FACULTAD DE ESTUDIOS ESTADÍSTICOS

GRADO EN ESTADISTICA APLICADA

Curso 2023/2024

Trabajo de Fin de Grado

TITULO: **Influencia de las Condiciones Meteorológicas en los Retrasos de Vuelos: Un Estudio Predictivo del Aeropuerto Internacional O'Hare**

***Alumno*: Angel Molpeceres Mingo**
***Tutora*: Inmaculada Gutiérrez García-Pardo**

Junio de 2024

UNIVERSIDAD COMPLUTENSE
MADRID

Contenido

Resumen

Este estudio aborda la problemática de los retrasos de vuelos en el Aeropuerto Internacional O'Hare de Chicago, utilizando datos recopilados del Departamento de Transporte de los Estados Unidos (DOT) y del Centro Nacional de Información Ambiental (NCEI). Se desarrollaron modelos de predicción de retrasos de vuelos basados en variables meteorológicas, utilizando técnicas de regresión logística binomial, árboles de clasificación, bosques aleatorios y XGBoost. Los resultados revelan la influencia significativa de las condiciones meteorológicas en los retrasos de vuelos y demuestran la eficacia de los modelos propuestos para predecir estos retrasos con precisión.

Abstract

This study addresses the issue of flight delays at Chicago O'Hare International Airport, using data collected from the United States Department of Transportation (DOT) and the National Centers for Environmental Information (NCEI). Predictive models for flight delays were developed based on weather variables, utilizing binomial logistic regression, classification trees, random forests, and XGBoost techniques. The results reveal the significant influence of weather conditions on flight delays and demonstrate the effectiveness of the proposed models in accurately predicting these delays.

1. Introducción y motivación del estudio

La puntualidad en los vuelos es un factor crítico en la industria de la aviación, tanto para las aerolíneas como para los pasajeros. Los retrasos en los vuelos pueden generar una serie de inconvenientes que afectan negativamente la experiencia del usuario, la operación de las aerolíneas y la economía en general. Entender y predecir estos retrasos es, por tanto, una necesidad esencial para mejorar la eficiencia del transporte aéreo y minimizar los inconvenientes para los pasajeros.

En este trabajo, partimos de dos conjuntos de datos clave. El primero, obtenido del Departamento de Transporte de los Estados Unidos, contiene información detallada sobre los retrasos de vuelos en el Aeropuerto Internacional O'Hare de Chicago (ORD) desde enero de 2003 hasta enero de 2023 [1]. Este aeropuerto es uno de los más transitados del mundo, y su análisis ofrece una visión representativa de los problemas que enfrentan las grandes terminales aéreas en términos de puntualidad.

El Departamento de Transporte de los Estados Unidos (DOT) es una entidad federal encargada de supervisar y coordinar las políticas y programas de transporte del país. El DOT recopila y proporciona datos sobre diversos aspectos del transporte, incluyendo la puntualidad de los vuelos, a través de su Oficina de Estadísticas de Transporte (BTS).

El segundo conjunto de datos proviene del Centro Nacional de Información Ambiental (NCEI), una división de la Administración Nacional Oceánica y Atmosférica (NOAA) [2]. Este conjunto incluye lecturas de una estación meteorológica ubicada en el aeropuerto ORD, proporcionando datos medioambientales esenciales como temperatura, velocidad del viento, precipitaciones, entre otros. La NOAA trabaja para mantener al público informado sobre pronósticos meteorológicos, advertencias de tormentas severas y monitoreo climático, jugando un papel crucial en la seguridad y la planificación de actividades relacionadas con el clima.

La relación entre las condiciones meteorológicas y los retrasos de vuelos es un área de estudio bien reconocida. Factores como tormentas, nieve, fuertes vientos y otros fenómenos meteorológicos pueden causar desde leves retrasos hasta cancelaciones de vuelos, afectando significativamente la operación de los aeropuertos y la experiencia de los pasajeros. A través de la combinación de estos dos conjuntos de datos, he desarrollado modelos de predicción de retrasos de vuelos utilizando variables medioambientales como predictores.

La capacidad de predecir retrasos en los vuelos tiene múltiples beneficios. Desde la perspectiva de las aerolíneas, permite una mejor planificación y gestión de recursos, optimizando el uso de aeronaves y personal. Para los aeropuertos, una predicción precisa de retrasos ayuda a gestionar el tráfico aéreo de manera más eficiente, reduciendo cuellos de botella y mejorando la experiencia del pasajero. Gracias a estas predicciones, los pasajeros pueden planificar mejor sus itinerarios, reducir el estrés asociado con la incertidumbre de los retrasos y, en algunos casos, considerar alternativas de transporte o ajuste de horarios.

Para abordar esta problemática, se han implementado varios enfoques de modelado predictivo. Entre los modelos utilizados se incluyen:

- Regresión Logística Binomial: Este modelo se utiliza para predecir si un vuelo sufrirá o no un retraso en función de las variables meteorológicas. La regresión logística es adecuada para problemas de clasificación binaria, donde el resultado es sí/no (retraso/no retraso).

- Árboles de Clasificación: Los árboles de decisión son modelos interpretables que segmentan los datos en subconjuntos más homogéneos en términos de la variable objetivo (retraso). Son útiles para capturar interacciones no lineales entre las variables predictoras.

- Bosques Aleatorios de Clasificación: Este método mejora la precisión de los árboles de decisión al promediar los resultados de múltiples árboles construidos sobre diferentes muestras del conjunto de datos. Los bosques aleatorios ayudan a reducir el sobreajuste y mejorar la generalización del modelo.

- XGBoost de Regresión: XGBoost es una técnica avanzada de boosting que combina múltiples árboles de decisión de manera aditiva para optimizar la precisión predictiva. Es conocido por su alta eficiencia y precisión en tareas de predicción.

Este trabajo busca integrar y analizar datos sobre retrasos de vuelos y condiciones meteorológicas para desarrollar modelos predictivos que mejoren la gestión del tráfico aéreo en el Aeropuerto Internacional O'Hare de Chicago. Los modelos desarrollados pueden servir como herramientas valiosas para aerolíneas, aeropuertos y pasajeros, ayudando a mitigar los impactos de los retrasos de vuelos y mejorando la eficiencia operativa en uno de los aeropuertos más transitados del mundo. La colaboración entre las instituciones como el DOT y la NOAA resalta la importancia de los datos abiertos y accesibles para el desarrollo de soluciones innovadoras que aborden desafíos críticos en la industria de la aviación [9] [10].

2. Depuración y Análisis descriptivo

He partido de dos conjuntos de datos:

- Un primer conjunto con los datos de retrasos de vuelos en el aeropuerto Chicago O'Hare International, procedente de la web oficial del departamento de transporte de los Estados Unidos [1]
- Un segundo conjunto de datos que contiene las lecturas de una estación meteorológica de este mismo aeropuerto, procedentes del Centro Nacional de Información ambiental NCEI, perteneciente a NOAA (National Oceanic and Atmospheric Administration) agencia que trabaja para mantener al público informado sobre pronósticos meteorológicos, advertencias de tormentas severas, monitoreo del clima, etc. [2]

2.1. Depuración de datos

Para obtener un conjunto de datos con el que trabajar, primero necesito depurar los dos conjuntos (el de retrasos y el de meteorología) y posteriormente unirlos en uno.

Conjunto de datos de tiempo meteorológico:

El conjunto inicial cuenta con 108 variables y 783 observaciones.

Elimino variables *Station, Latitude, Longitude, Elevation* y *Name* ya que son comunes para todas las observaciones y representan el código, latitud, longitud, elevación y nombre completo de la estación meteorológica. También creo las variables *year* y *month* a partir de la variable *date* que incluye el conjunto para poder usarlas como clave de unión con el conjunto de vuelos.

Me quedo únicamente con las observaciones posteriores al año 2005 ya que los años anteriores tienen una gran cantidad de datos faltantes

Conjunto de datos de vuelos:

El conjunto inicial consta de 21 variables y 3042 observaciones.

Elimino las variables *Carrier_name*, *airport* y *airport*_name que contienen respectivamente el nombre completo de la compañía que lleva el vuelo, el código del aeropuerto y el nombre completo del aeropuerto. Como con el conjunto de clima, me quedo con las observaciones posteriores al año 2005.

A parte, cambio los códigos referentes a las compañías “PSA Airlines Inc.” y “Comair Inc.” ya que ambas tienen el mismo código IATA (International Air Transport Association) “OH”. A “PSA Airlines Inc.” le asigno el código OH_P y a “Comair Inc.” le asigno el código OH_C.

Conjunto con todas las variables:

Una vez realizados estos primeros pasos, uno ambos conjuntos según las variables *year* y *month*, obteniendo un conjunto que cuenta con 120 variables y 2632 observaciones. Comienzo con la gestión de datos faltantes.

Lo primero que hago es ver cuántos missing tiene cada variable para después eliminar las variables que tengan más de un 5% de valores faltantes. Una vez eliminadas las variables con más datos faltantes, nos quedan las que se muestran en la tabla siguiente (el número debajo de cada una indica el número de observaciones que hay faltantes para esa variable).

Tabla 1 Número de valores faltantes para cada variable

year	month	carrier	arr_flights	arr_del15	carrier_ct	weather_ct	nas_ct
0	0	0	3	4	3	3	3
security_ct	late_aircraft_ct	arr_cancelled	arr_diverted	arr_delay	carrier_delay	weather_delay	nas_delay
3	3	3	3	3	3	3	3
security_delay	late_aircraft_delay	ADPT	ADPT_ATTRIBUTES	ASLP	ASLP_ATTRIBUTES	ASTP	ASTP_ATTRIBUTES
3	3	24	24	24	24	24	24
AWBT	AWBT_ATTRIBUTES	AWND	AWND_ATTRIBUTES	CDSD	CLDD	DP01	DP10
24	24	0	0	0	0	0	0
DP1X	DSND	DSNW	DT00	DT32	DX32	DX70	DX90
0	0	0	0	0	0	0	0
DYFG	DYNT	DYNT_ATTRIBUTES	DYSD	DYSD_ATTRIBUTES	DYSN	DYSN_ATTRIBUTES	DYXP
42	0	0	0	0	0	0	0
DYXP_ATTRIBUTES	DYXT	DYXT_ATTRIBUTES	EMNT	EMNT_ATTRIBUTES	EMSD	EMSD_ATTRIBUTES	EMSN
0	0	0	0	0	0	0	0
EMSN_ATTRIBUTES	EMXP	EMXP_ATTRIBUTES	EMXT	EMXT_ATTRIBUTES	HDSD	HTDD	PRCP
0	0	0	0	0	0	0	0
PRCP_ATTRIBUTES	RHAV	RHAV_ATTRIBUTES	RHMN	RHMN_ATTRIBUTES	RHMX	RHMX_ATTRIBUTES	SNOW
0	24	24	24	24	24	24	0
SNOW_ATTRIBUTES	TAVG	TMAX	TMAX_ATTRIBUTES	TMIN	TMIN_ATTRIBUTES	WDF2	WDF2_ATTRIBUTES
0	0	0	0	0	0	0	0
WDF5	WDF5_ATTRIBUTES	WSF2	WSF2_ATTRIBUTES	WSF5	WSF5_ATTRIBUTES		
0	0	0	0	0	0		

Podemos ver que hay varias que tienen 24 faltantes y varias que tienen 3 faltantes, lo cual puede indicar que estos faltantes estén presentes en las mismas observaciones, con lo que solo eliminando dichas observaciones estaríamos eliminando todos los valores faltantes de estas variables. Tras comprobar que esto es cierto, eliminamos las 24 observaciones que daban missing en las variables de tiempo meteorológico y las 3 observaciones que daban missing en las del conjunto de vuelos (también elimino aparte la única que tiene missing para la variable *arr_del15* para obtener un conjunto sin valores faltantes).

Hecho esto, divido las variables *TAVG*, *TMAX*, y *TMIN* entre 100 y las variables *CDSD*, *CLDD*, *HDSD*, *HTDD*, *EMNT* y *EMXT* entre 10, ya que al leer el archivo csv no se habían leído bien los decimales y tenían valores uno y dos órdenes de magnitud por encima de los que debían tener.

Elimino también las variables que terminan en *_ATTRIBUTES* ya que únicamente sirven para indicar si hay valores missing y a que causa se debe dicho missing, y no tienen ninguna capacidad predictiva.

También codifico las variables dirección del viento *WDF2* y *WDF5* para que en vez de tomar valores numéricos en grados en función de su orientación cardinal (0º/360º = norte, 90º = oeste, 180º = sur y 270º = este), sean dos factores que puedan tomar los valores “Norte”, “Sur”, “Este” y “Oeste”.

Por último, hay una serie de variables (*DX90, DYNT, DYSD, DYSN, DYXP, DYXT*) que, en la descripción original del conjunto de datos, tienen la misma definición (número de días en el mes con una temperatura máxima superior a 32.2ºC), por lo que al no saber que significa cada una, opto por suprimirlas todas del estudio.

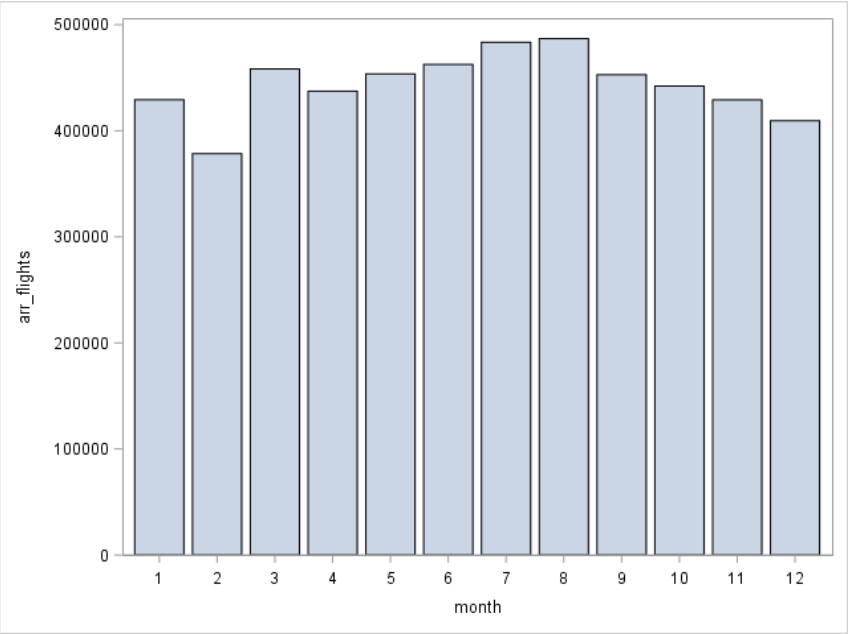
Tras esta depuración nos quedan 53 variables: *year, month, carrier, arr_flights, arr_del15, carrier_ct, weather_ct, nas_ct, security_ct, late_aircraft_ct, arr_cancelled, arr_diverted, arr_delay, carrier_delay, weather_delay, nas_delay, security_delay, late_aircraft_delay, ADPT, ASLP, ASTP, AWBT, AWND, CDSD, CLDD, DP01, DP10, DP1X, DSND, DSNW, DT00, DT32, DX32, DX70, EMNT, EMSD, EMSN, EMXP, EMXT, HDSD, HTDD, PRCP, RHAV, RHMN, RHMX, SNOW, TAVG, TMAX, TMIN, WSF2, WSF5, WDF2, WDF5*.

Nuestro conjunto de datos cuenta con estas variables explicativas y con 2604 observaciones, siendo cada observación una lectura del número de vuelos para una aerolínea concreta en un mes y un año concretos para las condiciones meteorológicas que hicieron ese mes y año.

2.2. Análisis descriptivo

Month

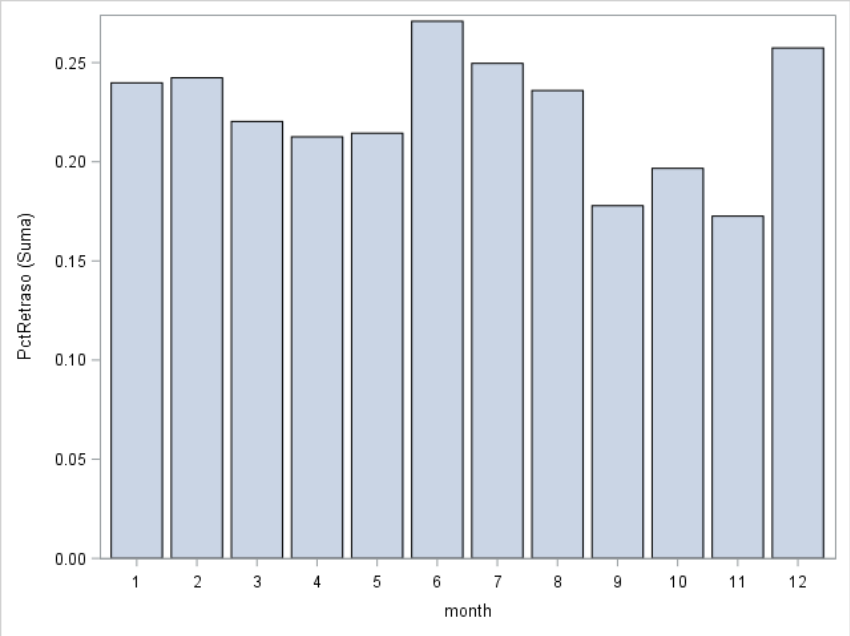
Para esta variable, el valor 1 representa enero y el valor 12 representa diciembre, estando comprendidos entre ambos valores el resto de meses en orden. Veamos cuales son los meses con la mayor cantidad de vuelos:



Gráfica 1 Histograma del número de llegadas por mes

A simple vista, podemos ver que no todos los meses tienen la misma cantidad de vuelos, habiendo en agosto unos cien mil vuelos más que en febrero. Los meses con más vuelos son los de verano (agosto, julio y junio) y los que menos vuelos tienen son los de final e inicio de año (febrero, diciembre, noviembre y enero).

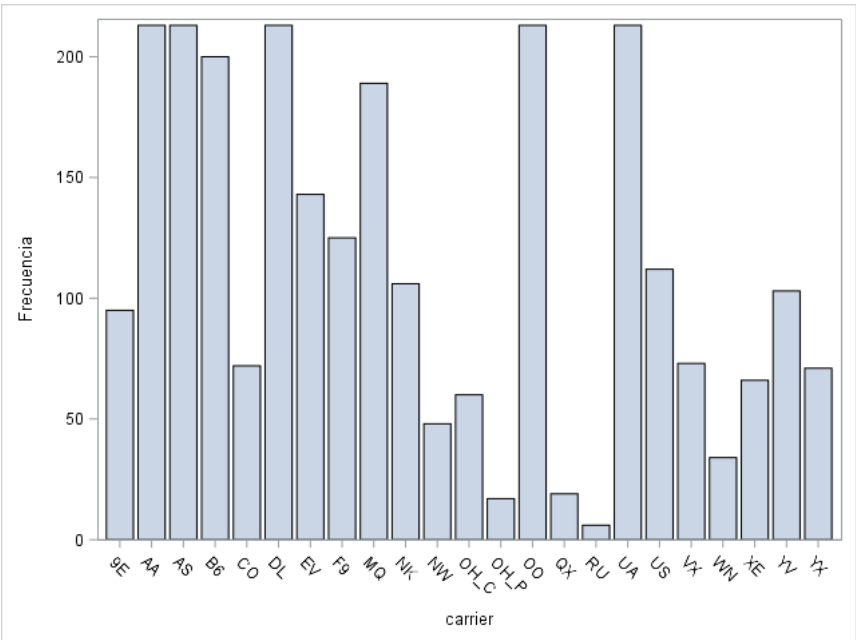
Nos podemos también preguntar cuáles son los meses con mayor porcentaje de retrasos. Para ello creo la variable proporción de retrasos dividiendo el número de vuelos retrasados 15 o más minutos entre el número total de vuelos de la aeronave para ese mes. Hago después una media mensual de esa variable obteniendo los siguientes resultados.



Gráfica 2 Histograma del porcentaje de retrasos para cada mes

Como podemos observar, los meses con una mayor probabilidad de retraso son junio y diciembre, siendo en el mes de junio 1.5 veces más probable que en el mes de noviembre que un vuelo se retrase.

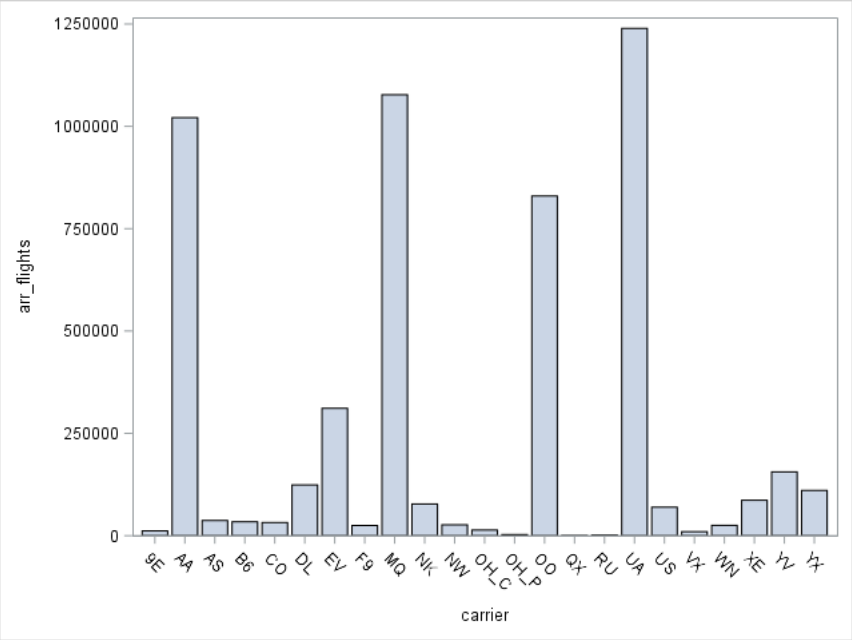
Carrier



Gráfica 3 Histograma de la frecuencia de aparición de aerolíneas

Las aerolíneas que más meses distintos llevan operando en nuestro conjunto de datos son: AA (American Airlines), AS (Alaska Airlines), B6 (Jetblue Airways), DL (Delta Airlines), OO (Skywest Airlines) y UA (United Air Lines).

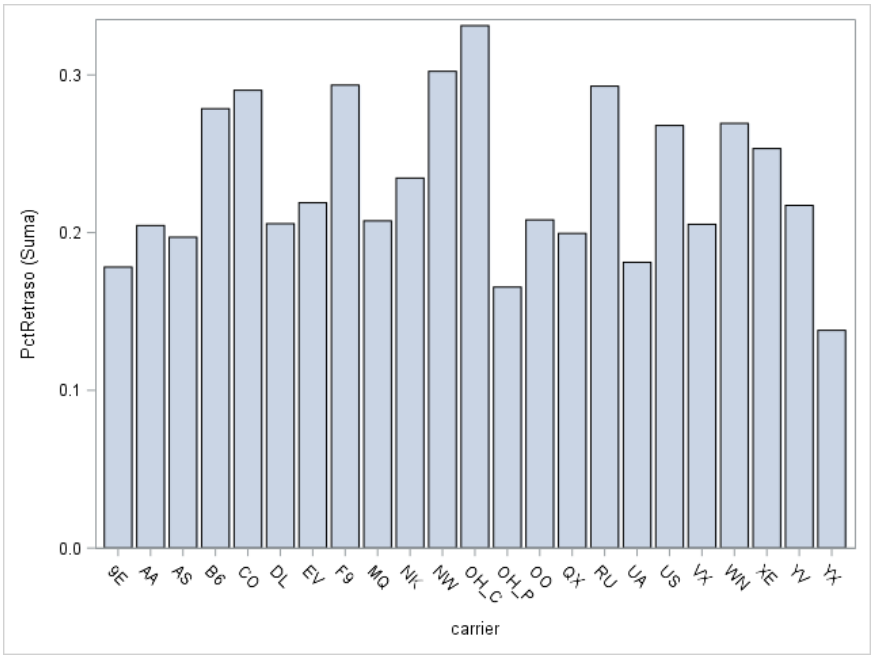
Si queremos ver realmente el número de vuelos que ha hecho cada aerolínea, tendríamos que ver la suma total de la variable *arr_flights* para cada valor de la variable *carrier*. Lo hacemos a continuación.



Gráfica 4 Histograma del número de vuelos totales por aerolínea

Las aerolíneas UA, MQ y AA son las que más vuelos tienen, aunque MQ (Envoy Air) no haya operado tanto tiempo como las otras dos, sí que es de las que más vuelos tiene.

Como con la variable de mes, es posible que nos interese saber qué aerolíneas tienen según nuestra base de datos un mayor porcentaje de retrasos.



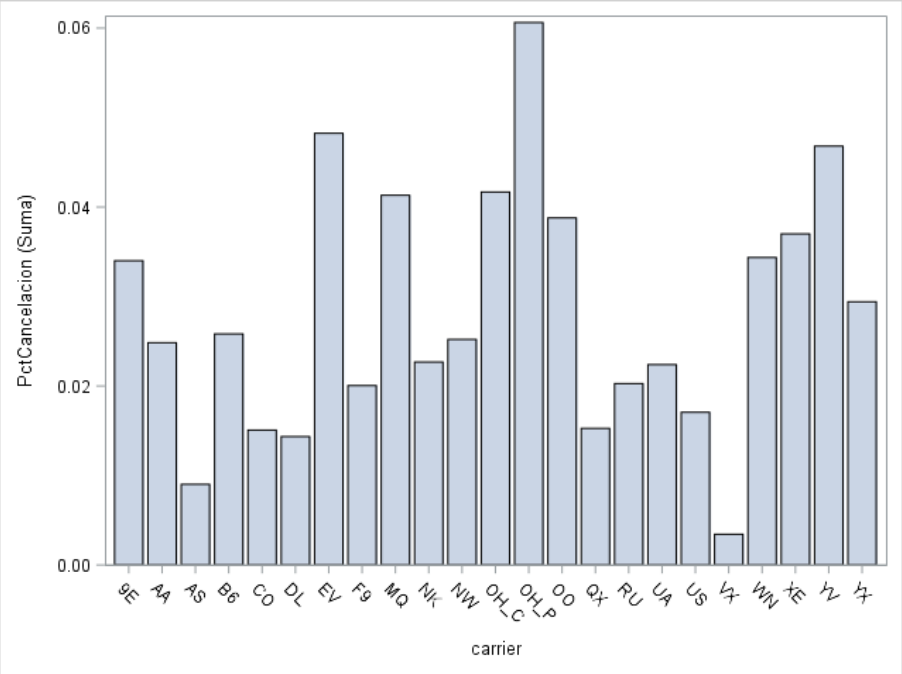
Gráfica 5 Histograma del porcentaje de retrasos medio para cada aerolínea

Las aerolíneas OH_C (Comair), NW (Northwest Airlines), F9 (Frontier Airlines), RU (ExpressJet Airlines) y CO (Continental Air Lines) son las que más porcentaje de retrasos

tienen, retrasándose 15 minutos o más sobre la hora prevista de llegada casi 1 de cada 3 vuelos.

La aerolínea más consistente a la hora de no retrasarse más de 15 minutos es YX (Republic Airlines), una aerolínea regional con sede en Indianápolis que lleva operando desde 2018.

También le puede interesar a un usuario de estas compañías, cual es la que más vuelos cancelados tiene.



Gráfica 6 Histograma del porcentaje de cancelación medio para cada aerolínea

La aerolínea con mayor porcentaje de vuelos cancelados sobre su total de vuelos es OH_P (PSA Airlines) con un 6% de sus vuelos, mientras que la que menor tiene es VX (Virgin America) con un 0.3% de sus vuelos cancelados.

Como tenemos multiples variables que cuentan el número de minutos de retraso totales debidos a los distintos tipos de retraso sobre el número total de minutos de retraso para esa aerolínea y mes, podemos ver para cada tipo de retraso qué aerolíneas son las que más se retrasan y cuales son las que menos se retrasan.

