supuesto una mejora del RSME de 1,2944, apreciando una mejor precisión en los resultados obtenidos.

Posteriormente se procede a la inserción de datos meteorológicos, obteniendo un RMSE calculado de 1,2867. Dicho conjunto de datos presenta una leve mejora de la precisión, además de que añadir nuevos parámetros incrementa la tasa de acierto.

#	RSME	Decription
1	26,6217	Datos en bruto.
2	01,2944	Datos modificados.
3	01,2867	Datos modificados + meteorológicos.

TABLA 3: RMSE

En el tercer experimento realizado, se ha modificado la profundidad y el número de árboles:

#	RSME	Time	#Depth	#Tree
0	1,37281	03,03m	5	15
1	1,30578	07,89m	5	25
2	1,32057	06,95m	5	35
3	1,30146	10,03m	5	45
4	1,29234	11,95m	5	50
5	1,28673	55,66m	5	100

TABLA 4: Comparativa de los experimentos, variando el parámetro de número de árboles

#	RSME	Time	#Depth	#Tree
0	1,30578	07,89m	5	25
1	1,40712	17,46m	10	25
2	1,40278	01,70h	15	25

TABLA 5: Comparativa de los experimentos, variando el parámetro de profundidad

Con una profundidad de cinco y un conjunto de cincuenta árboles, se obtiene un buen resultado con un tiempo de ejecución relativamente menor.

En las Tablas 4 y 5 se muestran las principales características de los experimentos realizados, variando tanto la profundidad como el número de árboles del algoritmo GBT.

Es posible mejorar el modelo añadiendo nuevos parámetros categóricos referentes a diferentes aspectos como pueden ser: el número de trabajadores en cada estación o las épocas migratorias de las aves, entre otros.

En dicho modelo hay factores que no se han podido tener en cuenta y que han afectado al resultado, como son: el volumen de datos, o contar únicamente con los valores meteorológicos del origen y destino, obviando los datos de los puntos intermedios por los que transita el avión durante el vuelo.

La reducción del volumen de datos en la fase predictiva ha supuesto una ventaja considerable a la hora de trabajar con Spark, siendo requeridos unos tiempos de entrenamiento entorno a los 10 minutos para todo el conjunto.

El alcance del proyecto ha obligado a utilizar dos fuentes de datos, una con la totalidad de los vuelos (APENDICE A.1.) y otra con los vuelos de 2004 (APENDICE A.2.), junto con los datos meteorológicos del origen y del destino de vuelo del aeropuerto de Oklajoma.

8 Conclusiones

En este trabajo se han analizado todos los vuelos comerciales realizados en los EEUU, durante los años 1987-2008, analizando la correlación entre los diversos factores que afectan a la puntualidad.

Se ha llevado a cabo la implementación de un sistema de análisis y predicción con tecnologías de procesamiento no tradicionales como MLlib de Spark o MongoDB, que han permitido analizar grandes conjuntos de datos de manera eficiente.

Se ha realizado un primer análisis estadístico que ha permitido conocer información relevante del conjunto de datos, como por ejemplo, el número de vuelos o el retraso en dichos vuelos según los meses del año.

El volumen del conjunto de datos ha implicado que se deban aplicar herramientas menos convencionales en el almacenamiento del conjunto de datos. En esta aplicación se ha requerido el uso de una base de datos NoSQL como MongoDB, ya que las bases de datos de tipo SQL tenían restricciones que hacían inviable su uso.

Finalmente se ha desarrollado un modelo predictivo que permite predecir el retraso en función del trayecto y de la previsión meteorológica.

En la fase de resultados, mediante GBT se ha obtenido un RMSE calculado de 1,30578, con una configuración de veinticinco conjuntos de árboles y cinco niveles de profundidad, además de ser una de las soluciones con un menor tiempo de ejecución.

Para continuar con el desarrollo de este proyecto se podrían completar los datos meteorológicos para poder tener un conjunto de entrenamiento y test más grande, que permita crear un modelo más preciso.

Por otro lado, y también como línea futura, se podría estudiar el uso de otros algoritmos predictivos para la generación del modelo, como por ejemplo, las redes neuronales o las *Support Vector Machines* (SVM).

AGRADECIMIENTOS

Escribir este trabajo ha tenido un gran impacto en mí y es por eso que me gustaría agradecer a todas aquellas personas que me han ayudado y apoyado durante este proceso.

Primero de todo me gustaría agradecer a mi familia, por consejos y comprensión. Estado a mi lado a lo largo de toda mi vida académica, sin los cuales no estaría hoy finalizando un ciclo de mi vida.

Finalmente a mí tutor Jordi Casa por orientarme a lo largo de todo el camino, en aquellos momento en los que he estado más desorientado.

REFERENCIAS

[1] ¿Qué es Zeppelin? (el Notebook BigData). (2018). Un poco de Java y +. Obtenido en: https://unpocodejava.com/2016/02/08/que-es-zeppelin-el-notebook-bigdata/

[2] Airline on-time performance. (2018). Statcomputing.org. Obtenido en: http://statcomputing.org/dataexpo/2009/

[3] Apache Hadoop, consultado en septiembre de 2017. Obtenido en: http://hadoop.apache.org/

[4] Apache Spark, consultado en septiembre de 2017. Obtenido en: https://spark.apache.org/

[5] AWS. (2018). Apache Spark en Amazon EMR. Obtenido de: https://aws.amazon.com/es/emr/details/spark/

[6] Barranco, R., (2018). ¿Qué es Big Data?. Ibm.com. Obtenido en: https://www.ibm.com/developerworks/ssa/local/im/que-es-big-data/

[7] Casas, J. (2016 no se el año seguro). Introducción al big data. Manuscrito no publicado, Universitat Oberta de Catalunya, cataluña.

