



# Desafío LATAM

## Objetivos del desafío:

Predicir la **probabilidad** de que un vuelo (aterrizaje/despegue) se atrase de acuerdo a la data dispuesta en operaciones concentradas en Aeropuerto de Santiago (SCL).

## Desarrollo del modelo:

El análisis en su totalidad fue desarrollado en **GoogleColab** para su difusión y análisis. El código para este desafío es posible encontrar en el siguiente link <https://drive.google.com/file/d/1HReIAQJ4HwengE6t0wVwxoZx8D-s-12Y/view?usp=sharing> (Acceso restringido para usuarios con permisos).

## Explicación del código:

El siguiente código, instala las librerías necesarias para el desarrollo del modelo en Colab. Debido al tiempo para el desarrollo de este desafío, se utilizará **pycaret** para la comparación de modelos respecto a las metricas accuracy, precision y recall. **Pycaret** es una librería open-source "low-code" que permite entrenar multiples modelos de manera eficiente utilizado para investigación en modelos de "prueba de concepto". Para mas informacion se adjunta el siguiente link: <https://pycaret.org/>

```
# Instalación de librerías necesarias para el desarrollo del modelo.  
!pip install pycaret
```

Las librerías utilizadas para el desarrollo de este desafío se importan en la 2da linea de código.

```
# Importar Librerías necesarias para el desafío  
import pandas as pd  
import numpy as np  
from pycaret.classification import *  
from sklearn.model_selection import train_test_split  
from sklearn.preprocessing import StandardScaler  
from sklearn.preprocessing import LabelEncoder  
import matplotlib.pyplot as plt  
%matplotlib inline  
from datetime import datetime, time, timedelta  
import seaborn as sb  
import time
```

Mediante **pandas** se realiza la carga del dataset entregado "**dataset\_SCL.csv**" cuyos datos se visualizan previamente con `df.head()`. Una vez cargados los datos se hace uso de las diferentes funciones integradas en el algoritmo que permiten poder comentar algunas cosas:

a) El dataset contiene 1 fila con datos nulos, el cual fue tratado dentro del desarrollo del código.

b) Mediante la función `"syntetic_features"` se procede a crear las variables solicitadas en el desafío exportándolas directamente en un .csv.

c) Para efectos de mejora en el modelo, se utilizó data relacionada al clima o condiciones meteorológicas por día, las cuales fueron unidas al dataset df mediante `pd.merge(how='left join')`.

Metadata:

Date: Fecha

Maximum Temperature: Maxima temperatura del dia

Minimum Temperature: Mínima temperatura del día

Temperature: Temperatura promedio del día

Precipitation: Cantidad de mm de lluvia caída

Wind Speed: Velocidad promedio del viento

Wind Direction: Direccion del viento

Visibility: Visibilidad en Kms

Cloud Cover: Porcentaje de Nubes en el cielo

Relative Humidity: Humedad Relativa

Conditions: Tipo de dia

d) Luego de aplicar la función `"unique_values"` es posible declarar lo siguiente:

1) La distribución de `vlo-I` se considera homogénea.

2) El 100% de los datos de la variable `ori-I` se concentra en SCEL indicando que todos los vuelos tienen como ciudad de origen Santiago de Chile (Aeropuerto Arturo Merino Benítez) lo cual a priori indica que la data entregada son en 100% despegues desde SCEL hacia otra ciudad.

3) Respecto a los datos, los 3 vuelos mas comunes son a la ciudad de Antofagasta (SCFA - 8.48%), Lima (SPJC - 7.73%) y Calama (SCCF - 7.54%). Los siguientes 17 destinos, concentran el 80% de los vuelos.

Des-I - Cantidad - Frecuencia - Frecuencia Acumulada

SCFA	5787	8.484591	8.484591
SPJC	5270	7.726593	16.211184
SCCF	5145	7.543325	23.754508
SCTE	4357	6.388001	30.142509
SCIE	3995	5.857256	35.999765
SCDA	3747	5.493652	41.493417
SBGR	3570	5.234144	46.727561
SAEZ	3240	4.750315	51.477876
SABE	3094	4.536258	56.014134
SCQP	2583	3.787057	59.801191
SCAR	2436	3.571533	63.372724
SCSE	2410	3.533413	66.906137
SCCI	2105	3.086239	69.992376
MPTO	1850	2.712371	72.704747
SCAT	1780	2.609741	75.314488
SAME	1625	2.382488	77.696977
SKBO	1604	2.351699	80.048676

4) En relación a la aerolínea programada, el 55.14% de los datos, es un vuelo operado por LAN, por lo que LAN, SKY, y TAM realizan el 80.57% de los vuelos registrados.

5) Un 6.62% de los vuelos, son operados a priori por una aerolínea diferente a la programada. Esto podría entenderse luego de realizar una investigación de relación en base a la operación del negocio.

6) No existe una clara diferenciación en cantidad de vuelos generados por Mes. Es posible verificar que la frecuencia de vuelo es mayor en Diciembre y Enero, y En Abril y Junio son los meses con menores vuelos.

7) Los días con mayor frecuencia de vuelo son Viernes, Jueves y Lunes (44.98% acumulado). Esto es factible entender debido a los precios en días de semana vs días de fin de semana.

8) Para e TIPOVUELO, se aprecia que el 54.2% es considerado "Nacional" lo cual hace sentido en relación a los destinos mas habituales.