Tabla 2 Distintos porcentajes del tiempo total de retraso según causa para cada aerolínea

Obs	carrier	pctEstadoGestion
1	B6	0.34322
2	DL	0.33289
3	WN	0.33139
4	OH_C	0.30202
5	OH_P	0.28857
6	NW	0.28450
7	QX	0.28321
8	AS	0.28183
9	AA	0.28103
10	F9	0.27773
11	EV	0.27315
12	OO	0.26466
13	UA	0.26044
14	YV	0.25983
15	9E	0.25972
16	US	0.25483
17	YX	0.25315
18	MQ	0.24413
19	NK	0.21629
20	VX	0.21314
21	XE	0.17271
22	CO	0.16637
23	RU	0.09437

Obs	carrier	pctSeguridad
1	VX	.007940419
2	RU	.005541479
3	NK	.005207053
4	CO	.003474209
5	WN	.002705341
6	US	.002350172
7	B6	.001703181
8	YX	.001540101
9	AA	.001500097
10	AS	.001487425
11	YV	.001170431
12	OH_C	.001164001
13	OO	.001065920
14	XE	.000973322
15	NW	.000867469
16	DL	.000832799
17	9E	.000680460
18	MQ	.000671296
19	UA	.000401283
20	EV	.000099072
21	OH_P	0
22	QX	0
23	F9	0

Obs	carrier	pctVueloAnterior
1	OH_P	0.44366
2	UA	0.37430
3	MQ	0.37006
4	F9	0.36527
5	AA	0.35725
6	YV	0.35606
7	OO	0.35595
8	WN	0.34148
9	EV	0.33276
10	YX	0.32585
11	VX	0.29698
12	XE	0.29138
13	NK	0.24471
14	B6	0.21699
15	9E	0.20476
16	DL	0.19279
17	US	0.19124
18	CO	0.19078
19	RU	0.18002
20	NW	0.11822
21	QX	0.11123
22	AS	0.10344
23	OH_C	0.05247

Las aerolíneas en las que el mayor porcentaje de su retraso es debido a temas de gestión o al estado de sus aviones son B6 (JetBlue Airways), DL (Delta Air Lines) y WN (SouthWest Airlines). La que menos es RU con solo un 9% de su retraso debido a este tipo de retraso.

Ninguna aerolínea tiene un porcentaje muy alto de su retraso debido a temas de seguridad, por lo que no parece un punto muy importante que tratar.

Sin embargo, vemos que el 44% de los minutos totales de retraso para la aerolínea OH_P (PSA Airlines) se debe a que el vuelo anterior iba con retraso, dato que contrasta mucho con el 5% de la aerolínea OH_C (Comair).

Variables temperatura (ºC) (*EMNT, TAVG, TMAX, TMIN, EMXT*)

Tabla 3 Medias de TAVG, TMAX y TMIN en ºC

Variable	Media
TAVG	10.70
TMAX	15.47
TMIN	5.94

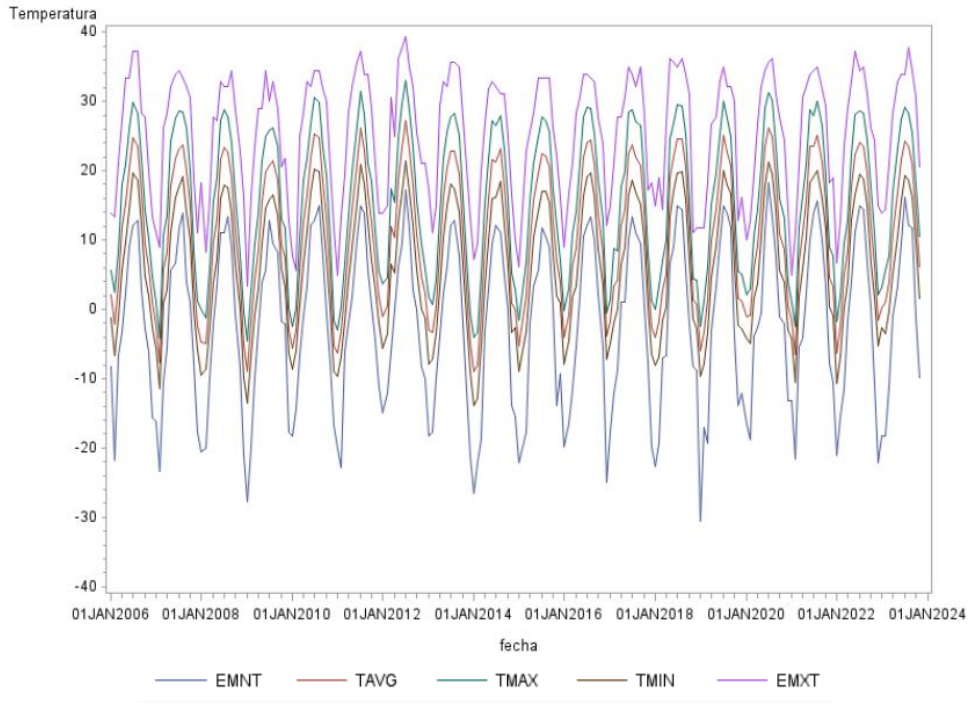
Podemos observar que la temperatura media (*TAVG*) toma en términos medios un valor de 10.70 ºC y que las temperaturas máximas y mínimas toman en media los valores de 15 ºC y 6 ºC. Chicago parece ser por lo tanto una ciudad mayormente fría.

Si queremos ver las temperaturas medias extremas (bajas *EMNT* y altas *EMXT*) para cada mes, obtenemos estas tablas.

Tabla 4 Media para cada mes de EMXT y EMNT en ºC

Obs	month	mediaEMXT	mediaEMNT
1	1	10.51	-19.8
2	2	12.83	-18.5
3	3	21.32	-10.0
4	4	27.77	-2.81
5	5	31.66	3.18
6	6	34.15	9.16
7	7	34.54	13.74
8	8	34.28	13.00
9	9	32.34	7.45
10	10	27.89	-0.29
11	11	19.90	-8.23
12	12	14.43	-15.7

Para los meses de invierno, los valores medios de las temperaturas bajas extremas son verdaderamente bajas, alcanzándose casi -20ºC en media, mientras que para los meses de verano la media de las temperaturas más altas para el mes apenas se alcanza los 35ºC. Todo esto nos hace ver que Chicago ha sido en los últimos 17 años mayoritariamente fría, con días de invierno muy duros y días de verano calurosos.



Gráfica 7 Evolución de los valores de las variables temperatura a lo largo del periodo

A la vista de este gráfico, no se observa un aumento general de los valores de nuestras variables temperatura durante el tiempo transcurrido en nuestro conjunto.

Variables presión (hPa) (*ASLP, ASTP*)

Tabla 5 Medias para cada mes de ASLP y ASTP en hPa

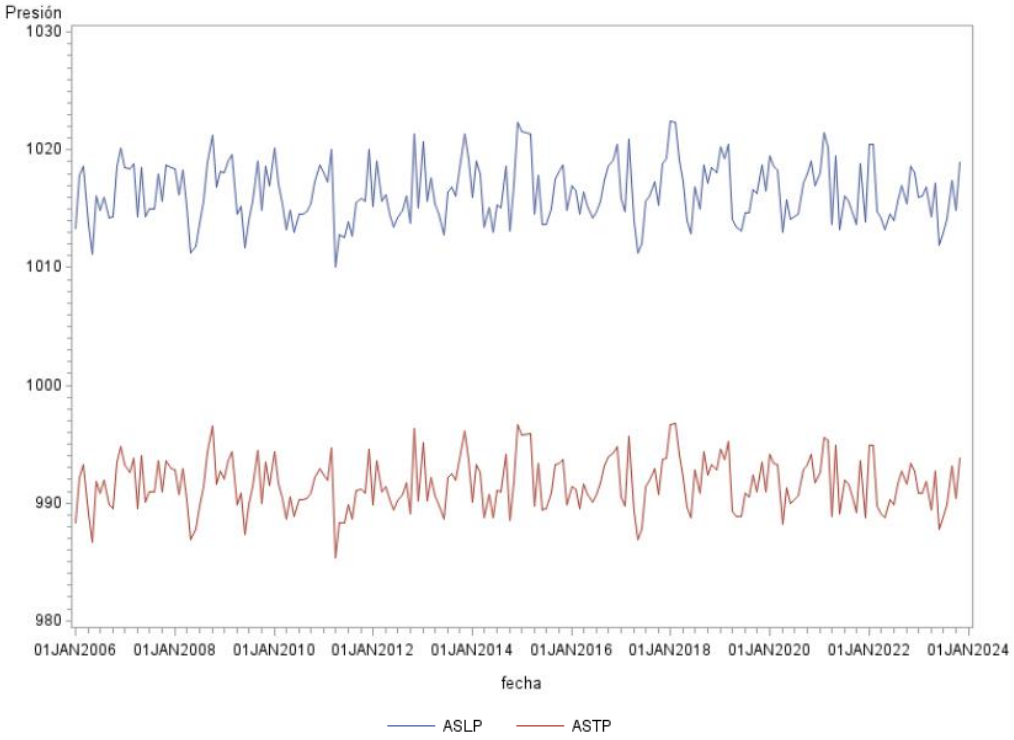
Obs	month	mediaASLP	mediaASTP
1	1	1018.4662162	992.88783784
2	2	1018.2559809	992.76746411
3	3	1018.240724	993.06877828
4	4	1014.3232877	989.51415525
5	5	1014.7245455	990.27454545
6	6	1013.2287671	989.04474886
7	7	1014.7272727	990.61590909
8	8	1015.155	991.01454545
9	9	1016.9465753	992.59543379
10	10	1015.9616505	991.28786408
11	11	1018.5540909	993.42045455
12	12	1018.0277512	992.60095694

Podemos observar que las medias de ambas variables permanecen relativamente constantes a lo largo del año, con pequeñas variaciones mensuales. Esto sugiere una estabilidad general en las condiciones atmosféricas a lo largo del año en términos de presión atmosférica.

Aunque las medias permanecen estables en general, hay algunas variaciones estacionales. Por ejemplo, durante los meses de invierno (diciembre, enero y febrero), las medias de ambas variables tienden a ser ligeramente más altas en comparación con los meses de verano (junio, julio y agosto), lo que puede reflejar patrones climáticos estacionales típicos.

Tabla 6 Medias de ASLP y ASTP en hPa

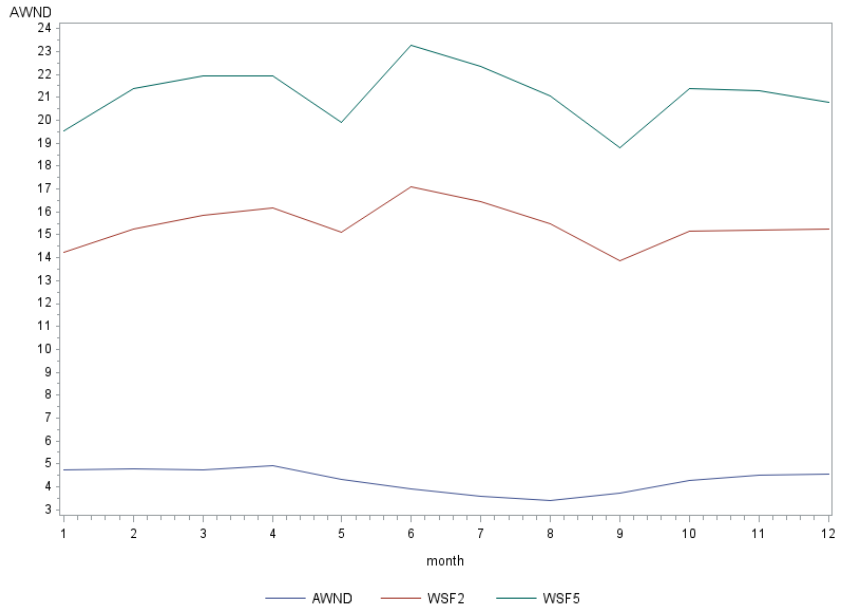
Variable	Media
ASLP	1016.38
ASTP	991.5864055



Gráfica 8 Evolución de los valores de las variables ASLP y ASTP durante el periodo

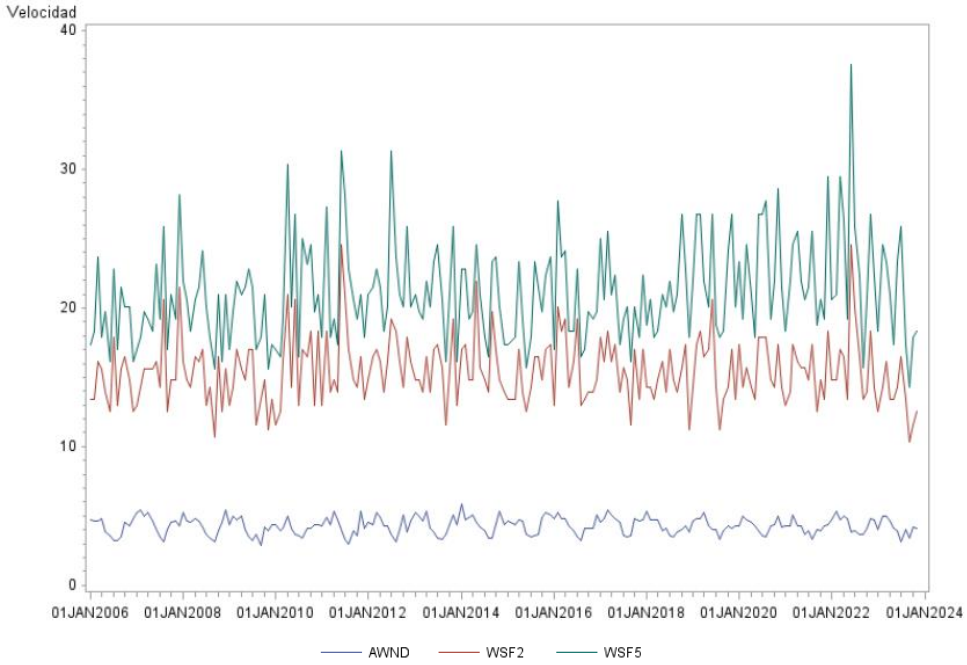
En general, la presión promedio a nivel del mar (*ASLP*) es más alta que la presión promedio a nivel de la estación (*ASTP*) en todos los meses del año. Esta diferencia se debe a la altitud de la estación ya que el aeropuerto de Chicago está a una altitud sobre el nivel del mar de 204 metros.

Variables velocidad viento (mi/h) (*AWND, WSF2, WSF5*)



Gráfica 9 Valores medios de las variables velocidad del viento para cada mes

Podemos observar que no existe una gran variación en las velocidades del viento para los distintos meses del año. Para las variables *WSF2* y *WSF5* que registran las velocidades máximas para vientos de 2 minutos y de 5 segundos, existen valles para los meses de mayo y septiembre en los que se alcanzan mínimos locales y también picos como para el mes de junio en el que se alcanzan los valores más altos en media. La variable *AWND* que representa la velocidad promedio mensual se mantiene más constante.



Gráfica 10 Evolución de los valores de las variables velocidad del tiempo durante el periodo

Esta es una representación de los valores de estas variables para la totalidad de observaciones de nuestro conjunto de datos. Podemos observar que las mediciones para los vientos más rápidos de 5 segundos han ido incrementando en los últimos años si las comparamos con los primeros registros del conjunto.

Variables dirección viento (*WDF2, WDF5*)

Nos puede interesar mirar las direcciones en las que sopla el viento más frecuentemente, por lo que realizamos un análisis de frecuencias de las distintas categorías de las variables.

Tabla 7 Tabla de frecuencias para las categorías de la variable WDF5

dirWDF5	Frecuencia	Porcentaje	Frecuencia acumulada	Porcentaje acumulado
Este	1311	50.35	1311	50.35
Norte	474	18.20	1785	68.55
Oeste	102	3.92	1887	72.47
Sur	717	27.53	2604	100.00

Tabla 8 Tabla de frecuencias para las categorías de la variable WDF2

dirWDF2	Frecuencia	Porcentaje	Frecuencia acumulada	Porcentaje acumulado
Este	1157	44.43	1157	44.43
Norte	768	29.49	1925	73.92
Oeste	112	4.30	2037	78.23
Sur	567	21.77	2604	100.00

Los vientos más fuertes de dos minutos y de cinco segundos soplan con una mayor frecuencia en la dirección este, y en menor medida hacia el sur y el norte. Soplan muy pocas veces al año hacia el oeste.

Variables grados día de refrigeración (*CDSD, CLDD*)

Para calcular los *CLDD* (Grados Día de Refrigeración), primero se ha registrado la temperatura media diaria para cada día del mes. Si la temperatura media diaria ha superado los 18.3 ºC, se calcula la diferencia entre esta temperatura y los 18.3 grados Celsius. Esta diferencia representa los grados día de refrigeración para ese día en particular. Este proceso se repite para cada día del período. Al finalizar el período, se suman todos los grados día de refrigeración diarios para obtener el total de *CLDD* para ese período.

Una vez entendido el cálculo de los *CLDD*, podemos abordar el *CDSD* (Grados Día de Refrigeración Acumulados). El *CDSD* se calcula acumulando los grados día de refrigeración a lo largo de toda la temporada. Es decir, se suma la cantidad de grados día de refrigeración de cada día desde el inicio de la temporada hasta el final del mes más reciente. Esta suma se realiza mes a mes, acumulando así los grados día de refrigeración de cada mes hasta el momento presente de la temporada, comenzando esta en enero en el hemisferio norte y en julio en el hemisferio sur.

Tabla 9 Valores medios de CDSD y CLDD para cada mes

Obs	month	mediaCDSD	mediaCLDD
1	1	0	0
2	2	0	0
3	3	13.547511312	13.547511312
4	4	49.150684932	35.479452055
5	5	432.45909091	384.34545455
6	6	1518.2968037	1083.2283105
7	7	3307.5272727	1783.7090909
8	8	4885.9227273	1578.3954545
9	9	5550.0502283	668.60273973
10	10	5678.7281553	106.48058252
11	11	5644.2636364	3.5
12	12	5608.3779904	0

Podemos observar que los meses de enero, febrero, y diciembre, no cuentan con días en los que la temperatura media haya superado los 18.3 ºC. Los meses con mayores valores medios de grados día de refrigeración son junio, julio y agosto, los más calurosos.

Variables grados día de calefacción (*HDSD, HTDD*)

Una vez visto lo que representan los grados día de refrigeración, resulta fácil entender como se calculan los de calefacción.

Los *HTDD* (Grados Día de Calefacción) se calculan cuando la temperatura promedio diaria es inferior a 18.3 ºC. Se registra la temperatura media diaria y, si es menor a 18.3 ºC, se calcula la diferencia entre la temperatura base de calefacción y la temperatura media diaria. La suma de estas diferencias para cada día del mes da como resultado los *HTDD*.

Por otro lado, los *HDSD* (Grados Día de Calefacción Acumulados) se calculan sumando los grados día de calefacción de cada día desde el inicio de la temporada hasta el final del mes más reciente. Representan el total acumulado de grados día de calefacción hasta la fecha. En este caso la temporada comienza en julio en el hemisferio norte y en enero en el hemisferio sur, al revés que con los grados día de refrigeración.

Tabla 10 Valores medios de HDSD y HTDD para cada mes

Obs	month	mediaHDSD	mediaHTDD
1	1	19350.711712	6903.7297297
2	2	25159.669856	5948.138756
3	3	29806.950226	4441.4705882
4	4	32496.251142	2682.2328767
5	5	33620.868182	1112.6272727
6	6	33715.205479	128.44748858
7	7	13.372727273	13.372727273
8	8	25.422727273	12.05
9	9	381.6803653	356.30593607
10	10	2375.7669903	1998.4466019
11	11	6382.7409091	3994.55
12	12	12405.382775	5966.4066986

Vemos que no hay años en nuestro conjunto de datos en los que haya habido algún mes en el que la temperatura media no fuera inferior a 18.3 ºC. Vemos que lógicamente los meses de invierno (enero, febrero, noviembre y diciembre) son los que más días de calefacción tienen.

Viendo los resultados de las variables días de refrigeración y días de calefacción, podemos concluir que Chicago es por lo general una ciudad fría en la que por lo general, la temperatura media es inferior a los 18.3 ºC.

Tabla 11 Medias de TAVG, TMAX y TMIN en ºC

Variable	Media
TAVG	10.70
TMAX	15.47
TMIN	5.94

Observando de nuevo la tabla de medias de las distintas variables de temperatura vemos que, por lo general, la temperatura media es inferior a 18.3 ºC e incluso que las temperaturas máximas no llegan en media a este valor.

Variables humedad (*RHAV, RHMN, RHMX*)

Las variables humedad se suelen interpretar como la cantidad de vapor de agua que está presente en el aire en relación con la cantidad máxima que el aire puede contener a esa temperatura. Se expresa en porcentajes y por lo general una humedad relativa moderada varía entre el 30% y el 70%, valores inferiores se consideran secos y valores superiores se consideran húmedos.

Tabla 12 Valores medios de RHAV, RHMN y RHMX em hPa

Variable	Media
RHAV	67.0552995
RHMN	49.4289555
RHMX	86.3521505

En media, los valores de humedad mínimos están dentro del intervalo de valores normales, el valor medio mensual es ligeramente inferior al 70% pero vemos que los valores máximos están por encima del 70%, con un valor medio de 86.35%.

Estos valores podrían estar indicándonos que el intervalo medio de las medidas de humedad se halla en una franja más húmeda que seca ya que se mueve entre valores por encima del 50% principalmente.

Tabla 13 Valores medios de RHAV para cada mes en hPa

Obs	month	mediaRHAV
1	1	73.445945946
2	2	69.703349282
3	3	65.895927602
4	4	61.351598174
5	5	62.027272727
6	6	63.03196347
7	7	64.472727273
8	8	66.627272727
9	9	67.899543379
10	10	67.936893204
11	11	68.072727273
12	12	74.674641148

Podemos observar que, en media, existe una mayor humedad en los meses de invierno y una menor en los de verano, aunque por lo general los valores son siempre superiores al 60%, por lo que no se puede decir que haya meses húmedos y meses secos.

También incluyo en este apartado las variables *ADPT* y *AWBT* ya que, aunque son temperaturas, sirven como indicadores de la humedad del aire.

La variable *ADPT* representa la temperatura promedio mensual del punto de rocío. El punto de rocío es la temperatura a la cual el aire debe estar saturado de vapor de agua para que el vapor de agua comience a condensarse en forma de rocío. Por lo tanto, la temperatura del punto de rocío es una medida de la humedad relativa del aire.

Para interpretar la variable *ADPT* debemos fijarnos en los valores que toma. Si *ADPT* es bajo, indica que el aire es seco, ya que si hay poco vapor de agua hacen falta temperaturas más frías para que se condense. Por otro lado, si *ADPT* es alto, sugiere que el aire es más húmedo ya que, si hay mucho vapor de agua en el ambiente, este es capaz de condensar a temperaturas más altas.

La variable *AWBT* representa la temperatura promedio mensual del termómetro húmedo. El termómetro húmedo mide la temperatura del aire, pero está enfriado por la evaporación del agua, lo que lo hace sensible a la humedad del aire.

Si *AWBT* es similar o mayor que la temperatura ambiente, indica que el aire es seco, ya que la evaporación del agua del termómetro húmedo no tiene un efecto significativo en su temperatura. Por otro lado, si *AWBT* es más bajo que la temperatura ambiente, sugiere que el aire es más húmedo, ya que la evaporación del agua en el termómetro húmedo causa un enfriamiento adicional.

Podemos por tanto interpretar la variable *ADPT* con sus valores absolutos, pero necesitamos una variable temperatura ambiente para poder usar de referencia a la hora de interpretar *AWBT*.

Tabla 14 Valores medios de ADPT y AWBT en ºC

Variable	Etiqueta	Media
ADPT	ADPT	4.3696928
AWBT	AWBT	7.6502842

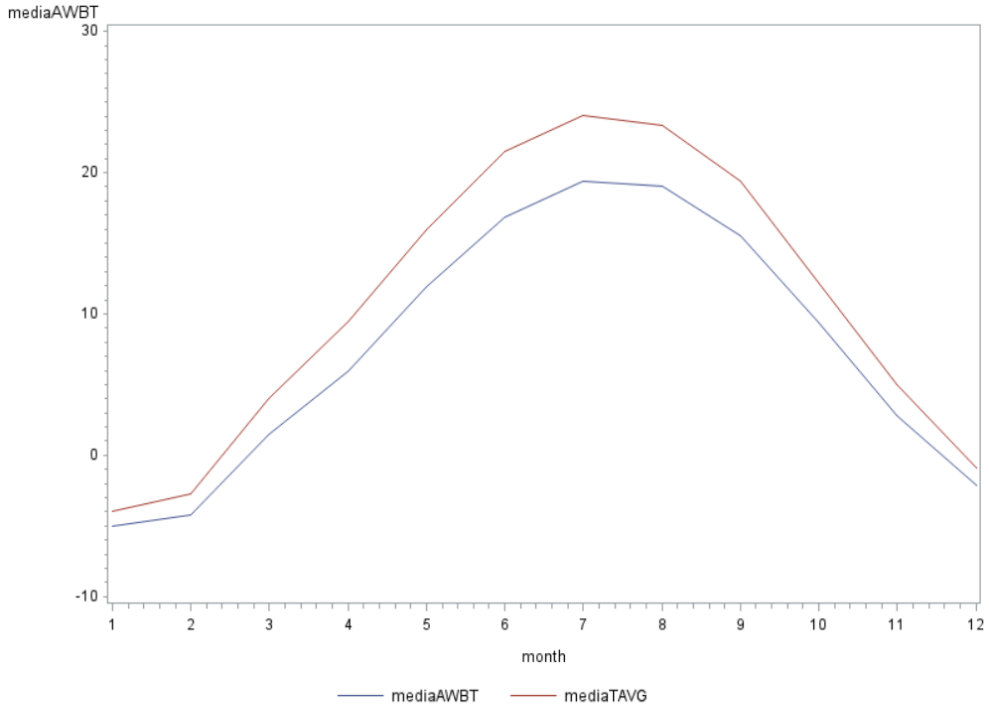
Por lo general, un punto de rocío en torno a los 5 ºC se considera idóneo. Por debajo de ese valor se considera que el aire es relativamente seco y por encima de ese valor que el aire es más bien húmedo. Teniendo esto en cuenta podríamos llegar a la conclusión de que el aire en Chicago es principalmente húmedo en valores medios.

Tabla 15 Valores medios de ADPT en ºC para cada mes

Obs	month	mediaADPT
1	1	-7.945585586
2	2	-7.69507177
3	3	-2.356561086
4	4	1.6183105023
5	5	8.0686818182
6	6	13.623652968
7	7	16.580454545
8	8	16.385681818
9	9	12.801461187
10	10	6.1630582524
11	11	-0.676954545
12	12	-4.871578947

Durante todo el año el aire tiende a ser bastante húmedo, algo más en los meses de verano y algo menos en los meses fríos de invierno.

Para interpretar la variable *AWBT* tomaré como temperatura ambiente de referencia la temperatura media mensual.



Gráfica 11 Medias de AWBT y TAVG en ºC para cada mes

Como he explicado antes, cuando *AWBT* es más bajo que la temperatura ambiente se puede decir que el aire es húmedo, y si es similar o superior, este se considera seco. En nuestro caso vemos que, durante todo el año, *AWBT* es inferior a la temperatura ambiente, por lo que no se puede concluir que en determinadas épocas del año el ambiente sea húmedo y en otras seco.

Variables precipitación (mm) (*EMXP, PRCP*)

Se entiende precipitación como la suma de lluvia y nieve. Realizamos unas medias mensuales usando los datos de todos los años del conjunto.