[8] Ciencia de datos mediante Scala y Spark en Azure. (2018). Docs.microsoft.com. Obtenido en: https://docs.microsoft.com/es-es/azure/machine-learning/team-data-science-process/scala-walkthrough

[9] Classification and regression - Spark 2.1.1 Documentation. (2018). Spark.apache.org. Obtenido en: https://spark.apache.org/docs/2.1.1/ml-classification-regression.html#gradient-boosted-tree-classifier

[10] Gironés, J., Casas, J., Minguillón, J., Caihuelas, R,. (2017). Minería de datos: Modelos y algoritmos. Barcelona: Editorial UOC.

[11] Guller, M., (2015). Big Data Analytics with Spark. Editorial Apress

[12] IBM Big Data & Analytics Hub. (2018). Ibmbigdatahub.com. Obtenido en: http://www.ibmbigdatahub.com/

[13] Jurney, R., (2017). Agile Data Science 2.0. Editorial O'Reillhy

[14] Las 4 Vs del Big Data. (2018). BAOSS. Obtenido en: https://www.baoss.es/las-4-vs-del-big-data/

[15] Mohanty, S., (2018). The Four Essential V's for a Big Data Analytics Platform - Dataconomy. Dataconomy. Obtenido en: http://dataconomy.com/2015/06/the-four-essentials-vs-for-a-big-data-analytics-platform/

[16] País, E., (2018). Estados Unidos tuvo en 1992 la mayor tasa de paro desde 1984. Obtenido en: https://elpais.com/diario/1993/01/09/economia/726534022_850215.html

[17] Radtka, Z., Miner, D., (2015). Hadoop with Python. Editorial O'Reillhy

[18] Raschka, S., (2015) Python Machine Learning. Editorial Packt Publishing

[19] Ryza, S. Laserson, U., Owen, S., Wills, Josh., (2017). Advanced Analytics with Spark. Editorial O'Reillhy

[20] Sandy Ryza, Uri Laserson, Josh Wills, Sean Owen. Advance Analytic with Spark: Patterns for Learning from Data at Scale. Abril de 2015. Editorial O'Reilly Media.

[21] Torres, J., Macías, M., Gómez, M., Tous, R,. (2015). Introducción a Apache Spark. Barcelona: 3 UOC

[22] Xinran, H., Pan, J., Jin, O., Xu, T., Liu, B., Xu, T., et al., (2014). Practical Lessons from Predicting Clicks on Ads at Facebook. Obtenido en: https://research.fb.com/publications/practical-lessons-from-predicting-clicks-on-ads-at-facebook/

APENDICE

A.1. Dataset aerolíneas

A continuación, mostraremos los *dataset* relacionados con la base de datos de las aerolíneas:

#	Field	Decription
1	Year	1987-2008
2	Month	1-12
3	DayofMonth	1-31
4	DayOfWeek	1 (Monday) - 7 (Sunday)
5	DepTime	actual departure time (local, hhmm)
6	CRSDepTime	scheduled departure time (local, hhmm)
7	ArrTime	actual arrival time (local, hhmm)
8	CRSArrTime	scheduled arrival time (local, hhmm)
9	UniqueCarrier	unique carrier code
10	FlightNum	flight number
11	TailNum	plane tail number
12	ActualElapsedTime	in minutes
13	CRSElapsedTime	in minutes
14	AirTime	in minutes
15	ArrDelay	arrival delay, in minutes
16	DepDelay	departure delay, in minutes
17	Origin	origin IATA airport code
18	Dest	destination IATA airport code
19	Distance	in miles
20	TaxiIn	taxi in time, in minutes
21	TaxiOut	taxi out time in minutes
22	Cancelled	was the flight cancelled?
23	CancellationCode	scheduled arrival time (A = carrier, B = weather, C = NAS, D = security)
24	Diverted	1 = yes, 0 = no
25	CarrierDelay	in minutes
26	WeatherDelay	in minutes
27	NASDelay	in minutes
28	SecurityDelay	in minutes
29	LateAircraftDelay	in minutes

TABLA 6: VUELOS

#	Field	Description
1	iata	international airport abbreviation code
2	name	name of the airport
3	city and country	place in which airport is located.
4	lat and long	latitude and longitude of the airport

TABLA 7: AEROPUERTOS

#	Field
1	ActualElapsedTime
2	AirTime
3	ArrDelay
4	CRSArrTime
5	CRSDepTime
6	CRSElapsedTime
7	DayofMonth
8	DayOfWeek
9	DepDelay
10	Dest
11	Dest_Conds
12	Dest_Dewpti
13	Dest_Dewptm
14	Dest_Fog
15	Dest_Hail
16	Dest_Hour
17	Dest_Rain
18	Dest_Snow
19	Dest_Tempi
20	Dest_Thunder
21	Dest_Tornado
22	Dest_Visi
23	Distance
24	Diverted
25	FlightNum
26	Month
27	Origin
28	Origin_Conds
29	Origin_Dewpti
30	Origin_Dewptm
31	Origin_Fog
32	Origin_Hail
33	Origin_Hour
34	Origin_Rain
35	Origin_Snow
36	Origin_Tempi
37	Origin_Thunder
38	Origin_Tornado
39	Origin_Visi
40	TailNum
41	TaxiIn
42	TaxiOut
43	UniqueCarrier

TABLA 8: OKC-2004

A.2. Dataset aerolíneas más meteorológico

El siguiente *dataset* es la suma de la información extraída del *dataset* meteorológico y el *dataset* de las aerolíneas, utilizado en la fase de aprendizaje automático Tabla 8.