Tabla 16 Medias de EMXP y PRCP en mm para cada mes

Obs	month	mediaEMXP	mediaPRCP
1	1	164.0	484.9
2	2	240.6	627.7
3	3	247.5	697.4
4	4	306.4	984.9
5	5	374.0	1205
6	6	372.9	1178
7	7	435.1	1119
8	8	316.3	939.1
9	9	379.4	897.3
10	10	308.3	921.7
11	11	203.7	497.4
12	12	234.2	680.0

Los meses con los valores más altos de *PRCP* y *EMXP* (mayo, junio y julio) son en media los meses más lluviosos del año. Esto indica que no solo hay una cantidad significativa de precipitación a lo largo del mes, sino que también puede haber eventos de lluvia intensos en estos meses, como lo sugiere el valor promedio de *EMXP*. En esos meses puede existir un mayor riesgo de inundaciones repentinas, erosión del suelo y otros impactos ambientales.

Los valores de *PRCP* y *EMXP* varían a lo largo del año, lo que refleja la variabilidad estacional de la precipitación. Los meses de primavera y verano tienden a tener mayores promedios de precipitación en comparación con los meses de invierno y otoño.

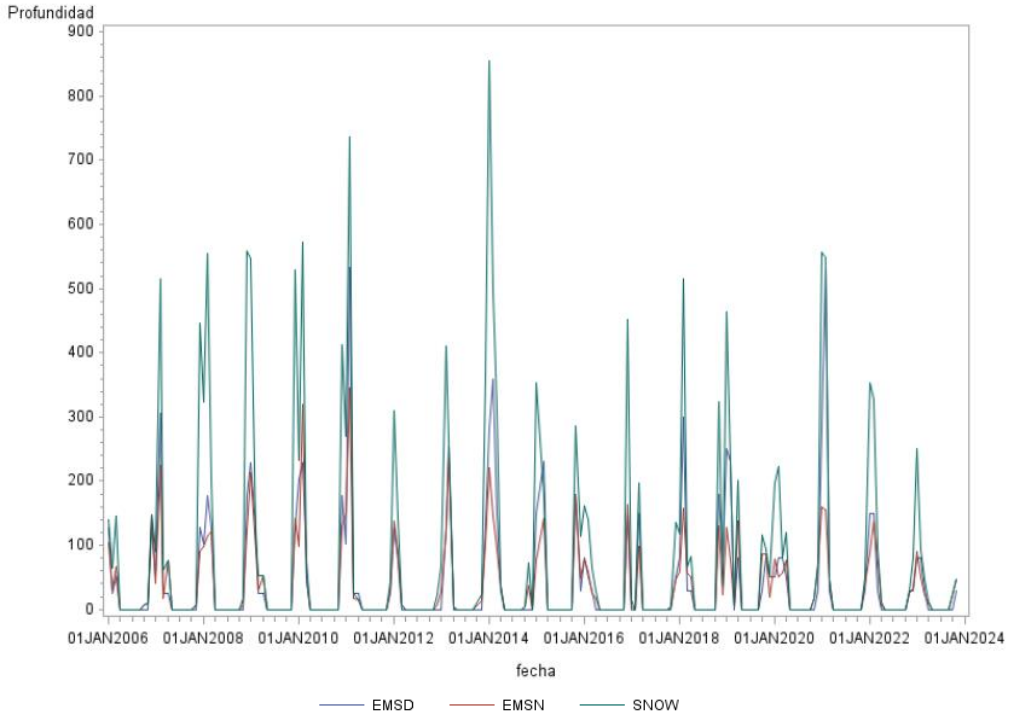
Variables profundidad nieve (mm) (*EMSD, EMSN, SNOW*)

Tabla 17 Medias de EMSD, EMSN y SNOW en mm para cada mes

Obs	month	mediaEMSD	mediaEMSN	mediaSNOW
1	1	131.5	103.7	295.2
2	2	200.4	132.1	342.6
3	3	73.07	64.47	102.9
4	4	14.98	27.51	36.35
5	5	0.00	0.00	0.00
6	6	0.00	0.00	0.00
7	7	0.00	0.00	0.00
8	8	0.00	0.00	0.00
9	9	0.00	0.00	0.00
10	10	1.75	6.86	8.61
11	11	27.41	30.15	50.62
12	12	82.35	72.93	217.5

Durante los meses de invierno, particularmente en enero, febrero y diciembre, se observan medias mensuales significativamente más altas tanto para la mayor profundidad diaria de nieve (*EMSD*) como para la mayor nevada diaria en el mes (*EMSN*), lo que sugiere una mayor acumulación de nieve durante estos meses. Además, la cantidad total mensual de nieve (*SNOW*) también muestra un aumento pronunciado en estos meses, lo que indica una mayor cobertura de nieve en el terreno durante el invierno. Estos meses son consistentemente los más fríos del año y suelen experimentar condiciones climáticas propicias para la formación y acumulación de nieve.

En contraste, durante los meses de primavera y verano (mayo a septiembre), las medias mensuales para todas estas variables relacionadas con la nieve son considerablemente más bajas, llegando incluso a cero en algunos casos. Esto refleja la disminución estacional en la presencia de nieve a medida que las temperaturas se vuelven más cálidas y el clima se vuelve menos propicio para la acumulación de nieve.



Gráfica 11 Evolución de los valores de EMSD, EMSN y SNOW para el periodo

Observamos que en los últimos años la profundidad de la nieve ha ido decreciendo, siendo los valores de la variable *SNOW* menores que al principio del conjunto. Las variables EMSD y EMSN parecen también haber seguido esta tendencia general.

Variables número de días

En relación con la temperatura: DT00, DT32, DX32, DX70

Tabla 18 Valores medios de DT00, DT32, DX32 y DX70 en días para cada mes

Obs	month	mediaDT00	mediaDT32	mediaDX32	mediaDX70
1	1	2.84	27.93	15.53	0.00
2	2	1.76	25.21	11.81	0.10
3	3	0.10	18.14	2.60	1.47
4	4	0.00	5.41	0.06	5.99
5	5	0.00	0.05	0.00	17.27
6	6	0.00	0.00	0.00	27.33
7	7	0.00	0.00	0.00	30.42
8	8	0.00	0.00	0.00	30.34
9	9	0.00	0.00	0.00	22.87
10	10	0.00	1.56	0.00	8.44
11	11	0.00	14.09	1.89	1.37
12	12	1.10	24.86	10.00	0.05

Las variables *DT00* y *DX32* hacen recuento de los días en los que la temperatura máxima es inferior a cierta temperatura, siendo *DT00* la más extrema (-17.8ºC), por lo que ambas variables tomarán valores más altos en los meses más fríos, los de invierno. *DT32* tiene en

cuenta los días en los que la mínima ha sido inferior a 0 grados, lo cual abre un margen mayor y por tanto toma valores más altos a lo largo del año.

Nos podemos fijar en que, de los 31 días que tiene enero, en media casi 28 de esos días tienen una temperatura mínima menor que 0 y también observamos que casi la mitad de los días de ese mes tienen máximas inferiores también a 0 grados.

Por otro lado, la variable *DX70* representa el número promedio de días con temperaturas máximas superiores a 21.1 ºC. Esta variable muestra una variabilidad estacional más pronunciada, con medias mensuales más altas en los meses de verano, indicando una mayor frecuencia de días cálidos durante esta época del año.

Vemos que de los 31 días que tienen julio y agosto, más de 30 de esos días tienen una temperatura máxima mayor de 21.1ºC.

Podemos estar interesados en ver cómo han ido evolucionando las temperaturas a lo largo de los años registrados en nuestro conjunto por lo que realizo las medias anuales para todas las variables.

Tabla 19 Valores medios de DT00, DT32, DX32 y DX7 en días 0 para cada año

Obs	year	mediaDT00	mediaDT32	mediaDX32	mediaDX70
1	2006	0.08	9.58	1.17	11.75
2	2007	0.77	10.28	4.41	13.28
3	2008	0.91	11.49	4.69	11.98
4	2009	0.86	9.84	4.97	10.27
5	2010	0.17	10.00	5.33	13.00
6	2011	0.33	9.42	3.67	10.50
7	2012	0.00	8.21	1.38	13.28
8	2013	0.53	12.04	4.59	11.91
9	2014	1.61	11.34	6.09	11.16
10	2015	0.48	7.87	2.53	12.95
11	2016	0.58	9.00	3.33	13.08
12	2017	0.33	8.50	2.50	13.33
13	2018	0.44	11.40	2.77	11.53
14	2019	0.61	11.12	4.28	11.34
15	2020	0.09	8.61	1.60	12.09
16	2021	0.22	8.59	2.33	13.48
17	2022	0.42	9.35	3.93	12.17
18	2023	0.19	6.92	1.51	13.97

Al examinar las medias anuales de las variables relacionadas con la temperatura a lo largo de varios años, podemos extraer varias observaciones significativas. Primero, se destaca una clara variación estacional en estas mediciones, donde los meses de invierno muestran un aumento en el número de días con temperaturas extremadamente bajas, mientras que los meses de verano exhiben más días con temperaturas máximas más altas.

Además, aunque hay fluctuaciones anuales en las medias, no se percibe una tendencia clara hacia arriba o hacia abajo en la mayoría de las variables a lo largo del tiempo. Esto

sugiere cierta estabilidad en las condiciones climáticas en términos de la frecuencia de días con temperaturas extremas.

Una observación interesante es el aumento gradual en el número medio de días con temperaturas máximas más altas en los últimos años, como se refleja en la variable mediaDX70. Este fenómeno podría indicar un posible calentamiento climático o cambios en los patrones climáticos a largo plazo.

Por último, se observan algunas fluctuaciones notables en ciertos años, como picos inusuales en el número medio de días con temperaturas extremadamente bajas, que podrían estar asociados con eventos climáticos extremos o anomalías temporales en las condiciones meteorológicas.

En relación con la lluvia: DP01, DP10, DP1X

Tabla 20 Valores medios de DP01, DP10 y DP1X en días para cada mes

Obs	month	mediaDP01	mediaDP10	mediaDP1X
1	1	11.50	5.09	0.11
2	2	10.39	5.74	0.47
3	3	10.59	5.90	0.56
4	4	12.62	7.47	0.67
5	5	12.37	8.14	1.19
6	6	12.07	7.83	1.22
7	7	10.48	6.59	0.96
8	8	9.30	6.20	0.76
9	9	8.96	5.83	0.95
10	10	12.06	7.16	0.59
11	11	8.64	4.39	0.39
12	12	11.42	5.89	0.49

En primer lugar, observamos que la variable *DP01*, que representa el número de días con una precipitación mínima de al menos 0.254 milímetros de agua (equivalente a l/m^2) muestra una tendencia relativamente constante a lo largo del año, con ligeros aumentos en los meses de abril y mayo. Esto sugiere que incluso durante los meses más secos, hay una presencia constante de al menos una pequeña cantidad de precipitación.

Por otro lado, la variable *DP10*, que indica el número de días con una precipitación más significativa de al menos 2.54 milímetros, sigue un patrón similar, con una cantidad constante de días con precipitación moderada a lo largo del año. Sin embargo, se observa un ligero aumento en los meses de primavera y otoño, lo que podría estar asociado con cambios estacionales en el clima y los patrones de precipitación.

Por último, la variable *DP1X*, que representa el número de días con precipitación sustancial de al menos 25.4 milímetros, muestra una variabilidad más pronunciada. Los meses de verano tienden a tener menos días con precipitación significativa, mientras que los meses

de invierno y primavera tienden a tener más días con este tipo de precipitación. Esto sugiere una mayor variabilidad estacional en los eventos de precipitación intensa.

Tabla 21 Valores medios de DP01, DP10 y DP1X en días para cada año

Obs	year	mediaDP01	mediaDP10	mediaDP1X
1	2006	11.08	6.17	0.83
2	2007	10.82	6.44	0.86
3	2008	11.53	6.44	1.01
4	2009	11.54	6.60	0.49
5	2010	9.67	5.83	0.50
6	2011	11.83	6.92	0.92
7	2012	9.06	5.41	0.33
8	2013	12.18	6.38	0.65
9	2014	10.88	5.95	0.73
10	2015	10.33	7.37	0.63
11	2016	10.83	6.58	0.50
12	2017	10.17	6.17	0.75
13	2018	12.25	7.22	1.23
14	2019	12.62	7.77	0.83
15	2020	9.84	6.40	0.58
16	2021	9.84	5.15	0.53
17	2022	9.57	5.46	0.50
18	2023	11.20	6.07	0.64

Al observar las medias anuales, vemos que hay cierta variabilidad entre los años para cada una de las variables. Por ejemplo, algunos años, como 2018 y 2019, exhiben medias anuales relativamente altas, mientras que otros, como 2012, muestran medias más bajas. Esta variabilidad interanual refleja la influencia de factores meteorológicos y climáticos específicos de cada año.

A pesar de esta variabilidad, al observar la evolución a lo largo del tiempo, no parece haber una tendencia clara de aumento o disminución en la cantidad de días con diferentes niveles de precipitación. A simple vista, no hay una dirección específica en la tendencia a largo plazo en la precipitación.

Además, al considerar los patrones estacionales, notamos que hay diferencias en la cantidad de días de precipitación según la época del año. Por ejemplo, los meses de verano tienden a tener medias más bajas de precipitación en comparación con los meses de invierno y primavera, lo cual es coherente con los patrones climáticos típicos en muchas regiones.

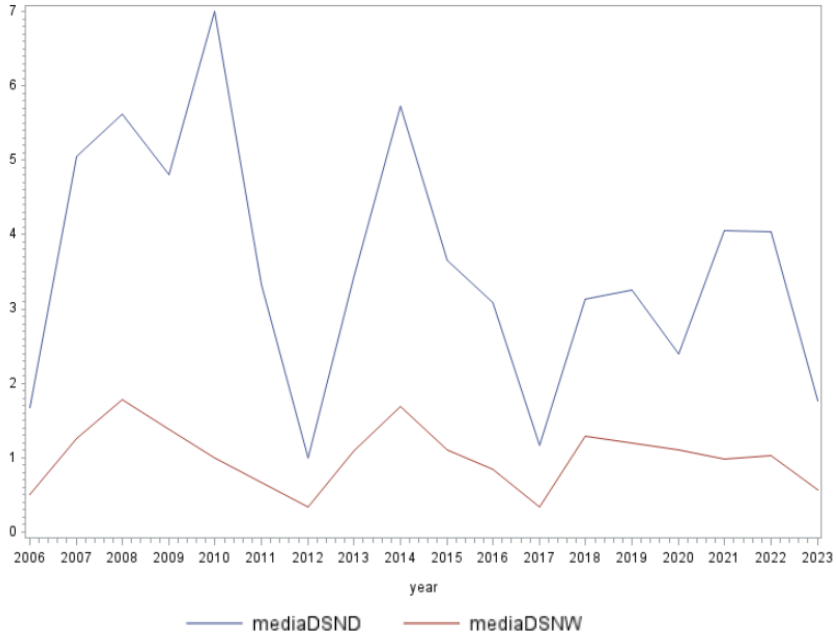
En relación con la nieve: DSND, DSNW

Tabla 22 Valores medios de DSND y DSNW para cada mes

Obs	month	mediaDSND	mediaDSNW
1	1	14.99	3.19
2	2	14.27	4.16
3	3	3.86	1.12
4	4	0.51	0.45
5	5	0.00	0.00
6	6	0.00	0.00
7	7	0.00	0.00
8	8	0.00	0.00
9	9	0.00	0.00
10	10	0.06	0.12
11	11	1.10	0.61
12	12	9.67	2.80

Las medias mensuales de *DSND* y *DSNW* muestran patrones estacionales claros con la presencia más pronunciada de nieve durante los meses de invierno y su ausencia durante los meses cálidos.

Nos puede interesar también qué años ha sido más frecuente la nieve y si, por lo tanto ha habido algún tipo de tendencia, ya sea creciente o decreciente, en el número de días que ha nevado más de una pulgada de profundidad.



Gráfica 13 Evolución de los valores medios de DSND y DSNW en días para cada año

Observamos una variabilidad marcada entre los años. Algunos años muestran una cantidad considerablemente mayor de días con nieve, como 2010, que registra la media más alta de *DSND* con 7 días, mientras que otros años, como 2012, muestran una cantidad

mínima de días de nieve, con apenas 1 día en promedio. Esta variabilidad sugiere que el clima y los patrones climáticos pueden variar significativamente de un año a otro.

Además, a lo largo del período de tiempo considerado, podemos observar fluctuaciones en las medias anuales de *DSND* y *DSNW*. Algunos años muestran un aumento en la cantidad de días de nieve, mientras que otros años tienen disminuciones significativas. Por ejemplo, 2014 muestra un aumento notable en comparación con el año anterior, mientras que 2017 registra una disminución significativa en la cantidad de días de nieve.

3. Aplicación de modelos de predicción

3.1. Modelo de regresión logística binomial

El primer modelo que vamos a proponer es uno de elección discreta, la regresión logística binomial, en el que la variable que pretendemos modelizar es una que toma valores 1 y 0 (suceso y no suceso). En estos modelos, la variable dependiente $\boldsymbol{Y}$ se intenta predecir a partir de un conjunto de **m** variables independientes $\boldsymbol{X_i}$ a través de la siguiente ecuación:

$$\boldsymbol{Y} = \mathbf{f}(\boldsymbol{X_1}, \boldsymbol{X_1}, \dots, \boldsymbol{X_m}) + \boldsymbol{\epsilon} \quad [1]$$

siendo $\boldsymbol{\epsilon}$ el factor que recoge todas aquellas interacciones que no pueden ser modelizadas y se consideran aleatorias. En cuanto a la función **f**, en este caso se recurre a la función de distribución logística, que toma la siguiente forma:

$$\mathbf{P}\,(\boldsymbol{Y} = \mathbf{1}|\boldsymbol{X_1}, \boldsymbol{X_1}, \dots, \boldsymbol{X_m}) = \frac{\mathbf{e}^{(\beta_0+\beta_1 X_1+\cdots+\beta_m X_m)}}{\mathbf{1}+\mathbf{e}^{(\beta_0+\beta_1 X_1+\cdots+\beta_m X_m)}} \quad [2]$$

Esta función es muy conveniente ya que los parámetros β nos ayudan en la interpretación del efecto que tiene cada variable independiente sobre las probabilidades de suceso o no suceso de la variable dependiente [3].

Hecha esta pequeña introducción matemática al modelo de regresión que vamos a abordar, comenzamos con la creación de este. Como hemos explicado, este tipo de modelos requieren de una variable objetivo binaria, por lo que generaremos una que indique cuando se puede considerar que un avión se retrasa y cuando no. Para ello primero creamos una variable que indique la proporción de vuelos retrasados para cada observación dividiendo el número total de vuelos que se ha retrasado entre el número total de llegadas de la aerolínea para dicha observación, creando así la variable *propRetrasos*. Esta variable toma valores entre 0 y 1 por lo que, para convertirla en binaria, consideraremos que un vuelo se ha retrasado (toma valor 1) si la proporción de retrasos es mayor o igual a 0.3 y si es menor, indicaremos que el vuelo no se ha retrasado (toma valor 0). La decisión de tomar 0.3 como punto de corte se debe a que, si un vuelo de cada tres se retrasa para unos valores concretos de las variables dependientes, parece interesante considerar que tiene bastante riesgo de retrasarse. Eligiendo este punto de corte obtenemos un conjunto con una proporción de ceros de 0.7941628 y una proporción de unos de 0.2058372. Con estas proporciones se podría considerar que el conjunto está ligeramente desbalanceado, aun así, estas proporciones de unos y ceros entran dentro de lo razonable ya que en nuestro conjunto hay una mayor proporción de vuelos que no se retrasan que vuelos con retraso. Para evitar un mayor desbalance hemos optado por no tomar 0.4 o 0.5 como punto de corte, ya que, en modelos tan desbalanceados, la capacidad predictiva es bastante más baja.

Una vez elegida nuestra variable objetivo, eliminamos del modelo todas las variables que no vayan a entrar en dicho modelo como explicativas, quedándonos únicamente con las variables *month*, *carrier*, y todas las variables climatológicas que han quedado tras la depuración de nuestro conjunto de datos.

Para poder entrenar nuestro modelo y luego tener un subconjunto de observaciones para poder cuantificar su capacidad predictiva, dividimos el conjunto de datos en dos, uno de entrenamiento y otro de prueba, siendo el de entrenamiento el 80% del conjunto total.

Primeramente, creamos el modelo de regresión con todas las variables, dando como resultado un modelo con 37 variables y 73 parámetros (tenemos tantos parámetros debido a que se han creado dummies para cada categoría de *carrier*, *month* y las variables dirección del viento). A continuación, debemos ver si todas las variables dependientes son significativas, es decir, tienen un efecto significativo sobre la variable objetivo. Para ello llevamos a cabo un análisis de tipo II, el cual cuantifica la disminución de la verosimilitud

del modelo debida a la eliminación de cada una de las variables independientes. En este contraste sencillamente se compara el modelo con cada variable en cuestión con el modelo sin esta variable para ver si empeora o mejora.

Tabla 23 Significatividad de las variables del modelo inicial (Análisis de tipo II)

```
## Analysis of Deviance Table (Type II tests)
##
## Response: retraso
##          LR Chisq Df Pr(>Chisq)
## month     51.730 11  3.052e-07 ***
## carrier  288.143 22  < 2.2e-16 ***
## ADPT       4.537  1  0.0331661 *
## ASLP       1.936  1  0.1641576
## ASTP       1.492  1  0.2219560
## AWBT       3.736  1  0.0532663 .
## AWND      10.257  1  0.0013616 **
## CDSD       6.172  1  0.0129792 *
## CLDD       7.892  1  0.0049643 **
## DP01      38.001  1  7.069e-10 ***
## DP10       2.617  1  0.1057359
## DP1X       8.860  1  0.0029145 **
## DSND      12.777  1  0.0003508 ***
## DSNW       0.939  1  0.3324607
## DT00       0.000  1  0.9858998
## DT32       6.602  1  0.0101864 *
## DX32       3.287  1  0.0698199 .
## DX70       2.049  1  0.1522812
## EMNT       0.731  1  0.3924926
## EMSD       5.089  1  0.0240794 *
## EMSN       2.464  1  0.1164753
## EMXP       0.154  1  0.6951567
## EMXT       0.223  1  0.6364434
## HDSD       1.404  1  0.2361280
## HTDD       7.194  1  0.0073139 **
## PRCP       0.574  1  0.4485162
## RHAV       1.774  1  0.1829236
## RHMN       0.125  1  0.7232132
## RHMX       0.028  1  0.8670822
## SNOW       2.000  1  0.1573178
## TAVG       0.069  1  0.7921364
## TMAX       0.114  1  0.7359523
## TMIN       0.145  1  0.7034005
## WSF2       0.142  1  0.7064948
## WSF5       0.224  1  0.6361505
## WDF2       6.166  3  0.1038301
## WDF5      10.926  3  0.0121344 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Fijándonos en estos resultados podemos ver los efectos de qué variables son significativos y de cuales no, teniendo las significativas desde un asterisco hasta tres en función del p-

valor de cada una para el test en cuestión. Retiramos por tanto las variables que no son significativas para el modelo y volvemos a generar un modelo de regresión binomial para las que sí lo son. Nuevamente hay algunas que no son significativas, por lo que repetimos el proceso hasta que todas lo sean.

Tabla 24 Significatividad de las variables del modelo (Análisis de tipo II)

```
## Analysis of Deviance Table (Type II tests)
##
## Response: retraso
##          LR Chisq Df Pr(>Chisq)
## month    139.175 11  < 2.2e-16 ***
## carrier  301.863 22  < 2.2e-16 ***
## AWND      18.998  1  1.309e-05 ***
## DP01      58.801  1  1.744e-14 ***
## DP1X       7.196  1  0.0073078 **
## DSND       7.656  1  0.0056576 **
## DT32      14.746  1  0.0001230 ***
## WDF5      19.292  3  0.0002378 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Acabamos obteniendo este modelo que únicamente contiene 8 variables y que cuenta con 42 parámetros.

Tabla 25 Significatividad de los parámetros del modelo

```
Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -11.95905    1.41223  -8.468  < 2e-16 ***
month2        0.29899    0.30619   0.976 0.328818
month3        1.15736    0.37462   3.089 0.002005 **
month4        1.81997    0.62451   2.914 0.003565 **
month5        2.86564    0.76933   3.725 0.000195 ***
month6        4.88437    0.78565   6.217 5.07e-10 ***
month7        4.74919    0.81703   5.813 6.15e-09 ***
month8        4.92564    0.83536   5.896 3.72e-09 ***
month9        2.73842    0.83487   3.280 0.001038 **
month10       2.49319    0.73898   3.374 0.000741 ***
month11       0.29811    0.53189   0.560 0.575154
month12       1.03282    0.31201   3.310 0.000932 ***
carrierAA     0.30239    0.47894   0.631 0.527798
carrierAS     0.61253    0.46729   1.311 0.189919
carrierB6     2.24382    0.44593   5.032 4.86e-07 ***
carrierCO     2.74730    0.51583   5.326 1.00e-07 ***
carrierDL     0.34395    0.47408   0.726 0.468138
carrierEV     0.74835    0.48041   1.558 0.119295
carrierF9     1.82793    0.46778   3.908 9.32e-05 ***
carrierMQ     0.45268    0.47853   0.946 0.344156
carrierNK     1.49233    0.48895   3.052 0.002272 **
carrierNW     2.90417    0.58895   4.931 8.18e-07 ***
carrierOH_C   3.46513    0.55096   6.289 3.19e-10 ***
carrierOH_P  -0.56594    1.15295  -0.491 0.623521
carrierOO    -0.01328    0.50046  -0.027 0.978825
carrierQX     0.95628    0.79151   1.208 0.226981
carrierRU     2.72364    1.10646   2.462 0.013833 *
carrierUA    -1.11049    0.59671  -1.861 0.062739 .
carrierUS     1.58577    0.48597   3.263 0.001102 **
carrierVX    -0.22110    0.67418  -0.328 0.742944
carrierWN     2.59838    0.58813   4.418 9.96e-06 ***
carrierXE     1.69590    0.52759   3.214 0.001307 **
carrierYV     1.15109    0.49373   2.331 0.019731 *
carrierYX   -14.07367  288.65780  -0.049 0.961114
AWND          0.90358    0.21033   4.296 1.74e-05 ***
DP01          0.17671    0.02360   7.486 7.08e-14 ***
DP1X          0.20253    0.07538   2.687 0.007211 **
DSND          0.04485    0.01636   2.741 0.006118 **
DT32          0.10452    0.02747   3.805 0.000142 ***
WDF5Norte    -0.07949    0.17670  -0.450 0.652817
WDF5Oeste    -1.85328    0.52858  -3.506 0.000455 ***
WDF5Sur      -0.29995    0.16608  -1.806 0.070915 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Estos son los estimadores de los coeficientes del modelo y su significatividad. Con esta información podemos elaborar la fórmula de nuestro modelo [6].

$$logit(p) = \ln\left(\frac{p}{1-p}\right) = -11.95905 + 1.15736 \times month3 + 1.81997 \times month4 + 2.86564 \times month5 + 4.88437 \times month6 + 4.74919 \times month7 + 4.92564 \times month8 + 2.73842 \times month9 + 2.49319 \times month10 + 1.03282 \times month12 + 2.24382 \times carrierB6 + 2.74730 \times carrierCO + 1.82793 \times carrierF9 + 1.49233 \times carrierNK + 2.90417 \times carrierNW + 3.46513 \times carrierOH\_C + 2.72364 \times carrierRU + 1.58577 \times carrierUS + 2.59838 \times carrierWN + 1.69590 \times carrierXE + 1.15109 \times carrierYV + 0.90358 \times AWND + 0.17671 \times DP01 + 0.20253 \times DP1X + 0.04485 \times DSND + 0.10452 \times DT32 - 1.85328 \times WDF5Oeste \quad [3]$$

Donde p es la probabilidad de que exista retraso. Esta fórmula también se puede expresar de la siguiente manera:

$$p = \frac{1}{1+e^{-logit(p)}} \quad [4]$$

Como he comentado antes, en este tipo de modelos, podemos interpretar el efecto que tiene cada variable independiente sobre las probabilidades de suceso o no suceso de la variable dependiente. Esto se hace a través del análisis de los odds-ratio, valores que permiten aproximar cuanto más o menos probable es que se dé el evento en función del valor que toma cada variable. Es importante destacar que únicamente los analizaremos en el caso de que el parámetro en cuestión sea significativo.

Tabla 26 Odds-ratio de los parámetros del modelo

(Intercept)	month2	month3	month4	month5	month6
6.401067e-06	1.348499e+00	3.181530e+00	6.171659e+00	1.756025e+01	1.322072e+02
month7	month8	month9	month10	month11	month12
1.154907e+02	1.377771e+02	1.546253e+01	1.209982e+01	1.347315e+00	2.808970e+00
carrierAA	carrierAS	carrierB6	carrierCO	carrierDL	carrierEV
1.353086e+00	1.845090e+00	9.429256e+00	1.560049e+01	1.410505e+00	2.113520e+00
carrierF9	carrierMQ	carrierNK	carrierNW	carrierOH_C	carrierOH_P
6.221016e+00	1.572522e+00	4.447448e+00	1.825016e+01	3.198058e+01	5.678239e-01
carrierOO	carrierQX	carrierRU	carrierUA	carrierUS	carrierVX
9.868047e-01	2.602009e+00	1.523575e+01	3.293960e-01	4.883041e+00	8.016352e-01
carrierWN	carrierXE	carrierYV	carrierYX	AWND	DP01
1.344189e+01	5.451574e+00	3.161627e+00	7.724700e-07	2.468427e+00	1.193290e+00
DP1X	DSND	DT32	WDF5Norte	WDF5Oeste	WDF5Sur
1.224502e+00	1.045875e+00	1.110182e+00	9.235899e-01	1.567227e-01	7.408580e-01

Se podrían analizar uno a uno los odds-ratio, pero lo más interesante parece ser obtener una idea de cómo aumenta o disminuye la probabilidad de que un vuelo se retrase en función del valor que toman las variables.

Para las variables cuantitativas (*AWND*, *DP01*, *DP1X*, *DSND*, *DT32*) la interpretación es bastante directa, pero para las variables categóricas es ligeramente distinto ya que lo que se hace es tomar una categoría de referencia ("enero" en el caso de *month*, "9E" en el caso de *carrier* y "*Este" en el caso de WDF5*) y comparar como afecta al valor de *retraso* que cada variable tome un valor en comparación con su categoría de referencia.

Para la variable *month* podemos observar que todos los meses hacen que las probabilidades de retraso sean mayores que enero, siendo junio, julio y agosto los que más

lo hacen. Vemos que si el mes de vuelo es en agosto la probabilidad de que la variable *retraso* tome el valor 1 es 137.7771 veces más grande que en enero en media.

Para la variable *carrier*, las compañías CO, NW, OH_C, RU, WN hacen que un vuelo tenga mayores probabilidades de retrasarse que con la compañía 9E (Endeavor Air). Teniendo un vuelo con Comair (OH_C) una probabilidad de retraso 31.98058 veces mayor en media que con Endeavor Air.

Para las variables de tiempo atmosférico, todas las continuas están relacionadas con un mayor retraso según aumentan sus valores, lo cual tiene sentido porque son variables (velocidad del viento, días con precipitación, días de frío) relacionadas con el mal tiempo. Estas variables son *AWND*, *DP01*, *DP1X*, *DSND*, *DT32* y, analizando más en profundidad, por cada día más en el mes con 0.254 mm de precipitación o más (DP01), la probabilidad de retraso aumenta un 19.3290% de media y por cada milla por hora más rápida que sea la velocidad del viento media ese mes (*AWND*) la probabilidad de que el vuelo se retrase es 2.468427 veces mayor en media.

El único parámetro significativo que hace que un vuelo tenga menos probabilidades de retraso (ya que todos los de *carrier* que son menores que 0 no son significativos) es WDF5Oeste, gracias al cual podemos afirmar que si la dirección del viento de 5 segundos de duración más rápido es Oeste un vuelo tiene un 84% menos de probabilidades de retrasarse que si el viento viene del Este.

Una vez interpretadas las salidas de nuestro modelo, pasamos a analizar la validez de nuestro modelo viendo su capacidad predictiva para datos con los que ha sido entrenado y datos con los que no ha sido entrenado.

Estadísticos para el conjunto de entrenamiento:

```
          Reference     Accuracy     Kappa
Prediction    0    1  0.8387716 0.4161443
         0 1577  258  Sensitivity Specificity
         1   78  171    0.3986014   0.9528701
```

Estadísticos para el conjunto de test:

```
          Reference    Accuracy     Kappa
Prediction   0   1   0.8326923 0.4082252
         0 389  63   Sensitivity Specificity
         1  24  44     0.4112150   0.9418886
```

Podemos observar que, tanto para entrenamiento como para test, los valores para los estadísticos no varían de gran manera, lo cual es algo a destacar ya que se puede afirmar que nuestro modelo predice datos con los que no ha sido entrenado igual de bien que con los que ha sido entrenado. Predecimos correctamente si se ha retrasado el vuelo o no un 83% de las veces y nuestra kappa toma valores entre 0.4 y 0.6 por lo que podemos decir que es moderadamente buena según los estándares habituales.

Contamos con una especificidad muy buena pero una sensibilidad muy lejos de idónea. Entendemos especificidad como precisión a la hora de predecir que no ha habido suceso cuando verdaderamente no lo ha habido y sensibilidad como capacidad de predecir correctamente que ha habido retraso cuando verdaderamente lo ha habido. Esta diferencia

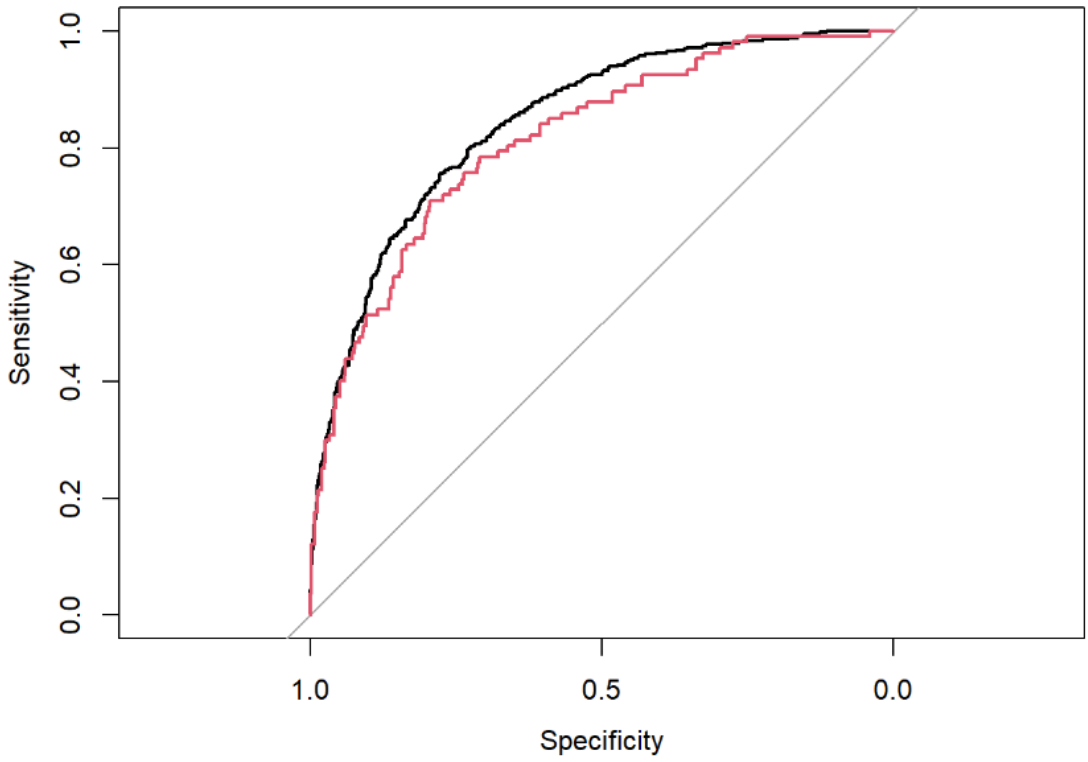
entre ambos estadísticos se debe al desbalance que tenemos entre ceros y unos en nuestro conjunto, por lo que debemos ajustar el punto de corte que sirve para clasificar las observaciones como evento o no evento para solucionarlo. Este parámetro que cambiamos a la hora de predecir nuestras observaciones determina qué consideramos como 1. Ajustándolo por tanto a 0.2 hará que consideremos que se debe predecir 1 cuando el modelo indique que en cierta observación hay un 20% o más probabilidades de que el vuelo se retrase (de forma predeterminada predeciría 1 si hubiera más de un 50% de probabilidades de que se retrasase el vuelo).

Nuevos estadísticos para el conjunto de test:

```
          Reference    Accuracy     Kappa
Prediction   0   1    0.7519231 0.3907800
         0 313  29    Sensitivity Specificity
         1 100  78      0.7289720   0.7578692
```

Ahora vemos que la capacidad de nuestro modelo para detectar casos en los que ocurre o no ocurre retraso es muy similar, acertando el 72.89% de los casos en los que el vuelo se retrasa y el 75.78% de los casos en los que el vuelo no se retrasa. Este balance ha venido en detrimento de la precisión general, que se ha visto reducida en unos 8 puntos porcentuales. Aunque esto pueda parecer negativo y parezcan preferibles las anteriores predicciones (con una precisión mayor), depende mucho del ámbito de aplicación del modelo o de las preferencias de la persona que elabore en modelo, en este caso es más importante que se predigan mejor los casos en los que un vuelo se va a retrasar que los que en los que no se va a retrasar.

Podemos ahora obtener la curva ROC, un gráfico que aporta una medida de evaluación del modelo sin tener que indicar como de desbalanceadas están nuestras observaciones.



Gráfica 14 Curva ROC del modelo de regresión logística binomial

La curva negra es para datos de entrenamiento y la curva roja para datos de prueba. Vemos graficadas sensibilidad frente a 1 – especificidad derivadas de todos los puntos de corte posibles entre 0 y 1 y las clasificaciones obtenidas mediante las probabilidades predichas. Cuanto más cóncava es la curva, mejor clasifica nuestro modelo y cuanto más pegada está a la recta, peor clasifica, representado esta recta el caso de que se estuvieran asignando aleatoriamente unos y ceros con la misma probabilidad.

El AUC (área bajo la curva ROC) es la forma de cuantificar numéricamente esta curva. Obtenemos un AUC del 84.45% para los datos de entrenamiento y una del 81.54% para los de prueba, vemos que no se pierde demasiada precisión cuando se predicen datos con los que no ha sido entrenado el modelo.

Existe una alternativa a la partición train-test que se conoce como validación cruzada repetida que consiste en dividir el conjunto total en submuestras con repetición y construir un modelo con todas las submuestras salvo una, la cual se usa para evaluarlo. Se predicen todas las observaciones y también contribuyen a la construcción del modelo en las anteriores iteraciones. Para asegurar la aleatoriedad del proceso, este se repite para diferentes semillas.

Tras aplicar la validación cruzada repetida para el modelo obtenido en la parte anterior, obtenemos los siguientes estadísticos de los estadísticos para las distintas repeticiones.

Tabla 27 Medias de las medidas de adecuación del modelo para las repeticiones

AUC	Accuracy	Kappa	Sensitivity	Specificity
Min. :0.7717	Min. :0.8010	Min. :0.2379	Min. :0.2442	Min. :0.9033
1st Qu.:0.8065	1st Qu.:0.8168	1st Qu.:0.3352	1st Qu.:0.3469	1st Qu.:0.9335
Median :0.8200	Median :0.8225	Median :0.3640	Median :0.3721	Median :0.9426
Mean :0.8194	Mean :0.8237	Mean :0.3668	Mean :0.3725	Mean :0.9406
3rd Qu.:0.8361	3rd Qu.:0.8318	3rd Qu.:0.3924	3rd Qu.:0.4082	3rd Qu.:0.9517
Max. :0.8726	Max. :0.8654	Max. :0.5282	Max. :0.5059	Max. :0.9668

En lo que a valores medianos se refiere, podemos observar que tenemos un AUC y una precisión por encima del 82% en las repeticiones y que el Kappa es justo según los valores de referencia (0.21 - 0.40). Podemos observar que predecimos mal los casos en los que hay retrasos, pero muy bien los casos en los que no los hay, acertando en media un 94% de las veces. Lo que significa que en el 94% de los casos en los que no hubo un retraso, nuestro modelo habría predicho que no lo iba a haber.

Vuelve por tanto a pasar lo mismo que en el apartado anterior, lo vemos en la matriz de confusión:

```
Cross-Validated (5 fold, repeated 20 times) Confusion Matrix

(entries are percentual average cell counts across resamples)

          Reference
Prediction   Si   No
        Si  7.7  4.7
        No 12.9 74.7

 Accuracy (average) : 0.8237
```

Nuestro modelo mayoritariamente tiende a decantarse más por el "no" a la hora de realizar una predicción, indicando 0 en el 87.6% de las observaciones e indicando 1 únicamente el

12.4% de las veces. Sabemos que nuestro modelo tiene casi un 20% de observaciones en las que ocurre un retraso por lo que ajustamos este valor a la hora de realizar predicciones.

Tabla 28 Medidas de adecuación del modelo antes y después de ajustar el parámetro

	parameter	prob_threshold	Accuracy	Kappa	Sensitivity	Specificity
1	none	0.2058	0.7439063	0.3831352	0.7412462	0.7445921
2	none	0.5000	0.8236596	0.3668318	0.3725267	0.9406042

En la primera fila tenemos los estadísticos habiendo ajustado la proporción de unos y ceros y en la segunda los estadísticos sin ajustarla.

Vemos ahora que ajustando este parámetro la precisión cae en 8 puntos porcentuales pero que la sensibilidad (probabilidad de haber predicho “si” en los casos en los que verdaderamente hubo un retraso) aumenta en gran medida del 37% al 74%. Estos cambios también afectan a la especificidad (haber predicho “no” en casos en los que no hubo retraso) que cae al 74%.

Aunque la precisión haya disminuido, es importante que nuestro modelo sea igual de capaz de discernir entre casos en los que hay retraso y casos en los que no lo hay. El modelo anterior era más preciso porque simplemente tendía a predecir cero en casi todas las observaciones, y como por lo general los vuelos no se retrasan, obtenía un gran porcentaje de acierto.

Por último, vamos a pasar a los métodos de selección automática de variables, que consisten en la selección sistemática de las variables de nuestro conjunto de datos, en función de algunos criterios matemáticos, para su inclusión en modelos de regresión logística.

Los criterios más extendidos de inclusión de variables en modelos predictivos son el AIC (Criterio de Información de Akaike) y el BIC (Criterio de Información Bayesiano). El AIC se basa en la teoría de la información y proporciona una estimación de la calidad relativa de un modelo estadístico para un conjunto de datos determinado y el BIC se basa en la aproximación de la probabilidad posterior de los modelos y tiene una forma similar a la del AIC, pero incluye una penalización más fuerte para la complejidad del modelo [3] [6].

Existen tres métodos distintos:

- Backward: comienza con un modelo inicial que incluye todas las variables independientes y elimina iterativamente la variable menos significativa (siguiendo el criterio que hayamos escogido) hasta que todas las variables restantes sean significativas.
- Forward: comienza con un modelo vacío sin variables independientes y añade iterativamente la variable que más mejora el ajuste del modelo hasta que no haya variables adicionales que mejoren significativamente el modelo.
- Stepwise: combina características de los métodos forward y backward, permitiendo tanto la adición de variables significativas como la eliminación de variables no significativas en cada paso. Se detiene cuando no hay más variables para añadir o eliminar según los criterios predefinidos.

Aplicando estos métodos al conjunto con todas las variables con los distintos criterios (seis combinaciones distintas en total) obtengo los siguientes modelos, siendo cada uno en orden de aparición: modelo stepwise con BIC, modelo stepwise con AIC, modelo forward con BIC, modelo forward con AIC, modelo backward con BIC, modelo backward con AIC.

```
[[1]]
retraso ~ carrier + DP01 + month + HTDD + AWND + RHMN

[[2]]
retraso ~ carrier + month + DP01 + AWND + WDF5 + RHMN + DX70 +
    DT32 + CDSD + DP1X + PRCP + HDSD + ASLP + WDF2 + DP10 + DSND +
    EMSD

[[3]]
retraso ~ carrier + DP01 + month + HTDD + AWND + RHMN

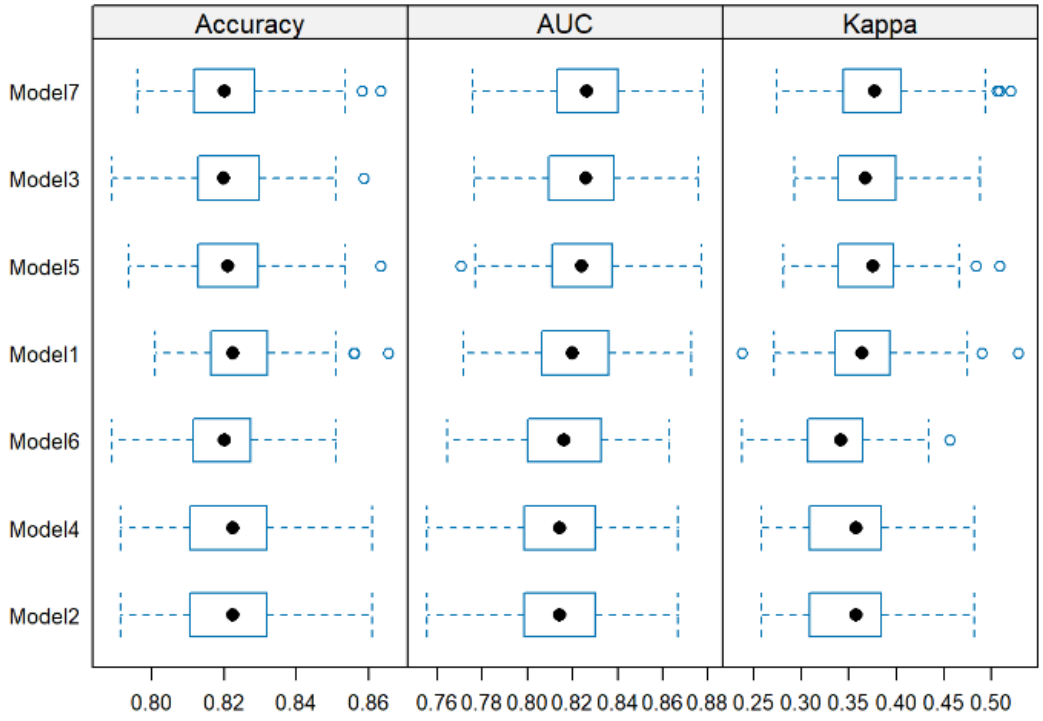
[[4]]
retraso ~ carrier + month + DP01 + HTDD + AWND + WDF5 + RHMN +
    DX70 + DT32 + CDSD + DP1X + PRCP + HDSD + ASLP + WDF2 + DP10 +
    DSND + EMSD + DX32

[[5]]
retraso ~ carrier + ADPT + ASLP + AWND + DP01 + DSND + DT32 +
    RHAV + TMIN + WDF5

[[6]]
retraso ~ month + carrier + ADPT + ASLP + AWND + CDSD + CLDD +
    DP01 + DP10 + DP1X + DSND + DT32 + DX32 + DX70 + EMSD + HDSD +
    HTDD + PRCP + TMAX + TMIN + WDF2 + WDF5
```

Tenemos 5 modelos distintos con los que trabajar ya que los modelos 1 y 3 (stepwise con BIC y forward con BIC) terminan su selección de variables con modelos idénticos.

Una forma clara y sencilla de comparar estos modelos es mediante un gráfico de caja y bigotes en el que se muestran distintos estadísticos de bondad predictiva para cada uno de ellos. Añadimos también el modelo generado en el primer apartado de forma manual para ver si es mejor o peor que los generados de forma automática (los modelos enumerados en el gráfico a continuación corresponden en orden con: logística Binomial a mano, stepwise con BIC, stepwise con AIC, forward con BIC, forward con AIC, backward con BIC y backward con AIC).



Gráfica 12 Gráficos de caja y bigotes de las principales medidas de precisión para los distintos modelos generados de forma automática

Para las tres métricas examinadas (precisión, área bajo la curva y kappa), nos fijaremos principalmente en los puntos negros de cada caja, que representan sus valores medianos, siendo mejores los valores más altos (más a la derecha).

Aunque los márgenes de mejora son muy pequeños, los tres mejores modelos son el 7, el 3 y el 5. Como son tan similares para la tres métricas, será preferible el que menos parámetros tenga. El modelo 7 (modeloBackAIC) tiene 58 parámetros, el modelo 3 (modeloStepAIC) tiene 53 parámetros y el 5 (modeloForwAIC) tiene 55 parámetros, por lo que nos decantamos por el modelo 3, obtenido mediante el método stepwise y usando el criterio AIC.

Comparemos ahora en detalle el modelo ganador con el obtenido a mano para ver si realmente compensa trabajar con los 53 parámetros de este o podemos sacrificar capacidad predictiva por interpretabilidad.

Tabla 29 Medidas de precisión del modelo a mano y el obtenido automáticamente

```
AUC
            Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
Model1 0.7716574 0.8065148 0.8200309 0.8194229 0.8361222 0.8726497    0
Model2 0.7763999 0.8097307 0.8259569 0.8244574 0.8381139 0.8755998    0

Kappa
            Min.  1st Qu.    Median      Mean   3rd Qu.      Max. NA's
Model1 0.2379285 0.335190 0.3639627 0.3668318 0.3924109 0.5282492    0
Model2 0.2929920 0.339524 0.3675458 0.3702421 0.3995313 0.4879085    0

Accuracy
            Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
Model1 0.8009592 0.8168177 0.8225420 0.8236596 0.8318317 0.8653846    0
Model2 0.7889688 0.8129496 0.8199277 0.8210666 0.8297362 0.8585132    0
```

El modelo 1 es el obtenido a mano y el 2 el stepwise con AIC. Vemos que las mejoras del modelo 2 son marginales, siendo incluso el modelo 1 más preciso a la hora de predecir valores, por lo que será preferible el más sencillo (menos parámetros) de los dos, es decir el obtenido a mano ya que cuenta con 42 parámetros frente a los 53 del obtenido mediante selección automática de variables. No solo cuenta con 11 parámetros menos, sino con 9 variables menos, el modelo obtenido a mano es, mucho menos complejo y es con el que nos quedamos en este apartado.

3.2. Arboles de clasificación

En este apartado vamos a tratar los árboles de clasificación, modelos usados para predecir variables de tipo binario como en el caso de la regresión logística binaria.

Los árboles de clasificación segmentan los datos mediante una serie de reglas simples aplicadas jerárquica y secuencialmente, creando segmentos o nodos que contienen subconjuntos de la muestra. Una vez lograda la segmentación "óptima", donde los nodos resultantes son homogéneos en cuanto a la variable dependiente y heterogéneos entre sí, se asigna un valor de predicción a los nodos terminales u hojas, de manera que todas las observaciones en una hoja se predicen con el mismo valor. Este valor de predicción no es más que la proporción observada para cada categoría de la variable independiente, en nuestro caso es la proporción de unos y ceros para la hoja en cuestión.

Durante la creación de estos árboles se pueden ajustar distintos parámetros en función de cómo queramos que sea nuestro árbol. El tamaño de hoja hace referencia al tamaño mínimo que puede tomar una hoja para continuar aplicando reglas de decisión, se indica como un porcentaje sobre el total de observaciones que representa esa hoja. La profundidad del árbol hace referencia a la distancia máxima que puede haber entre el nodo raíz o inicial y la hoja más alejada, un árbol demasiado profundo puede llevar a un sobreajuste que afecte a la capacidad predictiva de nuestro árbol [4].

Utilizaremos por tanto el mismo conjunto de datos y, consecuentemente, seguiremos optando por considerar que un vuelo se retrasa si la proporción de vuelos retrasados en una observación supera o iguala el valor de 0.3. Para poder comparar lo bueno que es este modelo con el generado anteriormente, consideraremos los mismos conjuntos de entrenamiento y prueba para ambos modelos.

A la hora de crear estos árboles de clasificación existen distintos criterios a seguir:

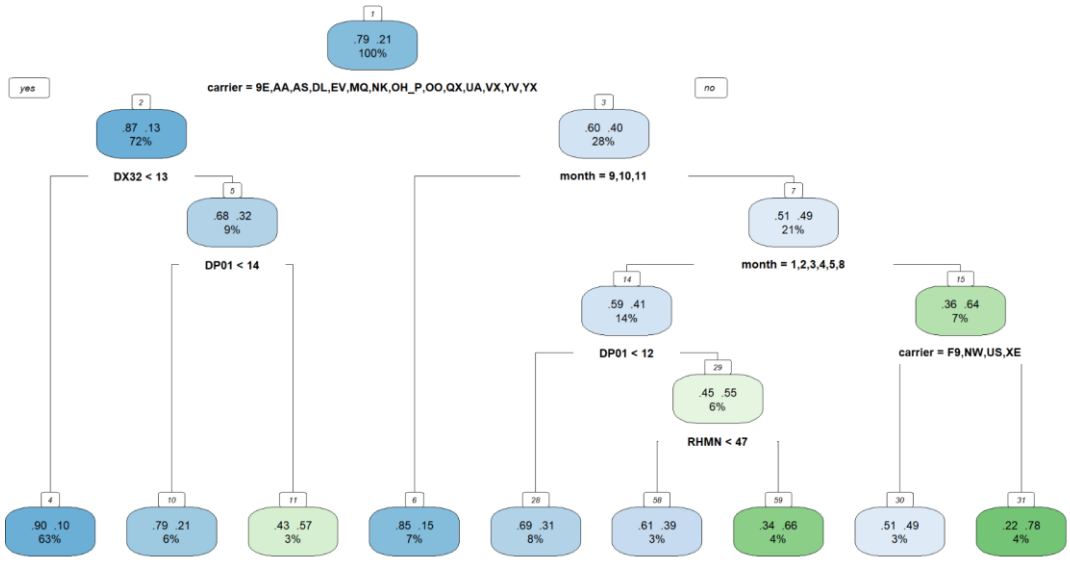
- Índice de Gini: El índice de Gini nos indica cómo de homogéneas son las observaciones de un grupo con respecto a una variable categórica (en nuestro caso la variable *retraso*). Esta es la fórmula de cálculo para cada nodo, donde p_k^j representa la proporción de observaciones de la categoría k en el nodo j, con K número de categorías distintas [4]:

$$IG(nodo_j) = 1 - \sum_{k=1}^{K} (p_k^j)^2 \qquad [5]$$

- Criterio de Entropía: Indica cómo de homogéneo es un grupo de observaciones teniendo en cuenta que en un grupo homogéneo la entropía es cero. Esta es la fórmula de cálculo de la entropía para cada nodo, donde nuevamente p_k^j indica la proporción de observaciones de la categoría k en el nodo j para K número de categorías disintas [4]:

$$E(nodo_j) = -\sum_{k=1}^{K} p_k^j \, log_2(p_k^j) \qquad [6]$$

Cuando se trata de buscar el mejor árbol, es recomendable probar con ambos criterios, por lo que comenzamos probando con el del Índice de Gini y ajustando el tamaño mínimo de hoja al 2%. No necesitamos modificar nada en nuestro conjunto ya que, en el caso de estos modelos, las categóricas no necesitan entrar como dummies.



Gráfica 13 Árbol de decisión obtenido por Gini y tamaño de hoja del 2%

Este diagrama es una representación de nuestro árbol. El color verde indica hojas en las que el 1 (vuelo se retrasa) tiene una mayor proporción de aparición que el 0 (el vuelo no se retrasa). Vemos que hemos obtenido un árbol con 9 hojas y una profundidad igual a 5.

Podemos ver que la primera regla de decisión que toma para ver si un vuelo se retrasa o no, es si el *carrier* pertenece al conjunto de "9E, AA, AS, DL, EV, MQ, NK, OH_P, OO, QX, UA, VX, YV, YX". Vemos que el 72% de las observaciones pertenecen a esos aviones y que el 28% no lo hacen. De este 28%, el 40% se retrasan y del 60% de esa hoja solo se retrasa el 15% (lo vemos en la hoja izquierda de la siguiente decisión).

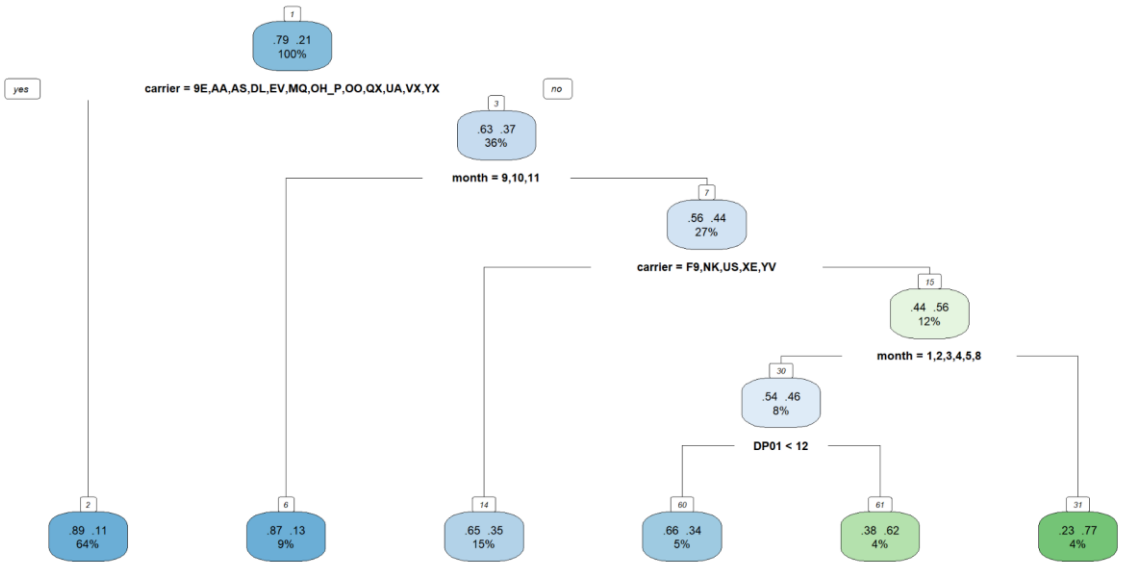
La siguiente regla que vemos que se aplica si se va por el camino de la derecha es si la observación pertenece a los meses 9, 10 y 11. Las observaciones que no pertenecen a este grupo, se retrasan el 49% de las veces.

En las últimas hojas (las de más abajo) podemos ver como se ha clasificado en función de los valores de las variables más importantes.

La hoja con una mayor proporción de vuelos retrasados (78%, la más verde) es la de la derecha del todo. Son vuelos no pertenecientes al conjunto de carriers "9E, AA, AS, DL, EV, MQ, NK, OH_P, OO, QX, UA, VX, YV, YX", que no han sido en "septiembre, octubre o noviembre", tampoco han sido en "enero, febrero, marzo, abril o agosto" y cuyas compañías tampoco pertenecían al conjunto de carriers "F9, NW, US y XE". Dicho de otra manera, vuelos ocurridos en los meses de junio, julio o diciembre cuyas compañías fueron JetBlue Airlines, Continental Airlines, Comair o Southwest Airlines. Esta hoja representa un 4% del total de las observaciones del conjunto.

La hoja con una menor proporción de vuelos retrasados (10%, la más azul) es la de la izquierda del todo. Son vuelos en los que la compañía pertenecía al conjunto "9E, AA, AS, DL, EV, MQ, OO, QX, UA, VX, YV, YX" y el mes en el que ocurrió el vuelo no se superaron los 13 días con temperatura máxima menor o igual a 0 ºC. Representa el 63% del total de observaciones del conjunto.

Si elaboramos el árbol eligiendo un tamaño de hoja mínimo del 2% y siguiendo esta vez el criterio de entropía obtenemos este resultado.



Gráfica 14 Árbol de decisión obtenido por entropía y tamaño de hoja del 2%

La hoja con mayor porcentaje de retrasos tiene un 77% y representa un 4% del total de observaciones de nuestro conjunto. Cumple las condiciones de: no estar incluido en el conjunto de carriers siguiente “9E, AA, AS, DL, EV, MQ, OH_P, OO, QX, UA, VX, YX”, no haber volado en los meses de septiembre, octubre o noviembre, no haber volado tampoco en las compañías “F9, NK, US, XE, YV” y tampoco haber volado en los meses “enero, febrero, marzo, abril, mayo y agosto”. Esto es lo mismo que decir que son vuelos que han volado en los meses de junio, julio o diciembre en las compañías JetBlue Airlines, Continental Airlines, Comair, Northwest Airlines o Southwest Airlines.

La hoja con menor porcentaje de retrasos tiene un 11% y representa un 64% del total de observaciones del conjunto. Son los vuelos cuya compañía pertenece a la siguiente lista “9E, AA, AS, DL, EV, MQ, OH_P, OO, QX, UA, VX, YX”.

Observamos que las hojas con un mayor porcentaje de retrasos para ambos criterios contenían observaciones muy similares en lo referente al mes y compañía de vuelo.

Podemos ahora probar a generar árboles con distintos tamaños mínimos de hoja (1% y 5%) para los dos criterios distintos, compararlos entre sí y con los otros dos que ya hemos analizado en detalle y ver cual ajusta mejor. Los compararemos en función de su AUC y complejidad para datos de entrenamiento y test.

Tabla 30 AUC en entrenamiento y prueba de los distintos árboles generados

	modeloGini1	modeloGini2	modeloGini5	modeloEnt1	modeloEnt2	modeloEnt5
AUC en entrenamiento	0.7949802	0.7495292	0.6915873	0.8702589	0.7195121	0.7145649
AUC en prueba	0.7717748	0.7538413	0.7265054	0.8028784	0.7404449	0.7376276

El mejor AUC para entrenamiento y prueba lo tiene el cuarto modelo, un árbol construido según el criterio de entropía con un tamaño de hoja del 1% y que cuenta con 26 hojas. Como vemos es mucho mejor para entrenamiento que para prueba siendo su AUC 7 puntos porcentuales inferior para el segundo. Este fenómeno se debe al hecho que, al tener tantas hojas, está sobreajustado a los datos con los que ha sido entrenado y predice mucho peor cuando se le dan nuevas observaciones. Además es demasiado complejo y difícil de interpretar al contar con tantas hojas, por lo que es preferible un modelo con menos hojas.

Los modelos generados con un tamaño mínimo del 5% (el tercer y sexto AUC). Tienen valores de área bajo la curva similares tanto para entrenamiento como para prueba, y aunque son ambos inferiores a los de tamaño mínimo de hoja del 2%, tienen ambos únicamente 4 hojas, por lo que parecen preferibles al resto. De estos dos últimos nos quedamos con el generado por el criterio de entropía (el último de la lista) ya que tiene mejores valores de AUC.

Del modelo de tamaño 5% y criterio de entropía podemos obtener más estadísticos resultantes de la clasificación del conjunto de prueba habiendo ya ajustado la proporción de unos y ceros de nuestro conjunto.

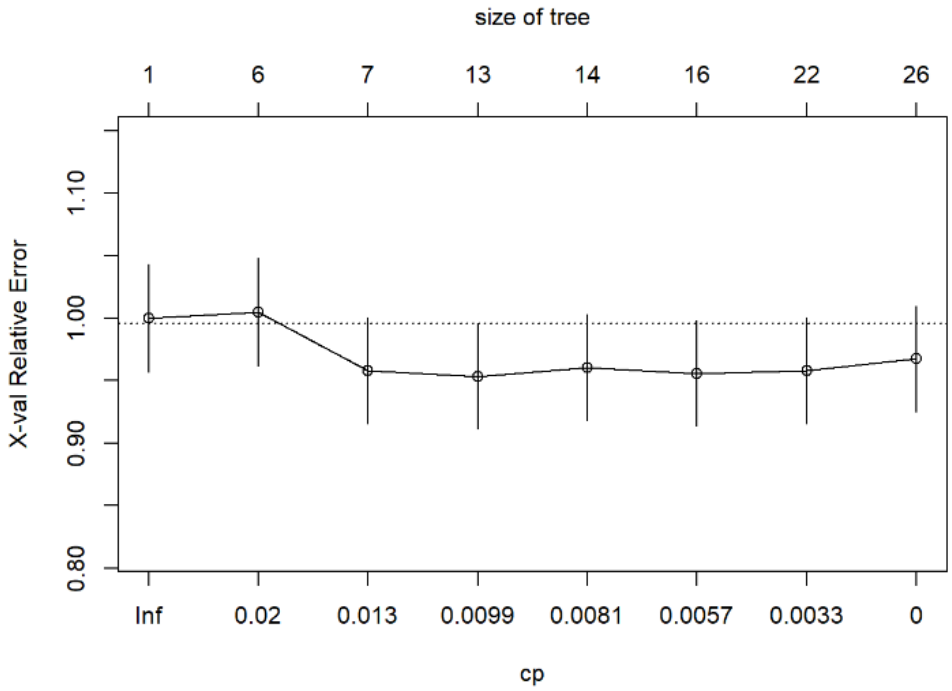
Tabla 31 Medidas de precisión del árbol generado

```
          Reference    Accuracy     Kappa
Prediction   0   1   0.7826923 0.3841962
         0 345  45   Sensitivity Specificity
         1  68  62     0.5794393   0.8353511
```

Tenemos un porcentaje de acierto del 78.27%, una kappa justo según estándares generales (0.21 - 0.4) y una especificidad y sensibilidad no muy balanceadas aún habiendo ajustado el parámetro pertinente.

Un proceso habitual que se sigue cuando tenemos un árbol que tiene muy buenos estadísticos pero que cuenta con muchas hojas es tratar de podarlo, eliminando hojas que sean redundantes y que no aporten al árbol para tratar de reducir también un sobreajuste del que podríamos no habernos dado cuenta [5]. En nuestro caso, aunque nos hemos quedado con un árbol muy sencillo, podemos probar a podar el árbol que mejor AUC tenía para ver si reduciendo su número de hojas podemos acabar con un árbol equiparable en complejidad al que ya tenemos pero que tenga una mayor capacidad predictiva.

Para ello, tomamos el árbol obtenido siguiendo el criterio de entropía con un tamaño de hoja del 1%, generamos la secuencia de valores críticos que da pie al aumento o disminución en tamaño del árbol y la graficamos para ver qué número de hojas es el más adecuado.



Gráfica 15 Valores críticos para el árbol

Parece que no existe una gran diferencia en error relativo entre tomar 6 o tomar 7 hojas, por lo que probamos con ambas podas y sacamos sus áreas bajo la curva para poder compararlas con el árbol generado siguiendo el criterio de entropía y tamaño de hoja del 5% y también el del 1% sin podar.

Tabla 32 Estadísticos número de hojas para las distintas podas

	Árbol Entropía 1%	Poda 1 Entropía 1%	Poda 2 Entropía 1%	Árbol entropía 5%
Número de hojas	26	6	7	4
AUC en prueba	0.8029	0.7404	0.7438	0.7376

Vemos claramente que no hay una gran mejora en el caso de 6 y 7 hojas sobre el árbol que teníamos antes de 4 hojas, por lo que si lo que buscamos es capacidad predictiva y no buscamos ser capaces de interpretar nuestro modelo, deberíamos tomar el árbol con más

hojas, y si lo que buscamos es uno que seamos capaces de interpretar, tomaremos el de criterio de entropía y tamaño mínimo de hoja del 5%.

3.3. Bosques Aleatorios para clasificación

Los bosques aleatorios hacen referencia a una técnica de ensamblado de árboles para formar modelos predictivos potentes con un mayor nivel de complejidad. Descendiente del bagging, este método trata de aprovechar la mayor desventaja que tienen los árboles (su gran variabilidad frente a cambios en el número de variables u observaciones) intentando agregar la información en forma de media, obtenida entre muchos árboles generados de forma distinta, obteniendo así una predicción mejor. Así, en vez de centrarnos en un solo árbol, que puede no estar siendo el mejor, generamos multitud de árboles distintos con la intención de reducir la variabilidad e incluir una mayor información en el proceso.

Para generar estos árboles recurrimos a submuestras obtenidas con reemplazamiento (una para cada árbol) e intentamos generar árboles muy profundos con un tamaño de hoja pequeño ya que, si hacemos árboles con un menor número de hojas, lo más probable es que haya muchos iguales en nuestro bosque y no consigamos el objetivo del bagging.

Los modelos de bosque aleatorio difieren del bagging en que no solo usan submuestras de observaciones con la intención de que los árboles que lo componen sean distintos entre sí, sino que también utilizan subconjuntos aleatorios de variables para las decisiones de cada nodo de cada árbol con la intención de evitar el efecto de "variables dominantes". De no ser por esta faceta de los bosques aleatorios, tenderían a aparecer siempre las mismas variables en los árboles y no tendríamos esa variabilidad que buscamos dentro de nuestro bosque, llevando esto a una pérdida de complejidad y capacidad predictiva [6].

Como seguimos prediciendo una variable binaria, utilizaremos los mismos conjuntos de entrenamiento y prueba que llevamos usando los últimos apartados. Empezamos creando un bosque de 500 árboles con un tamaño mínimo de hoja de 5% y valor por defecto del tamaño de la muestra de variables que se toma para realizar las decisiones en cada nodo (en este caso serán seis las variables tomadas aleatoriamente para cada muestra del conjunto de todas las variables).

```
                Type of random forest: classification
                      Number of trees: 500
No. of variables tried at each split: 6

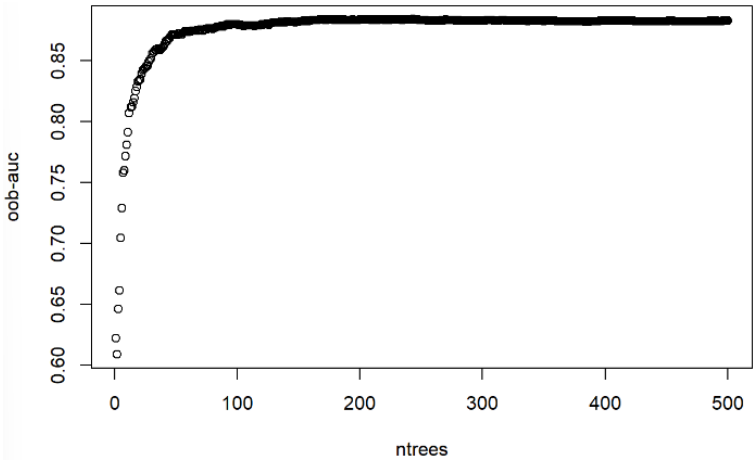
        OOB estimate of  error rate: 17.61%
Confusion matrix:
     0  1 class.error
0 1625 30  0.01812689
1  337 92  0.78554779
```

Estos parámetros nos devuelven un bosque con un error estimado del 17.61%, una especificidad del 98.2% (1-0.018) y una sensibilidad del 21.5% (1-0.78). Seguimos con el problema con el que nos hemos topado en los apartados anteriores, pero como es un tema de la naturaleza del conjunto de datos, no es preocupante. Podemos ver si con otros tamaños de hoja, el error estimado se reduce.

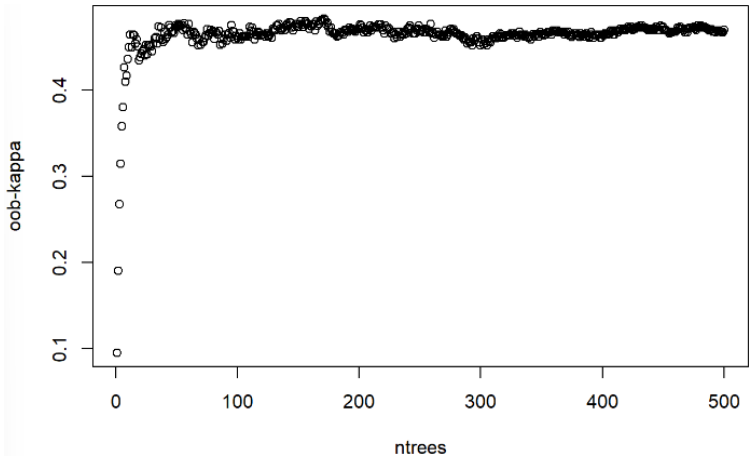
```
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 6

        OOB estimate of  error rate: 14.92%
Confusion matrix:
     0   1 class.error
0 1582  73  0.04410876
1  238 191  0.55477855
```

A partir de un 1% el valor del error estimado deja de reducirse, por lo que nos quedamos con este valor para nuestro bosque final. Pasamos ahora a estudiar cual podría ser el número óptimo de árboles para nuestro bosque.



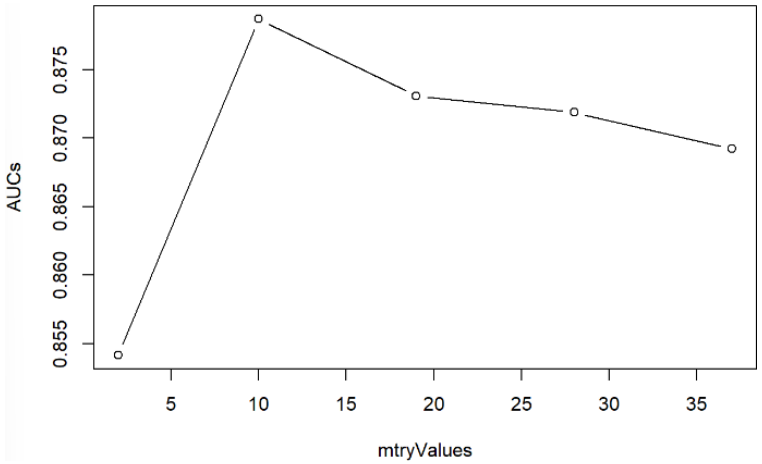
Gráfica 16 Evolución del AUC en función del número de árboles



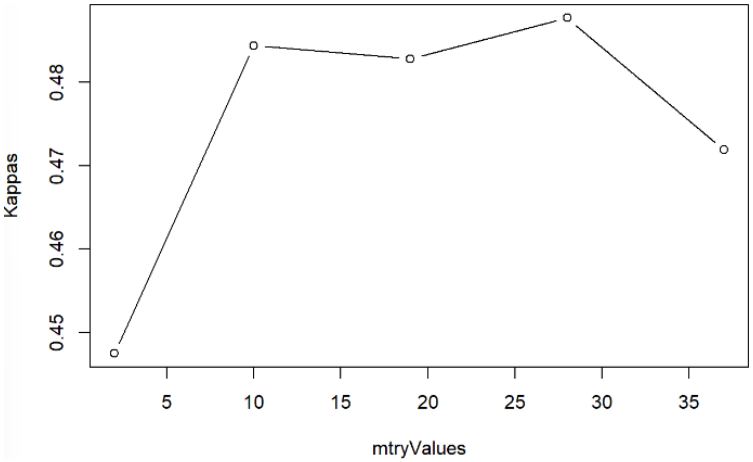
Gráfica 20 Evolución del kappa en función del número de árboles

Para el AUC, su valor se estabiliza a partir de los 100-200 árboles y para la kappa, vemos que es de 100 árboles se estabiliza, por lo que parece preferible un tamaño de 100 árboles para nuestro bosque.

Pasamos ahora a probar disintos tamaños de muestras para el número de variables que se toman a la hora de realizar las decisiones de cada nodo.



Gráfica 21 Evolución del AUC en función del número de variables



Gráfica 17 Evolución del kappa en función del número de variables

La mejor área bajo la curva se alcanza para 10 variables y, aunque la mejor kappa se alcance entre 25 y 30 variables, podemos ver que no difiere en gran manera con la alcanzada para 10 variables, por lo que tomamos este valor como óptimo. Una vez hecho esto veamos cómo queda el modelo ganador.

```
                Type of random forest: classification
                      Number of trees: 100
No. of variables tried at each split: 10

        OOB estimate of  error rate: 14.73%
Confusion matrix:
     0   1 class.error
0 1570  85  0.05135952
1  222 207  0.51748252
```

Hemos logrado reducir el error relativo en 2 décimas de punto porcentual, lo cual no representa una mejora exponencial pero sí que demuestra que siempre se puede obtener una mejor versión del bosque inicialmente generado optimizando sus parámetros.

Pasamos ahora a analizar la calidad predictiva de nuestro bosque, usando el mismo conjunto de prueba que hemos usado para analizar el resto de los modelos de este trabajo y ajustando la proporción de unos y ceros que tiene nuestro conjunto.

Tabla 33 Medidas de precisión del bosque aleatorio generado

```
          Reference   Accuracy     Kappa
Prediction   0   1  0.8019231 0.4879541
         0 335  25
         1  78  82  Sensitivity Specificity
                      0.7663551   0.8111380
```

Vemos que tiene una precisión del 80,19%, una kappa de 0.48 moderadamente buena según los estándares habituales (0.41 - 0.60) y unas capacidades de detectar positivos y negativos bastante altas, siendo la especificidad la más alta con un 81.11% seguida de la sensibilidad con un 76.63%.

3.4. Modelo XGBoost para regresión

Dado que todavía no hemos aplicado ningún modelo de predicción para predecir directamente que probabilidad hay de que nuestro vuelo se retrase (*propRetrasos*), he reservado este problema a, probablemente, uno de los algoritmos más potentes para hacer este tipo de predicciones.

El XGBoost (Extreme Gradient Boosting) es una implementación optimizada y de alto rendimiento del algoritmo de boosting de árboles de decisión. Es ampliamente utilizado en problemas de clasificación y regresión debido a su eficiencia, flexibilidad y precisión. Es una técnica de ensamblaje que combina varios modelos débiles para crear un modelo fuerte, mejorando esta técnica mediante la optimización de la función objetivo utilizando métodos de segunda orden, lo que permite una convergencia más rápida y precisa [7].

Este algoritmo destaca principalmente porque implementa la paralelización del árbol de decisión, lo que mejora significativamente el tiempo de entrenamiento, además incluye regularización tanto L1 (Lasso) como L2 (Ridge), lo que ayuda a prevenir el sobreajuste y mejora la generalización del modelo [8].

Al igual que en el caso de los bosques aleatorios, es un algoritmo de caja negra que no nos permite interpretar como funciona y del cual únicamente podemos obtener métricas de lo capaz que es a la hora de realizar predicciones.

La única pega que tiene este modelo es que solo admite variables contínuas, por lo que podemos barajar tres opciones para nuestras variables cualitativas (*carrier*, *month*, *WDF2* y *WDF5*):

- Eliminarlas del conjunto haciendo la predicción sin ellas.
- Crear variables dummy para cada categoría de cada variable. Estas variables dummy son variables que corresponden a cada categoría de una variable tipo factor y que toman 1 si su variable categórica de referencia ha tomado su valor y 0 si no lo ha hecho. Tenemos tantas variables dummy como categorías hay para cada variable menos una, ya que cuando todas las de una variable tomen valor 0 estaríamos indicando que se está tomando el valor de la que no tiene variable dummy. Este tipo de variables ya las habíamos usado con anterioridad cuando aplicábamos la regresión logística binaria, que las crea automáticamente.
- Sustituir cada categoría de estas variables por el valor medio de la variable objetivo de todas las observaciones para esa categoría.

Tras aplicar en cada caso los cambios pertinentes al conjunto que contiene la variable *propRetrasos* y crear el modelo XGBoost, hemos obtenido unos errores cuadráticos medios mediante la generación de cincuenta conjuntos de entrenamiento y prueba distintos (usando cada vez una semilla de generación distinta) calculando la media de todas esas

repeticiones. Para cada uno hemos elegido el parámetro de profundidad de árbol que mejor ECM (error cuadrático medio) nos daba.

Tabla 34 Medidas de error y profundidad elegidas para los modelos XGBoost

	Sin categóricas	Categóricas sustituidas	Con dummies
Profundidad max.	3	2	3
ECM	0.006362728	0.005381102	0.005373731
RECM	0.079766	0.073356	0.073306

Vemos que nuestros mejores modelos son los que no han optado por eliminar variables, lo cual es lógico ya que estas contienen información predictiva que no nos conviene perder. Podemos afirmar por tanto que en media la diferencia entre los valores reales y los predichos es de aproximadamente 0.073.

Si comparamos este algoritmo con otro más simple como una regresión lineal múltiple usando los mismos conjuntos observamos estos resultados.

Tabla 35 Medidas de error para los modelos XGBoost vs MLR

	XGBoost Sin categóricas	MLR Sin categóricas	XGBoost Categóricas sustituidas	MLR Categóricas sustituidas	XGBoost Con dummies	MLR Con dummies
ECM	0.006362728	0.008548129	0.005381102	0.006806971	0.005373731	0.006662925
RECM	0.079766	0.092456	0.073356	0.082504	0.073306	0.081626

Como podemos observar, XGBoost es ampliamente superior y está completamente justificada su aplicación para la resolución de nuestro problema.

4. Conclusiones

En el estudio que he desarrollado se ha abordado la problemática de predecir los retrasos de vuelos en el Aeropuerto Internacional O'Hare de Chicago, utilizando datos recopilados del Departamento de Transporte de los Estados Unidos (DOT) y del Centro Nacional de Información Ambiental (NCEI). Para ello he desarrollado diversos modelos de predicción de retrasos de vuelos basados en variables meteorológicas, utilizando técnicas de regresión logística binomial, árboles de clasificación, bosques aleatorios y XGBoost.

Para concluir este estudio, nos queda concluir qué modelo de los que hemos desarrollado ha sido mejor para el caso de clasificación de la variable *retraso* y la predicción de la variable *propRetrasos*.

Tabla 36 Capacidad predictiva de los modelos seleccionados al final de cada apartado

TEST	Regresión Logística	Árbol de clasificación	Bosque de clasificación
Precisión	74,39%	78,27%	80,19%
Sensibilidad	74,12%	57,94%	76,63%
Especificidad	74,46%	83,53%	81,11%

El mejor modelo de clasificación ha acabado siendo el más complejo, como era de esperar. El bosque aleatorio propuesto tiene un porcentaje de acierto en sus predicciones del 80,19% siendo 2 y 6 puntos porcentuales superior a las dos otras alternativas. A parte de eso, tiene la sensibilidad y la especificidad bastante balanceadas siendo esta última equivalente al 81,11%. Aunque en este último valor es inferior al del árbol de clasificación, es ampliamente superior en el resto de valores siendo su sensibilidad superior en casi 20 puntos porcentuales.

El bosque aleatorio que mejor comportamiento ha tenido, además, contaba con un área bajo la curva superior al 87.5% y un índice kappa de 0.487 moderadamente bueno.

Tabla 37 Error cuadrático medio cometido por cada modelo XGBoost

	Sin categóricas	Categóricas sustituidas	Con dummies
Profundidad max.	3	2	3
ECM	0.006362728	0.005381102	0.005373731
RECM	0.079766	0.073356	0.073306

Los modelos que mejor han resuelto el problema de regresión de la probabilidad de retraso de un vuelo mediante el algoritmo XGBoost son los obtenidos usando tanto el conjunto que implementa el uso de variables dummy para las variables categóricas como el conjunto que opta por sustituir para las categorías de cada variable el valor medio de la variable objetivo para cada una de esas categorías.

5. Bibliografía

1.- Web oficial del departamento de transporte de los Estados Unidos.(https://www.transtats.bts.gov/OT_Delay/OT_DelayCause1.asp?20=E, All carriers Chicago O'Hare International (01/06/2003,01/11/2023))

2.- Web official Centro Nacional de Información ambiental NCEI, https://www.ncei.noaa.gov/data/global-summary-of-the-month/access/, USW00094846.csv).

3.- HASTIE, T., TIBSHIRANI, R,, FRIEDMAN, J. (2001) The elements of Statistical Learning Data Mining, Inference and Prediction

4.- BREIMAN, L. FRIEDMAN, J.H., OLSHEN, R.A. STONE, C.J. (1998) Classification and Regression Trees. Chapman & Hall/CRC, Boca Raton

5.- KUHN, M., & JOHNSON, K. (2013). Applied predictive modeling (Vol. 26). New York: Springer

6.- JAMES, G., WITTEN, D., HASTIE, T., & TIBSHIRANI, R. (2013). An introduction to statistical learning with Applications in R. New York: Springer

7.- BRADLEY BOEHMKE & BRANDON GREENWELL (2020) Hands-On Machine Learning with R

8.- YUNG-CHIA C., KUEI-HU C., GUAN-JHIH W. (2018). Application of eXtreme gradient boosting trees in the construction of credit risk assessment models for financial institutions

9.- YUEMIN TANG. (2021) Airline Flight Delay Prediction Using Machine Learning Models (https://dl.acm.org/doi/fullHtml/10.1145/3497701.3497725)

10.- S. CHOI, Y. J. KIM, S. BRICENO AND D. MAVRIS (2016). Prediction of weather-induced airline delays based on machine learning algorithms

Anexo 1: Listado completo de variables

Tras unir los conjuntos de datos de retrasos de vuelos en el aeropuerto Chicago O'Hare International con el conjunto de datos que contiene las lecturas de una estación meteorológica de este mismo aeropuerto, usando como claves el mes y el año de cada observación, obtenemos un conjunto final que incluye las siguientes variables:

Variables mes y año

Tenemos las lecturas de los distintos vuelos que pasaron por el aeropuerto Chicago O'Hare International cada mes desde el año 2006 hasta noviembre de 2023 y el tiempo que hizo cada mes de dichas observaciones.

Variables retraso

Son las variables que originalmente pertenecen al conjunto de datos sobre retrasos de vuelos en el aeropuerto Chicago.

Carrier: contiene un código alfanumérico de 2 caracteres indicando la aerolínea de la que se trata. Nos quedamos con esta variable en vez de la que contiene el nombre completo de la aerolínea por comodidad a la hora de codificarla.

Arr_flights: variable numérica que representa el número total de llegadas de la aerolínea al aeropuerto de Chicago durante el mes y año de la observación.

Arr_del15: número total de vuelos que se ha retrasado del total de vuelos de esa aerolínea para esa observación. Se considera retraso a cualquier situación en la que el avión llegue más de 15 minutos tarde de la fecha prevista para ese vuelo.

carrier_ct: número de vuelos en los que la culpa del retraso la tiene el estado del avión (por ejemplo por falta de combustible o reparaciones). Es un subset de *Arr_del15*.

weather_ct: número de vuelos en los que la razón del retraso es el estado meteorológico. Es un subset de *Arr_del15*.

nas_ct: número de vuelos en los que la culpa del retraso la tiene es sistema nacional de aviación (National Aviation System). Es un subset de *Arr_del15*.

security_ct: número de vuelos en los que el retraso es debido a algún tema de seguridad del aeropuerto. Es un subset de *Arr_del15*.

late_aircraft_ct: número de vuelos en los que el retraso es consecuencia directa del retraso del un vuelo anterior que usaba el mismo avión. Es un subset de Arr_del15.

arr_cancelled: número de vuelos cancelados.

arr_diverted: número de vuelos desviados a otro aeropuerto.

arr_delay: suma total de minutos de retraso para todos los vuelos de esa observación que se consideran como retrasos.

Carrier_delay: minutos de retraso que se deben al estado o gestión del avión. Es un subset de *arr_delay*.

Weather_delay: minutos de retraso que se deben al tiempo atmosférico. Es un subset de *arr_delay*.

Nas_delay: minutos de retraso debidos a la organización del sistema nacional de aviación (National Aviation System). Es un subset de *arr_delay*.

Security_delay: minutos de retraso debido a alguna gestión de seguridad. Es un subset de *arr_delay*.

Late_aircraft_delay: minutos de retraso debidos al retraso de un vuelo anterior que usó el mismo avión. Es un subset de *arr_delay*.

Variables tiempo atmosférico

Son variables que originalmente pertenecen al conjunto de datos que contiene las lecturas de una estación meteorológica de ese mismo aeropuerto de Chicago. Todas estas variables son de tipo numérico ya sean enteros o con decimales.

ADPT: Temperatura Promedio Mensual del Punto de Rocío (ºC).

ASLP: Presión Promedio Mensual a Nivel del Mar (hPa).

ASTP: Presión Promedio Mensual a Nivel de Estación (hPa).

AWBT: Temperatura Promedio Mensual del Termómetro Húmedo (ºC).

AWND: Velocidad del Viento Promedio Mensual (mi/h).

CDSD: Grados día de refrigeración (desde el inicio de la temporada). Total acumulado de días de refrigeración mensuales hasta el final del mes más reciente. Cada mes se suma para producir un total de la temporada hasta la fecha. La temporada comienza en enero en el Hemisferio Norte y en julio en el Hemisferio Sur.

CLDD: Grados día de refrigeración. Se calcula cuando la temperatura media diaria es más de 18.3 grados Celsius.

DP01: Número de días en el mes con más de 0.254 milímetros de lluvia, siendo un milímetro de lluvia equivalente a un litro de agua por metro cuadrado.

DP10: Número de días en el mes con más de 2.54 milímetros de lluvia.

DP1X: Número de días en el mes con más de 25.4 milímetros de lluvia.

DSND: Número de días del mes con una profundidad de nieve superior a 25 milímetros.

DSNW: Número de días con nevadas de una profundidad mayor a 25 milímetros.

DT00: Número de días con temperatura máxima menor o igual a -17.8 grados Celsius.

DT32: Número de días con temperatura mínima menor o igual a 0 grados Celsius.

DX32: Número de días con temperatura máxima menor o igual a 0 grados Celsius.

DX70: Número de días con temperatura máxima mayor o igual a 21.1 grados Celsius.

DX90: Número de días con temperatura máxima mayor o igual a 32.2 grados Celsius

DYFG: Número de días con niebla

DYHF: Número de días con niebla densa (visibilidad inferior a 1/4 milla)

DYNT: Número de días con temperatura máxima >= 90 grados Fahrenheit/32.2 grados Celsius

DYSD: Número de días con temperatura máxima mayor o igual a 32.2 grados Celsius

DYSN: Número de días con temperatura máxima mayor o igual a 32.2 grados Celsius

DYTS: Número de días con tormentas eléctricas

DYXP: Número de días con temperatura máxima mayor o igual a 32.2 grados Celsius

DYXT: Número de días con temperatura máxima mayor o igual a 32.2 grados Celsius

EMNT: Temperatura mínima extrema para el mes. La temperatura mínima diaria más baja para el mes (ºC).

EMSD: Mayor profundidad diaria de nieve en el mes (mm).

EMSN: Mayor nevada diaria en el mes (mm).

EMXP: Total diario más alto de precipitación en el mes (mm).

EMXT: Temperatura máxima extrema del mes. Temperatura máxima diaria más alta para el mes (ºC).

EVAP: Evaporación mensual total (mm).

HDSD: Grados día de calefacción (temporada hasta la fecha). Total acumulado de días de calefacción mensuales hasta el final del mes más reciente. Cada mes se suma para producir un total de la temporada hasta la fecha. La temporada comienza en Julio en el Hemisferio Norte y en Enero en el Hemisferio Sur.

HNyz: Temperatura mínima del suelo más alta para el mes (ºC).

HTDD: Grados día de calefacción. Calculado cuando la temperatura promedio diaria es inferior a 18.3 grados Celsius.

HXyz: Temperatura máxima del suelo más alta para el mes (ºC).

LNyz: Temperatura mínima del suelo más baja para el mes (ºC).

LXyz: Temperatura máxima del suelo más baja para el mes (ºC).

MNPN: Temperatura mínima media mensual del agua de la bandeja de evaporación (ºC).

MNyz: Media mensual de la temperatura mínima diaria del suelo (ºC).

MXPN: Temperatura máxima media mensual del agua de la bandeja de evaporación (ºC).

MXyz: Media mensual de la temperatura máxima diaria del suelo (ºC).

PRCP: Total de precipitación mensual (mm).

PSUN: Promedio mensual de los porcentajes diarios de luz solar.

RHAV: Promedio mensual de la humedad relativa (%).

RHMN: Promedio mensual de la humedad relativa mínima (%).

RHMX: Promedio mensual de la humedad relativa máxima (%).

SNOW: Cantidad total mensual de nieve(mm).

TAVG: Temperatura promedio mensual (ºC).

TMAX: Temperatura máxima mensual (ºC).

TMIN: Temperatura mínima mensual (ºC).

TSUN: Total diario de horas de sol.

WDF1: Dirección del viento de 1 minuto de duración más rápido. Dado en direcciones de los puntos cardinales en grados (0º/360º = norte, 90º = oeste, 180º = sur y 270º = este).

WDF2: Dirección del viento de 2 minutos de duración más rápido.

WDF5: Dirección del viento de 5 segundos de duración más rápido.

WDFG: Dirección del viento para la velocidad máxima de ráfaga de viento.

WDFI: Dirección de la velocidad del viento instantáneo (1 a 3 segundos) más alta.

WDFM: Dirección del viento para la velocidad máxima del viento.

WDMV: Movimiento total del viento mensual sobre la bandeja de evaporación (mi).

WSF1: Velocidad máxima de los vientos de 1 minuto más rápidos (mi/h).

WSF2: Velocidad máxima de los vientos de 2 minutos más rápidos (mi/h).

WSF5: Velocidad máxima de los vientos de 5 segundos más rápidos (mi/h).

WSFG: Velocidad máxima de ráfaga de viento (mi/h).

WSFI: Velocidad máxima del viento instantáneo (1 a 3 segundos) para el mes (mi/h).

WSFM: Velocidad máxima del viento (mi/h).

Nuestro conjunto de datos cuenta con 2604 observaciones, cada lectura representa los vuelos de una compañía para cada mes y año que abarca el conjunto (de junio de 2003 a noviembre de 2023).

Anexo 2: Código de SAS utilizado para el trabajo

```
/*importo los datos sin las variables attributes*/
proc import out=datos_importados
   datafile =
"C:\Users\molpe\OneDrive\Escritorio\TFG_VUELOS\CHICAGO\datosUnidosSinAttrib.xlsx"
   dbms=xlsx;
   getnames=YES;
run;
data datos;
set datos_importados;
run;
data datos;
set datos;
id=_N_;
fecha=mdy(month,'01',year);
propRetrasos=arr_del15/arr_flights;
propCancelados=arr_cancelled/arr_flights;
run;
/*proc print data=datos;
run;*/
proc print data=datos (obs=10);
run;

data tiemp;
set datos;
drop carrier arr_flights arr_del15 carrier_ct weather_ct nas_ct security_ct
late_aircraft_ct arr_cancelled arr_diverted arr_delay carrier_delay
weather_delay nas_delay security_delay late_aircraft_delay id;
run;
proc sort data=tiemp out=tiempo noduprecs;
by year month;
run;

data tiempomonth;
set tiempo;
run;
proc sort data=tiempomonth;
by month;
run;

data tiempoyear;
set tiempo;
run;
proc sort data=tiempoyear;
by year;
run;

/*Variable month*/

proc sort data=datos;
by month;
run;
proc means data=datos sum noprint;
var arr_flights;
by month;
output out=vuelosMes sum=Total;
run;
data suma;
set vuelosMes;
drop _type_ _freq_;
```

```
run;
proc sort data=suma;
by descending Total;
run;
proc print data=suma;
run;
proc sgplot data=suma;
vbar month/ response=Total;
run;

data datos;
set datos;
propRetrasos=arr_del15/arr_flights;
run;
proc means data=datos mean noprint;
var propRetrasos;
by month;
output out=retrasosMes mean=PctRetraso;
run;
data media;
set retrasosMes;
drop _type_ _freq_;
run;
proc sort data=media;
by descending PctRetraso;
run;
proc print data=media;
run;
proc sgplot data=media;
vbar month/ response=PctRetraso;
run;

data datos;
set datos;
propWeather=Weather_delay/arr_delay;
run;
proc means data=datos mean noprint;
var propWeather;
by month;
output out=weatherMes mean=PctRetrasoWeather;
run;
data media;
set weatherMes;
drop _type_ _freq_;
run;
proc sort data=media;
by descending PctRetrasoWeather;
run;
proc print data=media;
run;
/*Variable carrier*/
/*freq de los carriers*/
proc freq data=datos;
tables carrier ;
run;
proc sort data=datos;
by carrier;
run;
proc sgplot data=datos;
vbar carrier;
run;
```

```
proc sort data=datos;
by carrier;
run;
proc means data=datos sum noprint;
var arr_flights;
by carrier;
output out=vuelosCarrier sum=Total;
run;
data suma;
set vuelosCarrier;
drop _type_ _freq_;
run;
proc sort data=suma;
by descending Total;
run;
proc print data=suma;
run;
proc sgplot data=suma;
vbar carrier/ response=Total;
run;


proc means data=datos mean noprint;
var propRetrasos;
by carrier;
output out=retrasosCarrier mean=PctRetraso;
run;
data media;
set retrasosCarrier;
drop _type_ _freq_;
run;
proc sort data=media;
by descending PctRetraso;
run;
proc print data=media;
run;
proc sgplot data=media;
vbar carrier/ response=PctRetraso;
run;

proc means data=datos mean noprint;
var propCancelados;
by carrier;
output out=canceladosCarrier mean=PctCancelacion;
run;
data media;
set canceladosCarrier;
drop _type_ _freq_;
run;
proc sort data=media;
by descending PctCancelacion;
run;
proc print data=media;
run;
proc sgplot data=media;
vbar carrier/ response=PctCancelacion;
run;
/*Para carrier categorías de retraso*/
data datos;
set datos;
```

```
propEstadoGestionAvion=Carrier_delay/arr_delay;
propSeguridad=Security_delay/arr_delay;
propVueloAnterior=Late_aircraft_delay/arr_delay;
run;

proc means data=datos mean noprint;
var propEstadoGestionAvion propSeguridad propVueloAnterior;
by carrier;
output out=retrasosCarrier mean=pctEstadoGestion pctSeguridad pctVueloAnterior;
run;
data media;
set retrasosCarrier;
drop _type_ _freq_;
run;

proc sort data=media;
by descending pctEstadoGestion;
run;
proc print data=media;
var carrier pctEstadoGestion;
run;

proc sort data=media;
by descending pctSeguridad;
run;
proc print data=media;
var carrier pctSeguridad;
run;

proc sort data=media;
by descending pctVueloAnterior;
run;
proc print data=media;
var carrier pctVueloAnterior;
run;

/**/
proc means data=datos;
var ADPT ASLP ASTP AWBT AWND CDSD CLDD DP01 DP10 DP1X DSND DSNW DT00
DT32 DX32
DX70 DX90 EMNT EMSD EMSN EMXP EMXT HDSD HTDD PRCP
RHAV RHMN RHMX SNOW TAVG TMAX TMIN WDF2 WDF5 WSF2 WSF5;
run;

/*Variables temperatura*/
proc sort data=datos;
by year month;
run;
proc means data=datos mean;
var TAVG TMAX TMIN;
run;
proc gplot data=datos;
plot (EMNT TAVG TMAX TMIN EMXT)*fecha / overlay legend=legend1 vaxis=axis1;
symbol i=join;
axis1 label=("Temperatura");
legend1 position=bottom;
run;
/*Bajas*/
proc gplot data=datos;
plot (EMNT TMIN)*fecha / overlay legend=legend1 vaxis=axis1;
symbol i=join;
```

```
axis1 label=("Temperatura");
legend1 position=bottom;
run;
/*altas*/
proc gplot data=datos;
plot (EMXT TMAX)*fecha / overlay legend=legend1 vaxis=axis1;
symbol i=join;
axis1 label=("Temperatura");
legend1 position=bottom;
run;
/*medias*/
proc gplot data=datos;
plot (TAVG)*fecha / overlay legend=legend1 vaxis=axis1;
symbol i=join;
axis1 label=("Temperatura");
legend1 position=bottom;
run;
/*medias por mes de las extremas*/
proc sort data=datos;
by month;
run;
proc means data=datos mean;
var EMXT EMNT;
by month;
output out=medias mean= mediaEMXT mediaEMNT;
run;
data medias;
set medias;
drop _TYPE_ _FREQ_;
run;
proc print data=medias;
run;
/*Variables presion*/
proc sort data=datos;
by month;
run;
proc means data=datos mean noprint;
var ASLP ASTP;
by month;
output out=media mean=mediaASLP mediaASTP;
run;
data media;
set media;
drop _type_ _freq_;
run;
proc print data=media;
run;

proc sort data=datos;
by year month;
run;
proc means data=datos mean;
var ASLP ASTP;
run;
proc gplot data=datos;
plot (ASLP ASTP)*fecha / overlay legend=legend1 vaxis=axis1;
symbol i=join;
axis1 label=("Presión");
legend1 position=bottom;
run;
```

```
/*Variables velocidad del viento*/
proc sort data=datos;
by month;
run;
proc means data=datos mean noprint;
var AWND WSF2 WSF5;
by month;
output out=media mean=mediaAWND mediaWSF2 mediaWSF5;
run;
data media;
set media;
drop _type_ _freq_;
run;
proc print data=media;
run;
proc gplot data=media;
plot (mediaAWND mediaWSF2 mediaWSF5)*month/overlay legend;
symbol i=join;
run;
proc means data=datos mean;
var AWND WSF2 WSF5;
run;

proc sort data=datos;
by year month;
run;
proc gplot data=datos;
plot (AWND WSF2 WSF5)*fecha / overlay legend=legend1 vaxis=axis1;
symbol i=join;
axis1 label=("Velocidad");
legend1 position=bottom;
run;

/*Variables dirección del viento*/
data direcViento;
set datos;
if WDF5>315 then dirWDF5="Norte";
if WDF5<45 then dirWDF5="Norte";
if WDF5>45 and WDF5<135 then dirWDF5="Oeste";
if WDF5>135 and WDF5<225 then dirWDF5="Sur";
if WDF5>225 and WDF5<315 then dirWDF5="Este";
if WDF2>315 then dirWDF2="Norte";
if WDF2<45 then dirWDF2="Norte";
if WDF2>45 and WDF2<135 then dirWDF2="Oeste";
if WDF2>135 and WDF2<225 then dirWDF2="Sur";
if WDF2>225 and WDF2<315 then dirWDF2="Este";
keep year month carrier WDF5 WDF2 dirWDF5 dirWDF2;
run;
proc freq data=direcViento;
tables dirWDF5 dirWDF2;
run;

proc sort data=datos;
by year month;
run;
proc gplot data=datos;
plot (WDF2 WDF5)*fecha / overlay legend=legend1 vaxis=axis1;
symbol i=join;
axis1 label=("Dirección");
legend1 position=bottom;
run;
```

```
/*Variables grados día de refrigeración*/
proc sort data=datos;
by month;
run;
proc means data=datos mean;
var CDSD CLDD;
by month;
output out=medias mean=mediaCDSD mediaCLDD;
run;
data medias;
set medias;
drop _TYPE_ _FREQ_;
run;
proc print data=medias;
run;

proc sort data=datos;
by year month;
run;
proc gplot data=datos;
plot (CDSD CLDD)*fecha / overlay legend=legend1 vaxis=axis1;
symbol i=join;
axis1 label=("Grados día");
legend1 position=bottom;
run;

/*Variables grados día de calefacción*/
proc sort data=datos;
by month;
run;
proc means data=datos mean;
var HDSD HTDD;
by month;
output out=medias mean=mediaHDSD mediaHTDD;
run;
data medias;
set medias;
drop _TYPE_ _FREQ_;
run;
proc print data=medias;
run;

proc sort data=datos;
by year month;
run;
proc gplot data=datos;
plot (HDSD HTDD)*fecha / overlay legend=legend1 vaxis=axis1;
symbol i=join;
axis1 label=("Grados día");
legend1 position=bottom;
run;

/*Variables humedad*/
proc means data=datos mean;
var RHAV RHMN RHMX;
run;

proc sort data=datos;
by month;
run;
```

```
proc means data=datos mean;
var RHAV;
by month;
output out=medias mean=mediaRHAV;
run;
data medias;
set medias;
drop _TYPE_ _FREQ_;
run;
proc print data=medias;
run;

proc sort data=datos;
by year month;
run;
proc gplot data=datos;
plot (RHAV RHMN RHMX)*fecha / overlay legend=legend1 vaxis=axis1;
symbol i=join;
axis1 label=("Humedad");
legend1 position=bottom;
run;

Proc means data=datos mean;
var ADPT AWBT;
run;

proc sort data=datos;
by month;
run;
proc means data=datos mean;
var ADPT;
by month;
output out=medias mean=mediaADPT;
run;
data medias;
set medias;
drop _TYPE_ _FREQ_;
run;
proc print data=medias;
run;

proc sort data=datos;
by month;
run;
proc means data=datos mean;
var AWBT TAVG;
by month;
output out=medias1 mean=mediaAWBT mediaTEMP;
run;
data medias1;
set medias1;
drop _TYPE_ _FREQ_;
run;
proc print data=medias1;
run;

proc sort data=datos;
by year month;
run;
proc gplot data=datos;
plot (AWBT TAVG)*fecha / overlay legend=legend1 vaxis=axis1;
```

```
symbol i=join;
axis1 label=("Temperatura");
legend1 position=bottom;
run;

proc sort data=datos;
by year month;
run;
proc gplot data=datos;
plot (ADPT AWBT)*fecha / overlay legend=legend1 vaxis=axis1;
symbol i=join;
axis1 label=("Temperatura");
legend1 position=bottom;
run;

/*Variables precipitación*/
proc sort data=datos;
by month;
run;
proc means data=datos mean noprint;
var EMXP PRCP;
by month;
output out=media mean=mediaEMXP mediaPRCP;
run;
data media;
set media;
drop _type_ _freq_;
run;
proc print data=media;
run;

proc sort data=datos;
by year month;
run;
proc gplot data=datos;
plot (EMXP PRCP)*fecha / overlay legend=legend1 vaxis=axis1;
symbol i=join;
axis1 label=("Precipitación");
legend1 position=bottom;
run;

/*Variables profundidad de la nieve*/
proc sort data=datos;
by month;
run;
proc means data=datos mean noprint;
var EMSD EMSN SNOW;
by month;
output out=media mean=mediaEMSD mediaEMSN mediaSNOW;
run;
data media;
set media;
drop _type_ _freq_;
run;
proc print data=media;
run;

proc sort data=datos;
by year month;
run;
proc gplot data=datos;
```

```
plot (EMSD EMSN SNOW)*fecha / overlay legend=legend1 vaxis=axis1;
symbol i=join;
axis1 label=("Profundidad");
legend1 position=bottom;
run;

/*Variables número de días*/
/*temp*/
proc sort data=datos;
by month;
run;
proc means data=datos mean noprint;
var DT00 DT32 DX32 DX70 DX90 DYNT;
by month;
output out=media mean=mediaDT00 mediaDT32 mediaDX32 mediaDX70 mediaDX90
mediaDYNT;
run;
data media;
set media;
drop _type_ _freq_;
run;
proc print data=media;
run;

proc sort data=datos;
by year;
run;
proc means data=datos mean noprint;
var DT00 DT32 DX32 DX70 DX90 DYNT;
by year;
output out=media mean=mediaDT00 mediaDT32 mediaDX32 mediaDX70 mediaDX90
mediaDYNT;
run;
data media;
set media;
drop _type_ _freq_;
run;
proc print data=media;
run;
/*Lluvia*/

proc sort data=datos;
by month;
run;
proc means data=datos mean noprint;
var DP01 DP10 DP1X;
by month;
output out=media mean=mediaDP01 mediaDP10 mediaDP1X;
run;
data media;
set media;
drop _type_ _freq_;
run;
proc print data=media;
run;

proc sort data=datos;
by year;
run;
proc means data=datos mean noprint;
var DP01 DP10 DP1X;
```

```
by year;
output out=media mean=mediaDP01 mediaDP10 mediaDP1X;
run;
data media;
set media;
drop _type_ _freq_;
run;
proc print data=media;
run;

/*Nieve*/

proc sort data=datos;
by month;
run;
proc means data=datos mean noprint;
var DSND DSNW;
by month;
output out=media mean=mediaDSND mediaDSNW;
run;
data media;
set media;
drop _type_ _freq_;
run;
proc print data=media;
run;

proc sort data=datos;
by year;
run;
proc means data=datos mean noprint;
var DSND DSNW;
by year;
output out=media mean=mediaDSND mediaDSNW;
run;
data media;
set media;
drop _type_ _freq_;
run;
proc print data=media;
run;

proc sort data=datos;
by year month;
run;
proc gplot data=media;
plot (mediaDSND mediaDSNW)*year/overlay;
symbol i=join;
run;
```

Anexo 3: Código de R utilizado para el trabajo

```
```{r,include=FALSE}
knitr::opts_chunk$set(echo=TRUE,eval=TRUE,message=FALSE,warning=FALSE)
```
```

Librerias

```
```{r,warning=FALSE}
library(dplyr)
library(openxlsx)#Para guardar un conjunto en formato xlsx
library(stargazer)
library(fastDummies)
library(cluster)
library(StatMatch)
library(psych)
library(corrplot)
library(pheatmap)
library(smacof)
library(DescTools)
library(questionr)
library(ClustOfVar)
library(OneR)
library(nnet)
library(car)
library(caret)
library(pROC)
library(MLmetrics)
library(MASS)
library(corrplot)
library(vcd)
library(caret)
library(naivebayes)
library(klaR)
library(rpart)
library(rpart.plot)
library(rattle)
library(randomForest)
library(OOBCurve)
library(xgboost)
library(fastDummies)
library(MASS)
# Para paralelizar
library(doParallel)
cluster <- makeCluster(detectCores() - 1)
registerDoParallel(cluster)
```
```

Depuración de los datos

Lectura de los datos

```{r}
vuelos <-
read.csv("C:/Users/molpe/OneDrive/Escritorio/TFG_VUELOS/CHICAGO/Airline_Delay_OHare.csv")
tiempo <-
readxl::read_excel("C:/Users/molpe/OneDrive/Escritorio/TFG_VUELOS/CHICAGO/datos_weather/TiempoChicago.xlsx")
```

Elimino y tranformo variables del conjunto de tiempo

(Station,Latitude,Longitude,Elevation,Name) Son siempre iguales no aportan

Creo variables mes y año a partir de la variable fecha

```{r}
vuelos<-vuelos[rev(row.names(vuelos)),]#Para que empiece por los datos más antiguos
tiempo<-tiempo[,c(-1,-3,-4,-5,-6)]
year<-as.integer(format(tiempo$DATE,"%Y"))
month<-as.integer(format(tiempo$DATE,"%m"))
tiempo<-cbind(year,month,tiempo[,-1])
```

Me quedo con los años con menos observaciones faltantes

```{r}
tiempo<-tiempo %>%
  filter(year>2005)
tiempo<-tiempo[c(-216,-217),]
```

Elimino variables del conjunto de vuelos

(Carrier_name,airport,airport_name)

Antes de hacerlo cambio los códigos (variable carrier) para las aerolíneas PSA y Comair ya que ambas tienen el código IATA "OH"

```{r}
for(i in (1:nrow(vuelos))){
  if(vuelos[i,]$carrier_name=="PSA Airlines Inc."){
    vuelos[i,]$carrier="OH_P"
  }
  if(vuelos[i,]$carrier_name=="Comair Inc."){
    vuelos[i,]$carrier="OH_C"
  }
}
```

```{r}
vuelos<-vuelos[,c(-4,-5,-6)]
```

```
vuelos$carrier<-as.factor(vuelos$carrier)
```

Me quedo con los valores desde el año 2006 (como en el de tiempo)

```{r}
vuelos<-vuelos %>%
 filter(year>2005)
```

Ahora uno ambos conjuntos en uno para poder empezar a trabajar en él

```{r}
datos<-vuelos %>%
 inner_join(tiempo,by=c("year","month"))
```

Elimino las variables con más NA

```{r}
colSums(is.na(datos))
naS <- colSums(is.na(datos))
variablesQueMantengo <- which(naS <= 0.05*2632)
datos <- datos[, variablesQueMantengo]
colSums(is.na(datos))
```

Voy a ver si las que tienen faltantes son siempre en la misma observación

```{r}
a<-which(is.na(datos[, "RHAV_ATTRIBUTES"]))
b<-which(is.na(datos[, "RHMN"]))
c<-which(is.na(datos[, "RHMX"]))
d<-which(is.na(datos[, "DYFG"]))
e<-which(is.na(datos[, "ASLP"]))
f<-which(is.na(datos[, "ADPT"]))
g<-which(is.na(datos[, "ASTP"]))
a == b
a == c
a == d
a == e
a == f
a == g
```

Elimino dichas observaciones y la variable DYFG ya que tiene tambien unos cuantos faltantes

```{r}
datos<-datos[-a,]
datos$DYFG<-NULL
colSums(is.na(datos))
```

```

Hago lo mismo con las variables que pertenecían al conjunto de vuelos

```{r}
a<-which(is.na(datos[,"arr_del15"]))
datos<-datos[-a,]
colSums(is.na(datos))
```

Falta formatear algunas de las variables para que tomen sus valores reales

```{r}
datos$TAVG<-datos$TAVG/100
datos$TMAX<-datos$TMAX/100
datos$TMIN<-datos$TMIN/100
datos$EMNT<-datos$EMNT/10
datos$EMXT<-datos$EMXT/10
datos$CDSD<-datos$CDSD/10
datos$CLDD<-datos$CLDD/10
datos$HDSD<-datos$HDSD/10
datos$HTDD<-datos$HTDD/10
```

Ya podemos trabajar con un conjunto de datos con cero observaciones faltantes habiendo solo perdido 28 observaciones de las 2632 que teníamos (y unas cuantas variables)

```{r}
datos$carrier<-as.character(datos$carrier)
datos$carrier<-as.factor(datos$carrier)
```

Guardo este conjunto en formato csv en mi ordenador

```{r}
write.xlsx(datos,file="C:/Users/molpe/OneDrive/Escritorio/TFG_VUELOS/CHICAGO/datos Unidos.xlsx")
```

Despues de mirar con detenimiento las variables attributes en SAS

Elimino las variables attributes ya que realemente no contienen ninguna información que vaya a aportar información predictiva al modelo

```{r}
indices <- names(datos)[grep("ATTRIBUTES", names(datos))]

datos <- datos[, !names(datos) %in% indices]

```
```

Convierto todas las variables a formato numérico ya que hay algunas con formato caracter

```{r}
datos_antiguos<-datos
datos <- lapply(datos, function(x) {
 if(is.character(x)) {
  as.numeric(x)
 } else {
  x
 }
})

datos$year<-as.numeric(datos$year)
datos$month<-as.numeric(datos$month)

datos<-do.call(data.frame,datos)
datos[,3]<-datos_antiguos[,3]
```

Codifico las variables WDF2y WDF5 para pasarlas a categóricas

```{r}
datos$WDF2Nuevo<-ifelse(datos$WDF2<45 |
datos$WDF2>315,"Norte",ifelse(datos$WDF2>=45 &
datos$WDF2<135,"Oeste",ifelse(datos$WDF2>=135 & datos$WDF2<225,"Sur","Este")))
datos$WDF5Nuevo<-ifelse(datos$WDF5<45 |
datos$WDF5>315,"Norte",ifelse(datos$WDF5>=45 &
datos$WDF5<135,"Oeste",ifelse(datos$WDF5>=135 & datos$WDF5<225,"Sur","Este")))
datos$WDF2Nuevo<-as.factor(datos$WDF2Nuevo)
datos$WDF5Nuevo<-as.factor(datos$WDF5Nuevo)
datos<-datos[,-which(names(datos) %in% c("WDF2", "WDF5"))]
colnames(datos)[colnames(datos) == "WDF2Nuevo"] <- "WDF2"
colnames(datos)[colnames(datos) == "WDF5Nuevo"] <- "WDF5"
```

Las variables DX90, DYNT, DYSD, DYSN, DYXP, DYXT TIENEN TODAS LA MISMA DESCRIPIPCION POR LO QUE LAS ELIMINO

```{r}
datos<-datos[,-which(names(datos) %in% c("DX90", "DYNT", "DYSD", "DYSN", "DYXP",
"DYXT"))]
```

Actualizo por último las categorías de la variable factor carrier ya que han quedado categorías sin observaciones al hacer la depuración de los datos

```{r}
datos$carrier<-as.character(datos$carrier)
datos$carrier<-as.factor(datos$carrier)
```

Ha pasado de ser un factor con 25 niveles (3 de ellos sin observaciones) a uno con 22 niveles (todos ellos con observaciones en el conjunto en mayor o menor medida)

Guardado de datos

Lo guardo en un xlsx nuevo
```{r}
write.xlsx(datos,file="C:/Users/molpe/OneDrive/Escritorio/TFG_VUELOS/CHICAGO/datosUnidosSinAttrib.xlsx")
```

Aplicación de modelos predictivos

Modelo de regresión logística binomial

No tenemos una variable 0,1

Puedo codificarlo como si propRetrasos>0.3 -> retraso=1

Importante: Igual que he decidido hacerla con mayor que 0.3 podría probar a ver si salen mejores modelos con otros valores

Tomo 0.3 ya que, que se retrase 1/3 vuelos ya me parece lo suficientemente riesgoso como para que se considere que tu vuelo tiene altas probabilidades de retrasarse.

También lo hago en parte porque hay muy pocas observaciones en las que haya 0.4 o incluso 0.5 de proporción de retrasos/vuelosTotales y los modelos tan descompensados entre 1 y 0 no predicen tan bien, por lo que es bueno que tengan una proporción algo más parecida

Primero creo un conjunto que contenga la variable que queremos predecir y las variables explicativas que la predecirán (variables tiempo atmosférico, mes, código del avión)

Elimino por tanto variables: year, arr_flights, arr_del15, carrier_ct, weather_ct, nas_ct, security_ct, late_aircraft_ct, arr_cancelled, arr_diverted, arr_delay, carrier_delay, weather_delay, nas_delay, security_delay, late_aircraft_delay

```{r}
names(datos[,c(1,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18)])
conjExplicativas<-datos[,c(-1,-4,-5,-6,-7,-8,-9,-10,-11,-12,-13,-14,-15,-16,-17,-18)]
propRetrasos<-datos[,5]/datos[,4]
datosPredecirProp<-cbind(propRetrasos,conjExplicativas)
```

```{r}
datosRegLogis<-datosPredecirProp
datosRegLogis <- datosRegLogis %>%
  mutate(retraso=ifelse(propRetrasos>=0.3, 1,0))
table(datosRegLogis$retraso)/nrow(datosRegLogis)
datosRegLogis<-datosRegLogis[,-1]
```

```
datosRegLogis$retraso<-as.factor(datosRegLogis$retraso)
```

Tenemos 4 veces más ceros que unos pero tampoco está exageradamente descompensado

Voy también a codificar mes como factor para que se interpreten como categorías en vez de numéricamente

```{r}
datosRegLogis$month<-as.factor(datosRegLogis$month)
```

Creo conjuntos de datos de entrenamiento y prueba

```{r}
set.seed(12345)
trainIndex <- createDataPartition(datosRegLogis$retraso, p=0.8, list=FALSE)
data_trainRegLogis <- datosRegLogis[trainIndex,]
data_testRegLogis <- datosRegLogis[-trainIndex,]
```

Creo el modelo de regresion logística binaria

```{r}
modeloLogBin<-glm(retraso~., data=data_trainRegLogis, family=binomial)
modeloLogBin$rank
```

Contamos con 73 parámetros en nuestro modelo

Saco los p-valores y la significatividad de los parametros de nuestro modelo

```{r}
summary(modeloLogBin)
```

No todos los parámetros son significativos

Realizamos un análisis de tipo II de las variables para ver cuales podemos retirar del modelo.

```{r}
Anova(modeloLogBin, type = "II")
```

Si tomamos un alfa de 0.05, nos quedaremos tan solo con las variables: Mes, código del avión, y las variables meteorológicas ADPT, AWND, CDSD, CLDD, DP01, DP1X, DSND, DT32, EMSD, HTDD y WDF5

De acuerdo a estos resultados volvemos a elaborar el modelo para solo estas variables:

```{r}
modeloLogBin2<-glm(retraso~ month + carrier + ADPT + AWND + CDSD + CLDD + DP01 + DP1X + DSND + DT32 + EMSD + HTDD + WDF5, data=data_trainRegLogis, family=binomial)
print(modeloLogBin2$rank)
```

Nuestro modelo tiene ahora 47 parámetros

Miramos nuevamente si todas las variables son significativas

```{r}
Anova(modeloLogBin2,type = "II")

```
CDSD, CLDD, EMSD y HTDD han dejado de ser significativas por lo que tambien las eliminamos del modelo
```{r}
modeloLogBin3<-glm(retraso~ month + carrier + ADPT + AWND + DP01 + DP1X + DSND + DT32 + WDF5, data=data_trainRegLogis, family=binomial)
print(modeloLogBin3$rank)
```

Tenemos 43 parámetros

```{r}
Anova(modeloLogBin3,type = "II")
```

La variable ADPT no es significativa

```{r}
modeloLogBin4<-glm(retraso~ month + carrier + AWND + DP01 + DP1X + DSND + DT32 + WDF5, data=data_trainRegLogis, family=binomial)
print(modeloLogBin4$rank)
```

Tenemos 42 parámetros

```{r}
Anova(modeloLogBin4,type = "II")
```
Ahora todas mis variables son significativas

Estos son los estimadores de los parámetros

```{r}
summary(modeloLogBin4)
```

Pasamos pues a analizar los odds-ratio

```{r}
```

```
exp(coef(modeloLogBin4))
```

SOLO SE PUEDEN INTERPRETAR LOS PARÁMETROS QUE SON SIGNIFICATIVOS

No voy a analizarlos todos ahora mismo pero si decir cuales hacen que un avión tienda a retrasarse y cuales hacen lo contrario

Odds ratio mayores que 1 hacen que por cada ud que aumentan,las probabilidades de que retraso sea 1 aumenten un oddratio%

Odds ratio menores que 1 hacen que por cada ud que aumentan,las probabilidades de que retraso sea 1 disminuyan un oddratio%

Notar que para los aviones la categoría de referencia es 9E por lo que la aeronave en cuestión que mencione se estará retrasando más en comparación con la 9E

En el caso de month, el mes de referencia es Enero

En el caso de WDF5 la direccion de referencia es Este

Factores que retrasan:

- Meses (al parecer los aviones se retrasan menos en enero que en el resto de meses, los meses en los que más probabilidad de retraso hay son los de verano (junio,julio,agosto))

- Carriers: CO NW OH_C RU WN

- AWND (Velocidad del viento media)

- DP01 (Días con 0.254 mm (0.1 in) de precipitación o más)

- DP1X (Días con 25.4 mm (1 in) de precipitación o más)

- DSND (Dias con 25 mm (1 in) de profundidad de nieve)

- DT32 (Dias con t. mínima menor o igual a 0)

Factores que no retrasan:

- Carriers: no hay ninguno significativo al nivel 0.05 por lo que no podemos interpretarlo.

- WDF5Oeste (es el único que podemos interpretar)

Voy a analizar ahora la validez de este modelo

Creo un conjunto de probabilidades que predice el modelo

```{r}
probstrain <-predict(modeloLogBin4, data_trainRegLogis, type="response")
```

Creo la matriz de confusion

```{r}
cmtrain<-
confusionMatrix(data=as.factor(ifelse(probstrain>=0.5,1,0)),reference=data_trainRegLogi
s$retraso, positive="1")
```

Predicciones y estadisticos

```{r}
cmtrain$table
cmtrain$overall[1:2]
cmtrain$byClass[1:2]
```

Vemos que la precisión a la hora de acertar observaciones con las que el modelo ha sido entrenado es del 83.87% lo cual está bastante bien, y que además, el modelo es bueno tanto a la hora de predecir los casos donde ha habido retraso como en los que no lo ha habido (siendo más preciso en los que no ha habido)

La kappa también es moderadamente buena según los estándares habituales (0.41 - 0.60 Moderado)

Veamos ahora cuan bueno es el modelo a la hora de predecir datos con los que no ha sido entrenado

```{r}
probstest <-predict(modeloLogBin4, data_testRegLogis, type="response")
```

Creo la matriz de confusion

```{r}
cmtest<-
confusionMatrix(data=as.factor(ifelse(probstest>=0.5,1,0)),reference=data_testRegLogis$
retraso, positive="1")
```

Predicciones y estadisticos

```{r}
cmtest$table
cmtest$overall[1:2]
cmtest$byClass[1:2]
```

Observamos que se acierta un 83.26% de las veces, la kappa sigue siendo moderadamente buena y la sensibilidad y especificidad no varían en gran manera
Como la proporción de 1 y 0 en nuestro conjunto está desbalanceada, probamos ahora a ajustar el punto de corte de 0.5 a 0.2058372.

```{r}
```

```
cmtest<-
confusionMatrix(data=as.factor(ifelse(probstest>=0.2058372,1,0)),reference=data_testRe
gLogis$retraso, positive="1")
cmtest$table
cmtest$overall[1:2]
cmtest$byClass[1:2]
```

Vemos ahora que la tasa de acierto es del 75%, menor que antes, pero que la sensibilidad y la especificidad se han quedado prácticamente iguales.

Calculamos ahora el AUC (area bajo la curva ROC) y pintamos la curva ROC tanto para los datos de prueba como para los de entrenamiento

Datos entrenamiento

```{r}
curvaROC<-roc(data_trainRegLogis$retraso, probstrain)
curvaROC$auc
plot(curvaROC)
```

Datos prueba

```{r}
curvaROCtest<-roc(data_testRegLogis$retraso, probstest)
curvaROCtest$auc
plot(curvaROCtest)
```

La AUC para los datos de entrenamiento es de 0.8445

La AUC para los datos de prueba es de 0.8154

```{r}
plot(curvaROC)
plot(curvaROCtest, add=T, col=2)
```

Están ambas bastante bien y sobretodo podemos observar que no se pierde gran cantidad de precisión y que nuestro modelo sirve para predecir bien observaciones con las que no ha sido entrenado

Validación cruzada

Como no pueden ser 1 o 0 los valores, convierto el 1 en "si" y el 0 en "no"

```{r}
dataVCR<-data_trainRegLogis
dataVCR$retraso<-factor(dataVCR$retraso, levels=rev(levels(dataVCR$retraso)),
labels=c("Si","No"))
table(dataVCR$retraso)/nrow(dataVCR)
```

```

Procedemos a la validacion cruzada

```{r}
set.seed(12345)
vcr<-caret::train(formula(modeloLogBin4),data = dataVCR,method = "glm",
family="binomial",trControl = trainControl(method="repeatedcv", number=5,
repeats=20,summaryFunction=multiClassSummary,classProbs=TRUE, savePredictions =
TRUE)
)
```

Obtenemos resultados estadisticos

```{r}
vcr$results[c(3,5,6,8,9,17,19,20,22,23)]
```

Estadisticos de los estadisticos para las repeticiones

```{r}
summary(vcr$resample[c(2,4,5,7,8)])
```

En lo que a valores medianos se refiere, podemos observar que tenemos AUC y precisión por encima del 82% en las repeticiones y que el Kappa es justo (0.21 - 0.40).

Podemos observar que predecimos mal los casos en los que hay retrasos pero muy bien los casos en los que no los hay, acertando en media un 94% de las veces. Lo que significa que en el 94% de los casos en los que no hubo un retraso, nuestro modelo habría predicho que no lo iba a haber.

Esto se debe, en parte, a la descompensación entre los casos de retraso y no retraso que sufre nuestro modelo

Matriz de confusion

```{r}
confusionMatrix(vcr)
```

Vemos que nuestro modelo mayoritariamente tiende a decantarse más por el No a la hora de hacer una predicción, indicando no en el 87.6% de las observaciones e indicando que si va a haber retraso solo un 12.4% de las veces

Como esto se puede deber a que la variable de estudio esta descompensada en 0 y 1 vamos a intentar balancearlo un poco

```{r}
thresholder(vcr,c(0.5,0.2058), statistics = c("Accuracy", "Kappa","Sensitivity", "Specificity"))
```
```

En la primera fila tenemos los estadísticos habiendo ajustado la proporcion de 1 y 0 y en la segunda los estadísticos sin ajustarla.

Vemos ahora que ajustando este parámetro la precisión cae en 8 puntos porcentuales pero que la sensibilidad (prob de haber predicho "si" en los casos en los que verdaderamente hubo un retraso) aumenta en gran medida del 37% al 74%. Estos cambios también afectan a la especificidad (haber predicho "no" en casos en los que no hubo retraso) que cae al 74%.

Creo que aunque la precisión haya disminuido, es importante que nuestro modelo sea igual de capaz de discernir entre casos en los que hay retraso y casos en los que no lo hay. El modelo anterior era bueno porque simplemente tendía a predecir no en casi todas las observaciones, y como por lo general los vuelos no se retrasaban, tenía un gran pocentaje de acierto.

Métodos automáticos de selección de variables (AIC y BIC)

Recurrimos a los métodos backward, forward y stepwise utilizando como criterio el AIC y el BIC.

```{r,include=FALSE}
null<-glm(retraso~1,data=data_trainRegLogis,family=binomial)
full<-glm(retraso~.,data=data_trainRegLogis,family=binomial)
modeloStepBIC<-step(null, scope=list(lower=null, upper=full), direction="both",
k=log(nrow(data_trainRegLogis)))
```

```{r}
modeloStepAIC<-step(null, scope=list(lower=null, upper=full), direction="both",trace=F)
modeloForwBIC<-step(null, scope=list(lower=null, upper=full),
direction="forward",k=log(nrow(data_trainRegLogis)),trace=F)
modeloForwAIC<-step(null, scope=list(lower=null, upper=full),
direction="forward",trace=F)
modeloBackBIC<-step(full, scope=list(lower=null, upper=full),
direction="backward",k=log(nrow(data_trainRegLogis)),trace=F)
modeloBackAIC<-step(full, scope=list(lower=null, upper=full),
direction="backward",trace=F)
```

```{r}
modelos<-
list(modeloStepBIC,modeloStepAIC,modeloForwBIC,modeloForwAIC,modeloBackBIC,
modeloBackAIC)
sapply(modelos,function(x) formula(x))
```

Numero de parametros de los modelos

```{r}
sapply(modelos,function(x) x$rank)
```

```

Obtenemos 5 modelos distintos con diferentes números de parámetros (El 1 y el 3 son iguales)

Una forma sencilla y gráfica de comparar estos modelos entre si es con un gráfico de caja y bigotes(comparamos el que ya teníamos con los nuevos creados)

```{r,warning=FALSE}
modelos<-
list(modeloLogBin4,modeloStepBIC,modeloStepAIC,modeloForwBIC,modeloForwAIC,modeloBackBIC,modeloBackAIC)
vcrTodosModelos<-list()
for (i in 1:length(modelos)){
set.seed(12345)
vcr<-caret::train(formula(modelos[[i]]), data = dataVCR,
method = "glm", family="binomial",
trControl = trainControl(method="repeatedcv", number=5, repeats=20,
summaryFunction=multiClassSummary, classProbs=TRUE,
savePredictions = TRUE)
)
vcrTodosModelos[[i]]<-vcr
}
bwplot(resamples(vcrTodosModelos), metric=c("AUC", "Kappa", "Accuracy"),
scales = list(x = list(relation = "free")))
```

Los nombres model(num) están asignados según aparecen el la lista modelos de la primera lína del chunk.

Como para las tres métricas buscamos el modelos que las maximicen, nos vamos a fijar en que modelos tienen su mediana (punto negro) más a la derecha.

Aunque los márgenes de mejora son muy pequeños (si vemos la escala del eje x nos damos cuenta), los tres mejores modelos son el 7 el 3 y el 5 (Aunque no son los más precisos).

Como son muy similares tanto en precisión como en área bajo la curva como en Kappa, nos decantaremos por el modelo más simple (con menos parámetros) ya que va a ser el más sencillo el que sea más facil de interpretar y del que podamos sacar conclusiones más claras que dependan de menos factores.

Si nos vamos al chunk en el que mirábamos el número de parámetros de cada modelo:

```{r}
sapply(modelos,function(x) x$rank)
```

De entre el 3 el 5 y el 7 el que menos parámetros tiene es el 3, con 53 parámetros (2 y 5 menos que los otros dos). Este modelo (modeloStepAIC) es el obtenido mediante el método stepwise y el criterio de parada AIC.
```

Veamos con que variables cuenta este modelo

```{r}
formula(modeloStepAIC)
```

También nos puede interesar ver los estadísticos más en detalle

```{r}
summary(resamples(vcrTodosModelos[c(1,3)]), metric=c("AUC", "Kappa", "Accuracy"))
```

Model1 es el primero que hicimos (modeloLogBin4) y Model2 el modeloStepAIC

Vemos que están muy parejos en lo que se refiere a estadísticos por pocos puntos porcentuales, por lo que será preferible el más sencillo (menos parámetros)

```{r}
modeloLogBin4$rank
formula(modeloLogBin4)
modeloStepAIC$rank
formula(modeloStepAIC)
```

EL modelo incial cuenta con 11 parámetros menos y lo que es más importante, cuenta con 9 variables menos, por lo que realmente no es preferible el obtenido por selección automática de variables y nos quedamos con el hecho a mano.

Árboles de clasificación

En esta parte vamos a tratar los árboles de clasificación, modelos usados para predecir variables de tipo binario (0, 1).
Debido a la naturaleza de las variables objetivo de estos modelos, utilizaremos el conjunto datosRegLogis que contenía todas nuestras variables explicativas y una variable "retraso" que toma el valor 0 cuando la proporcion de vuelos retrasados en esa observación ha sido menor que 0.3 y 1 para los casos en los que esta proporción sea mayor o igual a 0.3.

Realizamos primero la partición del conjunto de datos en conjunto de train y conjunto de test.
```{r}
set.seed(12345)
trainIndex <- createDataPartition(datosRegLogis$retraso, p=0.8, list=FALSE)
data_clasif_train <- datosRegLogis[trainIndex,]
data_clasif_test <- datosRegLogis[-trainIndex,]
```
```{r}
table(datosRegLogis$retraso)/nrow(datosRegLogis)
```
Vemos que hay casi un 80% de las observaciones clasificadas como no retraso y un 20% de las observaciones clasificadas como retraso

En árboles de clasificación hay varios criterios que se pueden seguir.

- Criterio Índice de Gini:
El índice de Gini no indica cómo de homogéneas son las observaciones de un grupo con respecto a una variable categórica (en nuestro caso la variable retraso)

Tomamos un tamaño de hoja del 2%
```{r}
set.seed(12345)
modeloGini2<-rpart(retraso~., data=data_clasif_train, method = "class",
minbucket=ceiling(0.02*nrow(data_clasif_train)),cp=0,
parms=list(split="gini"),maxsurrogate = 0)
```

Sacamos el número de hojas del árbol

```{r}
sum(modeloGini2$frame$var == "<leaf>")
```

Sacamos la profundidad del árbol

```{r}
max(rpart:::tree.depth(as.numeric(rownames(modeloGini2$frame))))
```

Pintamos el arbol

```{r}
rpart.plot(modeloGini2,extra=105,nn=TRUE,tweak=1.2)
```

Este diagrama es una representación de nuestro árbol. El color verde indica hojas en las que el 1 (vuelo se retrasa) tiene una mayor proporción de aparición que el 0 (el vuelo no se retrasa).

Podemos ver que la primera regla de decisión que toma para ver si un vuelo se retrasa o no, es si el carrier pertenece al conjunto de "9E,AA,AS,DL,EV,MQ,NK, OH_P, OO, QX,UA,VX,YV,YX". Vemos que el 72% de las observaciones pertenecen a esos aviones y que el 28% no lo hacen. De este 28%, el 40% se retrasan y del 60% solo se retrasa el 15%.

La siguiente regla que vemos que se aplica si se va por el camino de la derecha es si la observación pertenece a los meses 9, 10 y 11. Las observaciones que no pertenecen a este grupo, se retrasan el 49% de las veces.

En las últimas hojas (las de más abajo) podemos ver como se ha clasificado en función de los valores de las variables más importantes.

- La hoja con una mayor proporción de vuelos retrasados (78%, la más verde) es la de la derecha del todo. Son vuelos no pertenecientes al conjunto de carriers "9E,AA,AS,DL,EV,MQ,NK, OH_P, OO, QX,UA,VX,YV,YX", que no han sido en "Septiembre, Octubre o Noviembre", tampoco han sido en "enero, febrero, marzo, abril o agosto" y

cuyas compañias tampoco pertenecían al conjunto de carriers "F9, NW, US y XE". Dicho de otra manera, vuelos ocurridos en los meses de "junio, julio o diciembre" cuyas compañias fueron "B6, CO, OH_C o WN". Representa un 4% de todas las observaciones del conjunto.

- La hoja con una menor proporción de vuelos retrasados (10%, la más azul) es la de la izquierda del todo. Son vuelos en los que la compañía pertenecía al conjunto "9E,AA,AS,DL,EV,MQ,OO,QX,UA,VX,YV,YX" y el mes en el que ocurrió el vuelo no se superaron los 13 días con temperatura máxima menor o igual a 0 grados Celsius. Representa el 63% del total de observaciones del conjunto.

El siguiente código nos permite ver las reglas de decisión de una manera escrita

```{r}
asRules(modeloGini2)
```

- Criterio de Entropía:
Indica como homogéneo es un grupo de observaciones teniendo en cuenta que en un grupo homogéneo la entropía es cero

```{r}
set.seed(12345)
modeloEnt2<-rpart(retraso~., data=data_clasif_train, method = "class",
minbucket=ceiling(0.02*nrow(data_clasif_train)),cp=0,
parms=list(split="information"),maxsurrogate = 0)
```

Número de hojas

```{r}
sum(modeloEnt2$frame$var == "<leaf>")
```

Profundidad del árbol

```{r}
max(rpart:::tree.depth(as.numeric(rownames(modeloEnt2$frame))))
```

```{r}
rpart.plot(modeloEnt2,extra=105,nn=TRUE,tweak=1.2)

```

- La hoja con mayor porcentaje de retrasos tiene un 77% y representa un 4% del total de observaciones de nuestro conjunto. Cumple las condiciones de: no estar incluido en el conjunto de carriers siguiente "9E,AA,AS,DL,EV,MQ,OH_P,OO,QX,UA,VX,YX", no haber volado en los meses de septiembre, octubre o noviembre, no haber volado tampoco en las compañias "F9, NK, US, XE, YV" y tampoco haber volado en los meses "enero, febrero, marzo, abril, mayo y agosto". Esto es lo mismo que decir que son vuelos que han volado

en los meses de "junio, julio o diciembre" en alguna de las compañías pertenecientes al conjunto de "B6, CO, NW, OH_C, RU y WN".

- La hoja con menor porcentaje de retrasos tiene un 11% y representa un 64% del total de observaciones del conjunto. Son los vuelos cuya compañia pertenece a la siguiente lista "9E,AA,AS,DL,EV,MQ,OH_P,OO,QX,UA,VX,YX".

```{r}
asRules(modeloEnt2)
```

Estos son los dos criterios principales a la hora de elaborar los árboles de clasificación.

Podemos ahora jugar con el parámetro tamaño de hoja, que indica hasta cuando puede clasificar nuevas hojas el diagrama (cuando llega a hojas de tamaño 2% en el caso de los árboles que acabamos de crear).

Podemos crear árboles con tamaños de hoja 1% y 5% usando ambos criterior para compararlos posteriormente con los que acabamos de comentar.

```{r}
set.seed(12345)
modeloGini1<-rpart(retraso~., data=data_clasif_train, method = "class",
minbucket=ceiling(0.01*nrow(data_clasif_train)),cp=0,
parms=list(split="gini"),maxsurrogate = 0)
set.seed(12345)
modeloGini5<-rpart(retraso~., data=data_clasif_train, method = "class",
minbucket=ceiling(0.05*nrow(data_clasif_train)),cp=0,
parms=list(split="gini"),maxsurrogate = 0)
set.seed(12345)
modeloEnt1<-rpart(retraso~., data=data_clasif_train, method = "class",
minbucket=ceiling(0.01*nrow(data_clasif_train)),cp=0,
parms=list(split="information"),maxsurrogate = 0)
set.seed(12345)
modeloEnt5<-rpart(retraso~., data=data_clasif_train, method = "class",
minbucket=ceiling(0.05*nrow(data_clasif_train)),cp=0,
parms=list(split="information"),maxsurrogate = 0)
```

Meto los modelos generados en una lista

```{r}
modelos<-
list(modeloGini1,modeloGini2,modeloGini5,modeloEnt1,modeloEnt2,modeloEnt5)
```

Veamos cuales tienen mejor AUC. Acompañamos a estos resultados con el número de hojas de cada árbol para tener en cuenta la complejidad de nuestros modelos a la hora de elegir cual es el mejor.

Con los valores de train

```{r}
sapply(modelos,
function(x) roc(data_clasif_train$retraso,
predict(x,data_clasif_train,type="prob")[,2], direction="<")$auc)

sapply(modelos,function(x) sum(x$frame$var == "<leaf>"))

```

Con los valores de test

```{r}
sapply(modelos,
function(x) roc(data_clasif_test$retraso,
predict(x,data_clasif_test,type="prob")[,2], direction="<")$auc)

sapply(modelos,function(x) sum(x$frame$var == "<leaf>"))
```

Vemos que los que mejor clasifican para ambos criterios y conjuntos (train y test) son los que más hojas tienen (15 y 26).

Eligiendo los modelos más simples sacrificamos un AUC para tener un modelo con menos hojas.

Parece preferible por tanto, en mi opinion, el último modelo de la lista de modelos (modeloEnt5), que consta de 4 hojas y que tiene un AUC con datos de entrenamiento del 71.45% y un AUC con datos de test del 73.76%.

Podemos también querer evaluar la tasa de acierto, kappa y sensibilidad del modelo ganador

Estadísticos para el conjunto de entrenamiento

```{r}
probs <-predict(modeloEnt5,data_clasif_train,type="prob")
cm<-confusionMatrix(data=as.factor(ifelse(probs[,2]>=0.5,1,0)),
reference=data_clasif_train$retraso,positive="1")
cm$table
cm$overall[1:2]
cm$byClass[1:2]
```

Tenemos un porcentaje de buena clasificación del 80.85% y un kappa justo de 0.3 (0.21 - 0.40)

Tenemos una especificidad muy alta (clasificamos correctamente el 93.29% de los casos en los que no existe retraso), pero una sensibilidad muy baja (clasificamos correctamente solo el 32.87% de los casos en los que existe retraso).

Podemos probar a ajustar las proporciones de 0 y 1 para que se balanceen un poco las sensibilidad y la especificidad

```{r}
probs <-predict(modeloEnt5,data_clasif_train,type="prob")
cm<-confusionMatrix(data=as.factor(ifelse(probs[,2]>=0.2058372,1,0)),
reference=data_clasif_train$retraso,positive="1")
cm$table
cm$overall[1:2]
cm$byClass[1:2]
```

Como podemos ver la especificidad se ha reducido al 80% pero la sensibilidad (capacidad para identificar correctamente los casos en los que existe retraso) ha aumentado notablemente hasta casi el 60% de lo casos.

La precisión se ha visto reducida en 4 puntos porcentuales.

Hacemos ahora lo mismo pero con los datos de test

```{r}
probs <-predict(modeloEnt5,data_clasif_test,type="prob")
cm<-confusionMatrix(data=as.factor(ifelse(probs[,2]>=0.5,1,0)),
reference=data_clasif_test$retraso,positive="1")
cm$table
cm$overall[1:2]
cm$byClass[1:2]
```

```{r}
probs <-predict(modeloEnt5,data_clasif_test,type="prob")
cm<-confusionMatrix(data=as.factor(ifelse(probs[,2]>=0.2058372,1,0)),
reference=data_clasif_test$retraso,positive="1")
cm$table
cm$overall[1:2]
cm$byClass[1:2]
```

Como habiamos visto brevemente antes con los AUC de la lista de todos los modelos, este arbol tiene de particular que clasifica mejor para los datos de test siendo en este caso superior en 2 puntos porcentuales en sensibilidad, especificidad, precisión e incluso kappa.

Una vez hemos encontrado el arbol óptimo, nos puede interesar su poda. Este proceso consiste en ver si eliminando alguna hoja, el árbol predice o no mejor debido a la eliminación de un sobreajuste del que podemos no habernos dado cuenta.

Como nuestro arbol tiene tan pocas hojas, no parece necesario realizar ninguna poda con él. Sin embargo podríamos coger el árbol que mejor AUC tenía generado anteriormente (modeloEnt1) que contaba con 26 hojas.

Primero generamos la secuencia de valores críticos que da pie al aumento o disminución en tamaño del arbol y la graficamos para ver que número de hojas es el adecuado para nuestro arbol

```{r}
modeloEnt1$cptable
```
Representacion gráfica
```{r}
plotcp(modeloEnt1)
```

```{r}
modelo1poda1 <- prune(modeloEnt1, cp = modeloEnt1$cptable[6,1])
rpart.plot(modelo1poda1 ,extra=105,nn=TRUE,tweak=1.2)
```

```{r}
modelo1poda2 <- prune(modeloEnt1, cp = modeloEnt1$cptable[5,1])
rpart.plot(modelo1poda2,extra=105,nn=TRUE,tweak=1.2)
```

comparar arbol podado con árbol sin podar en Rcuadrado en test y entrenamiento

```{r}
sum(modeloEnt1$frame$var == "<leaf>")
roc(data_clasif_test$retraso,
predict(modeloEnt1,data_clasif_test,type="prob")[,2], direction="<")$auc
sum(modelo1poda1$frame$var == "<leaf>")
roc(data_clasif_test$retraso,
predict(modelo1poda1,data_clasif_test,type="prob")[,2], direction="<")$auc
sum(modelo1poda2$frame$var == "<leaf>")
roc(data_clasif_test$retraso,
predict(modelo1poda2,data_clasif_test,type="prob")[,2], direction="<")$auc

```

Lo comparamos también con el que escogimos anteriormente

```{r}
rpart.plot(modeloEnt5,extra=105,nn=TRUE,tweak=1.2)
```

```{r}
sum(modeloEnt5$frame$var == "<leaf>")
roc(data_clasif_test$retraso,
predict(modeloEnt5,data_clasif_test,type="prob")[,2], direction="<")$auc
```

Es preferible la segunda poda, ya que tiene un AUC de casi 0.8 y aunque tiene 14 hojas, es preferible a un AUC del 73%

Árboles de regresión

En esta parte vamos a tratar los árboles de regresión, modelos usados para predecir variables de tipo continuo.
Debido a la naturaleza de las variables objetivo de estos modelos, utilizaremos el conjunto datosPredecirProp que contenía todas nuestras variables explicativas y una variable "propRetrasos" que representa la proporcion de vuelos retrasados en esa observación sobre el total de vuelos para la misma.

```{r}
median(datosPredecirProp$propRetrasos)
```

La proporción de vuelos retrasados es de aproximadamente el 21%

Este es un histograma que representa como se distribuye la variable objetivo

```{r}
hist(datosPredecirProp$propRetrasos)
```

La variable se distribuye principalmente en torno a 0.1 y 0.3

Realizamos primero la partición del conjunto de datos en conjunto de train y conjunto de test.

```{r}
set.seed(12345)
trainIndex <- createDataPartition(datosPredecirProp$propRetrasos, p=0.8, list=FALSE)
data_rg_train <- datosPredecirProp[trainIndex,]
data_rg_test <- datosPredecirProp[-trainIndex,]
```

Creamos el arbol

```{r}
set.seed(12345)
modeloReg5<-rpart(propRetrasos~., data=data_rg_train, method = "anova",
minbucket=ceiling(0.05*nrow(data_rg_train)),
cp=0, maxsurrogate = 0)
```

Numero de hojas

```{r}
sum(modeloReg5$frame$var == "<leaf>") #cuenta el número de hojas
```

Profundidad del arbol

```{r}
max(rpart:::tree.depth(as.numeric(rownames(modeloReg5$frame)))) #cuenta la
profundidad
```

Pintar arbol

```{r}
rpart.plot(modeloReg5, nn=TRUE, tweak=1.1, digits=3)
```

- La hoja que contiene las observaciones con una mayor proporción de retrasos es la de la derecha del todo (son un 13.7% de las observaciones y tienen una proporción media de retraso de 0.31). Estas observaciones tienen en comun no pertenecer al conjunto de carriers "9E,AA,AS,DL,EV,MQ,OH_P,OO,QX,UA,VX,YV,YX", y que el número de días en el mes con más de 0.254 milímetros de lluvia haya sido mayor que 12.

- La hoja que contiene las observaciones con una menor propocion media de retrasos es la de la izquierda del todo (son un 7.0% del total de observaciones y se retrasan en media un 14% de las veces). Estas observaciones tienen en comun pertenecer al conjunto de carriers "9E,AA,AS,DL,EV,MQ,OH_P,OO,QX,UA,VX,YV,YX", que el número de días en el mes con más de 0.254 milímetros de lluvia haya sido menor que 12 y que el numero total de grados día de refrigeración de lo que se lleva de temporada sea mayor o igual a 620.

Estas son las normas que se han seguido para cada hoja de forma escrita

```{r}
asRules(modeloReg5)
```

Podemos probar, como hicimos con los arboles de clasificación, a variar el tamaño de hoja para ver si mejora o empeora la capacidad predictiva de nuestro modelo

```{r}
set.seed(12345)
modeloReg1<-rpart(propRetrasos~., data=data_rg_train, method = "anova",
minbucket=ceiling(0.01*nrow(data_rg_train)),
cp=0, maxsurrogate = 0)

modeloReg2<-rpart(propRetrasos~., data=data_rg_train, method = "anova",
minbucket=ceiling(0.02*nrow(data_rg_train)),
cp=0, maxsurrogate = 0)

modeloReg10<-rpart(propRetrasos~., data=data_rg_train, method = "anova",
minbucket=ceiling(0.1*nrow(data_rg_train)),
cp=0, maxsurrogate = 0)
```

R cuadrado de distintos modelos

```{r}
modelos<-list(modeloReg1, modeloReg2, modeloReg5, modeloReg10)
sapply(modelos, function(x) R2(predict(x,data_rg_train), data_rg_train$propRetrasos))
sapply(modelos, function(x) R2(predict(x,data_rg_test), data_rg_test$propRetrasos))
sapply(modelos, function(x) sum(x$frame$var == "<leaf>"))
```

Parece que en caso de la regresión, no parece que seamos capaces de hacer buenas predicciones, al menos usando el R cuadrado como métrica.

Random forest (modelos bagging) para clasificación

Entendido como funciona un árbol pasamos al bagging, técnica la cual se basa en la generación de múltiples árboles para reunir la información de todos estos en una única predicción (ensamblado de árboles).

Se recurre a las muestras boostrap, submuestras de la población con reemplazamiento.

La idea principal es reducir la variabilidad en la que incurre al tomar un solo árbol, optando por generar múltiples árboles con profundidades altas y un tamaños de hoja pequeños.

Este método no permite interpretar la forma en la que se generan estas predicciones pero si se puede sacar un ranking de variables más importantes para entender cuales son las que más peso tienen a la hora de predecir

El modelo bagging que vamos a llevar a cabo se conoce como bosque aleatorio (RandomForest)

Creamos un RF de 500 árboles con un tamaño de hoja del 5%

```{r}
set.seed(12345)
RFInicial_Retraso<-randomForest(retraso~., data=data_clasif_train, keep.inbag = TRUE,
nodesize=ceiling(0.05*nrow(data_clasif_train)))
print(RFInicial_Retraso)
```

Tenemos un modelo con una tasa de error del 17.61%

Pasemos a estudiar cual es el número óptimo de árboles de nuestro bosque

```{r}
auxtask <- makeClassifTask(data = data_clasif_train, target = "retraso")
results <- OOBCurve(RFInicial_Retraso, measures = list(auc,kappa),
task = auxtask, data = data_clasif_train)
plot(results$auc, ylab = "oob-auc", xlab = "ntrees")
```

```{r}
results$auc[c(1,100,200,300,400,500)]

```

El AUC parece estabilizarse a los 100 - 200 árboles por lo que optaríamos 200

```{r}
```

```
plot(results$kappa, ylab = "oob-kappa", xlab = "ntrees")
```

```{r}
results$kappa[c(1,100,200,300,400,500)]
```

La kappa tiene un pequeño valle en torno a 200 árboles y vuelve a mejorar a los 300, por lo que nos quedaremos finalmente con 300 (que tiene un mejor AUC además)

Veamos ahora si reduciendo el tamaño de hoja observamos algún tipo de mejora

```{r}
set.seed(12345)
RF_1_retraso<-randomForest(retraso~., data=data_clasif_train, keep.inbag = TRUE,
nodesize=ceiling(0.01*nrow(data_clasif_train)))
print(RF_1_retraso)
```

Hay una mejora de 2.5 puntos porcentuales en el error estimado. Si seguimos reduciendo el tamaño de la hoja no mejoramos mucho más este error por lo que nos quedamos con un tamaño del 1%.

Veamos que número de árboles es el mejor para este caso

```{r}
auxtask <- makeClassifTask(data = data_clasif_train, target = "retraso")
results <- OOBCurve(RF_1_retraso, measures = list(auc,kappa),
task = auxtask, data = data_clasif_train)
plot(results$auc, ylab = "oob-auc", xlab = "ntrees")
```
```{r}
results$auc[c(1,100,200,300,400,500)]

```

A partir de 100 es muy similar el AUC

```{r}
plot(results$kappa, ylab = "oob-kappa", xlab = "ntrees")
```

```{r}
results$kappa[c(1,100,200,300,400,500)]
```

La kappa parece también estabilizarse a partir de 100 árboles por lo que este parece el número óptimo de árboles.

Podemos ahora probar a ajustar el parámetro mtry, que hace referencia al tamaño de los subconjuntos aleatorios de variables explicativas entre las que los árboles eligen a la hora de dividir sus hojas

```{r}
mtryValues<-floor(seq(2,ncol(data_clasif_train)-1,length.out=5))
modelos_clasif<-list()
AUCs<-c()
Kappas<-c()
for (i in 1:length(mtryValues)){
set.seed(12345)
modelos_clasif[[i]]<-randomForest(retraso~., data=data_clasif_train, keep.inbag = TRUE,
nodesize=ceiling(0.01*nrow(data_clasif_train)),
ntree=100,
mtry=mtryValues[i])
results_loop<-OOBCurve(modelos_clasif[[i]], measures = list(auc, kappa),
task = auxtask, data = data_clasif_train)
AUCs <- c(AUCs, results_loop$auc[modelos_clasif[[i]]$ntree])
Kappas <- c(Kappas, results_loop$kappa[modelos_clasif[[i]]$ntree])
}
plot(mtryValues, AUCs, type="b")
```

El mejor AUC se alcanza para un valor de mtry de 10 variables

```{r}
plot(mtryValues, Kappas, type="b")
```

El mejor valor de mtry se alcanza entre 25 y 30 variables pero vemos que para 10 variables el valor de Kappa es muy similar, por lo que nos quedamos con un mtry de 10 variables

Estas son las características de nuestro modelo ganador

```{r}
print(modelos_clasif[[2]])
```

Podemos mostrar también las variables que más importan a la hora de clasificar observaciones para este modelo
```{r}
barplot(t(modelos_clasif[[2]]$importance)/sum(modelos_clasif[[2]]$importance), las=2)
```

La más importante es carrier, seguida de month, DP01 y el resto de variables.

Damos por último unas estimaciones de la calidad del modelo generado

```{r}
probs <-predict(modelos_clasif[[2]],data_clasif_test,type="prob")
cm<-confusionMatrix(data=as.factor(ifelse(probs[,2]>=0.5,1,0)),
reference=data_clasif_test$retraso,positive="1")
cm$table
cm$overall[1:2]
cm$byClass[1:2]
```

```

Si ajustamos el porcentaje de unos y ceros de nuestro conjunto.

```{r}
cm2<-confusionMatrix(data=as.factor(ifelse(probs[,2]>=0.2058372,1,0)),
reference=data_clasif_test$retraso,positive="1")
cm2$table
cm2$overall[1:2]
cm2$byClass[1:2]
```

Perdemos precisión pero tenemos una capacidad de predecir casos en los que hay y no hay retrasos más balanceada

XGBoost regresión

XGBoost eliminando categóricas

```{r,results='hide'}
sumaError<-0
for (i in 1:50){
 set.seed(i)
 trainIndex <- createDataPartition(datosPredecirProp$propRetrasos, p=0.8, list=FALSE)
 data_rg_train <- datosPredecirProp[trainIndex,]
 data_rg_test <- datosPredecirProp[-trainIndex,]

 datosEntrenamientoXGReg<-data_rg_train[,-c(2,3,37,38)]

 datosEntrenamientoXGReg <- datosEntrenamientoXGReg %>%
  dplyr::select(-propRetrasos) %>%
  as.matrix() %>%
  xgb.DMatrix(data = ., label = data_rg_train$propRetrasos)

 datosTestXGReg<-data_rg_test[,-c(2,3,37,38)]

 modeloXGBoostReg<-xgboost(data = datosEntrenamientoXGReg,
                   objective = "reg:squarederror",
                   nrounds = 60, max.depth = 3, eta = 0.5, nthread = 2,verbose=0)

 datospredXGReg <- data_rg_test[,-c(2,3,37,38)]
 datospredXGReg <- datospredXGReg %>%
  dplyr::select(-propRetrasos) %>%
  as.matrix() %>%
  xgb.DMatrix(data = ., label = data_rg_test$propRetrasos)

 prediccionReg <- predict(modeloXGBoostReg, datospredXGReg)
 sumaError<-sumaError+mean((prediccionReg-datosTestXGReg$propRetrasos)^2)
}
```

Error cuadrático medio en las 50 repeticiones
```

```{r}
sumaError/50
```

XGBoost sustituyendo cada categoría de las variables factor por el valor medio de la variable objetivo para esa categoría

```{r}
conjuntoXGMedios<-datosPredecirProp
conjuntoXGMedios$month<-as.factor(conjuntoXGMedios$month)

#Variable WDF5
valoresMedios <- conjuntoXGMedios %>%
 group_by(WDF5) %>%
 summarise(mediasPropRetrasos = mean(propRetrasos, na.rm = TRUE))

conjuntoXGMedios <- conjuntoXGMedios %>%
 left_join(valoresMedios, by = "WDF5")

conjuntoXGMedios <- conjuntoXGMedios %>%
 mutate(WDF5 = mediasPropRetrasos) %>%
 dplyr::select(-mediasPropRetrasos)


#Variable WDF2
valoresMedios <- conjuntoXGMedios %>%
 group_by(WDF2) %>%
 summarise(mediasPropRetrasos = mean(propRetrasos, na.rm = TRUE))

conjuntoXGMedios <- conjuntoXGMedios %>%
 left_join(valoresMedios, by = "WDF2")

conjuntoXGMedios <- conjuntoXGMedios %>%
 mutate(WDF2 = mediasPropRetrasos) %>%
 dplyr::select(-mediasPropRetrasos)


#Variable month
valoresMedios <- conjuntoXGMedios %>%
 group_by(month) %>%
 summarise(mediasPropRetrasos = mean(propRetrasos, na.rm = TRUE))

conjuntoXGMedios <- conjuntoXGMedios %>%
 left_join(valoresMedios, by = "month")

conjuntoXGMedios <- conjuntoXGMedios %>%
 mutate(month = mediasPropRetrasos) %>%
 dplyr::select(-mediasPropRetrasos)


#Variable carrier
```

```
valoresMedios <- conjuntoXGMedios %>%
 group_by(carrier) %>%
 summarise(mediasPropRetrasos = mean(propRetrasos, na.rm = TRUE))

conjuntoXGMedios <- conjuntoXGMedios %>%
 left_join(valoresMedios, by = "carrier")

conjuntoXGMedios <- conjuntoXGMedios %>%
 mutate(carrier = mediasPropRetrasos) %>%
 dplyr::select(-mediasPropRetrasos)
```

```{r,results='hide'}
sumaErrorCatMed<-0
for (i in 1:50){
 set.seed(i)
 trainIndex <- createDataPartition(conjuntoXGMedios$propRetrasos, p=0.8, list=FALSE)
 data_rg_train <- conjuntoXGMedios[trainIndex,]
 data_rg_test <- conjuntoXGMedios[-trainIndex,]

 datosEntrenamientoXGRegCatMed <- data_rg_train %>%
  dplyr::select(-propRetrasos) %>%
  as.matrix() %>%
  xgb.DMatrix(data = ., label = data_rg_train$propRetrasos)

 datosTestXGRegCatMed<-data_rg_test

 modeloXGBoostRegCatMed<-xgboost(data = datosEntrenamientoXGRegCatMed,
                 objective = "reg:squarederror",
                 nrounds = 60, max.depth = 2, eta = 0.5, nthread = 2,verbose=0)

 datospredXGRegCatMed <- data_rg_test %>%
  dplyr::select(-propRetrasos) %>%
  as.matrix() %>%
  xgb.DMatrix(data = ., label = data_rg_test$propRetrasos)

 prediccionRegCatMed <- predict(modeloXGBoostRegCatMed, datospredXGRegCatMed)
 sumaErrorCatMed<-sumaErrorCatMed+mean((prediccionRegCatMed-
datosTestXGRegCatMed$propRetrasos)^2)
}
```

Podemos calcular el error cuadrático medio

```{r}
sumaErrorCatMed/50
```

XGBoost haciendo las variables categóricas dummy (una variable para cada categoría que tome 1 cuando la variable original tomaba dicha categoría y 0 cuando no lo hacía)

```{r}
```

```
conjuntoXGDummies<-datosPredecirProp
conjuntoXGDummies$month<-as.factor(conjuntoXGDummies$month)

conjuntoXGDummies <- dummy_cols(conjuntoXGDummies, select_columns = "month",
remove_first_dummy = TRUE, remove_selected_columns = TRUE)

conjuntoXGDummies <- dummy_cols(conjuntoXGDummies, select_columns = "carrier",
remove_first_dummy = TRUE, remove_selected_columns = TRUE)

conjuntoXGDummies <- dummy_cols(conjuntoXGDummies, select_columns = "WDF2",
remove_first_dummy = TRUE, remove_selected_columns = TRUE)

conjuntoXGDummies <- dummy_cols(conjuntoXGDummies, select_columns = "WDF5",
remove_first_dummy = TRUE, remove_selected_columns = TRUE)
```

```{r}
sumaErrorDummy<-0
for (i in 1:50){
 set.seed(i)
 trainIndex <- createDataPartition(conjuntoXGDummies$propRetrasos, p=0.8, list=FALSE)
 data_rg_train <- conjuntoXGDummies[trainIndex,]
 data_rg_test <- conjuntoXGDummies[-trainIndex,]

 datosEntrenamientoXGRegDummy <- data_rg_train %>%
  dplyr::select(-propRetrasos) %>%
  as.matrix() %>%
  xgb.DMatrix(data = ., label = data_rg_train$propRetrasos)

 datosTestXGRegDummy<-data_rg_test

 modeloXGBoostRegDummy<-xgboost(data = datosEntrenamientoXGRegDummy,
                 objective = "reg:squarederror",
                 nrounds = 60, max.depth = 3, eta = 0.5, nthread = 2,verbose = 0)

 datospredXGRegDummy <- data_rg_test %>%
  dplyr::select(-propRetrasos) %>%
  as.matrix() %>%
  xgb.DMatrix(data = ., label = data_rg_test$propRetrasos)

 prediccionRegDummy <- predict(modeloXGBoostRegDummy, datospredXGRegDummy)
 sumaErrorDummy<-sumaErrorDummy+mean((prediccionRegDummy-
datosTestXGRegDummy$propRetrasos)^2)
}
```

Podemos calcular el error cuadrático medio

```{r}
sumaErrorDummy/50
```

Todos los errores

```{r}
print("XGBoost sin categóricas")
sumaError/50
print("XGBoost categóricas sustituidas")
sumaErrorCatMed/50
print("XGBoost con dummies")
sumaErrorDummy/50
```

Regresion lineal multiple sin categóricas

```{r}
sumaErrorMLRSinCat<-0
for (i in 1:50){
 set.seed(i)
 trainIndex <- createDataPartition(datosPredecirProp$propRetrasos, p=0.8, list=FALSE)
 data_rg_trainMLRSinCat <- datosPredecirProp[trainIndex,-c(2,3,38,37)]
 data_rg_testMLRSinCat <- datosPredecirProp[-trainIndex,-c(2,3,38,37)]

 mlrSinCat<-lm(propRetrasos~ .,data=data_rg_trainMLRSinCat)

 prediccionMLRSinCat<- predict(mlrSinCat, data_rg_testMLRSinCat[,-1])
 sumaErrorMLRSinCat<-sumaErrorMLRSinCat+mean((prediccionMLRSinCat-
data_rg_testMLRSinCat$propRetrasos)^2)
}
sumaErrorMLRSinCat/50
```

Regresion lineal multiple sustituidas categorías

```{r}
sumaErrorMLRCatMed<-0
for (i in 1:50){
 set.seed(i)
 trainIndex <- createDataPartition(conjuntoXGMedios$propRetrasos, p=0.8, list=FALSE)
 data_rg_trainMLRCatMed <- conjuntoXGMedios[trainIndex,]
 data_rg_testMLRCatMed <- conjuntoXGMedios[-trainIndex,]

 mlrCatMed<-lm(propRetrasos~ .,data=data_rg_trainMLRCatMed)

 prediccionMLRCatMed <- predict(mlrCatMed, data_rg_testMLRCatMed[,-1])
 sumaErrorMLRCatMed<-sumaErrorMLRCatMed+mean((prediccionMLRCatMed-
data_rg_testMLRCatMed$propRetrasos)^2)
}
sumaErrorMLRCatMed/50
```

Regresion lineal multiple DUMMIES

```{r}
sumaErrorMLRDummy<-0
```

```
for (i in 1:50){
  set.seed(i)
  trainIndex <- createDataPartition(conjuntoXGDummies$propRetrasos, p=0.8, list=FALSE)
  data_rg_trainMLRDummy <- conjuntoXGDummies[trainIndex,]
  data_rg_testMLRDummy <- conjuntoXGDummies[-trainIndex,]

  mlrDummy<-lm(propRetrasos~ .,data=data_rg_trainMLRDummy)

  prediccionMLRDummy <- predict(mlrDummy, data_rg_testMLRDummy[,-1])
  sumaErrorMLRDummy<-sumaErrorMLRDummy+mean((prediccionMLRDummy-
data_rg_testMLRDummy$propRetrasos)^2)
}
sumaErrorMLRDummy/50
```

Todos los errores

```{r}
print("MLR sin categóricas")
sumaErrorMLRSinCat/50
print("MLR categóricas sustituidas")
sumaErrorMLRCatMed/50
print("MLR con dummies")
sumaErrorMLRDummy/50
```