9) Las aerolíneas con mayor cantidad de vuelos son Grupo LATAM (60%), Sky-Airline (20.96%) las cuales ambas representan el 80.91% de los vuelos registrados.

10) Para el campo "SIGLADES", el destino mas habitual es Buenos Aires con un 9.28%, seguido de Antofagasta, Lima y Calama. Sin embargo, este destino no aparece dentro de los mas visitados por Des-I. Esto se explica debido que para Buenos Aires aparecen 3 siglas de Destino lo que hace aperturar el dato. (SAEZ, SABLE, y SEGU).

11) Respecto a la distribución del target a predecir "atraso\_15" se tiene 81.51% para vuelos sin atraso y un 18.49% con atraso. Esto es considerado un dataset balanceado, en función de las mejores practicas que aluden a tener un 80-20.

12) Un 27.86% de los vuelos ocurre en "temporada alta" de acuerdo a las fechas dispuestas en el desafío.

13) El 74.56% de los vuelos, ocurren durante "mañana" y "tarde". Solo el 25.43% ocurre durante la "noche"

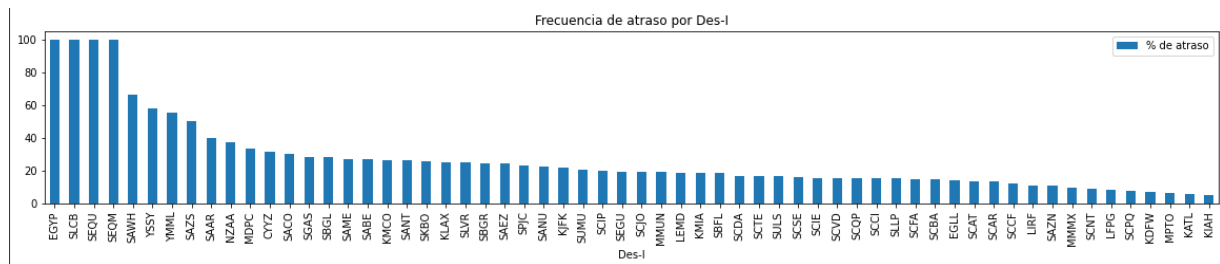
14) Si bien, el objetivo es predecir el atraso de un vuelo. Debido al desconocimiento del negocio, no es posible afirmar que aquellas variables se tengan al momento de correr el algoritmo para predecir. Por ende, por ahora se utilizaran para obtener datos, pero no serán utilizadas en el modelo.

15) Un 27.37% de los vuelos programados, son operados finalmente por otra aerolínea.

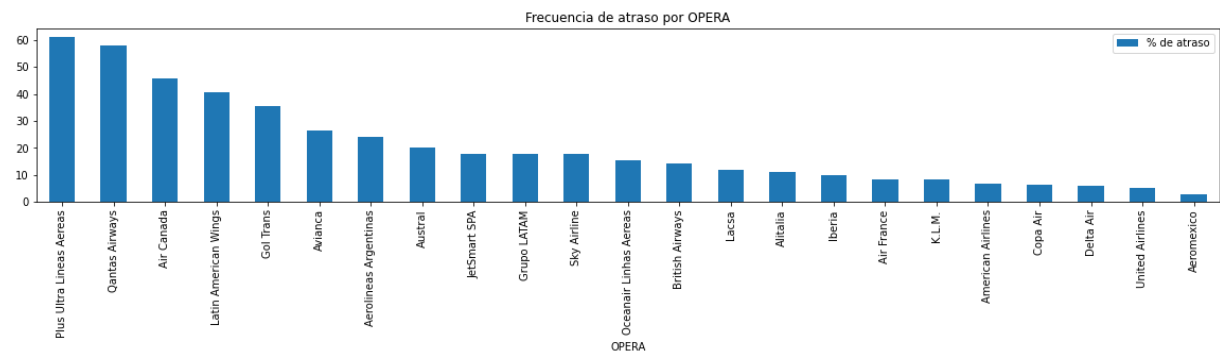
16) El 57.08% de los vuelos, fueron realizados con condiciones climáticas "totalmente despejado".

e) Luego de aplicar la función "frecuencia\_atraso" es posible obtener los siguientes resultados para los atrasos. (Considerar índice de retraso en la distribución de atrasos y no atrasos en los viajes de acuerdo a cada variable)

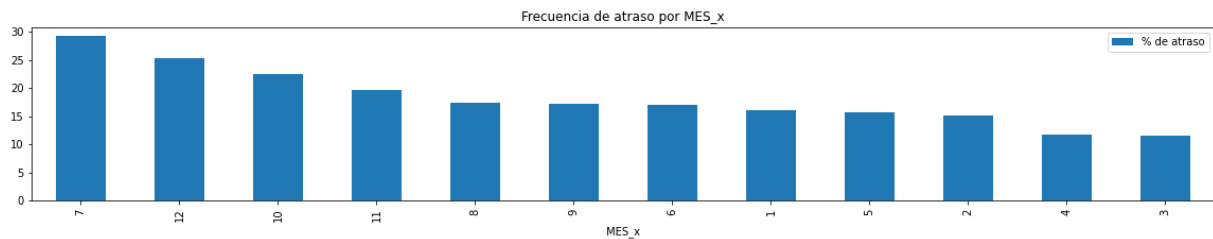
1) Los **destinos** con mayor índice retraso. Para efectos prácticos se utilizo una medida relativa lineal, y no ponderada por la cantidad de viajes que se realizaron.



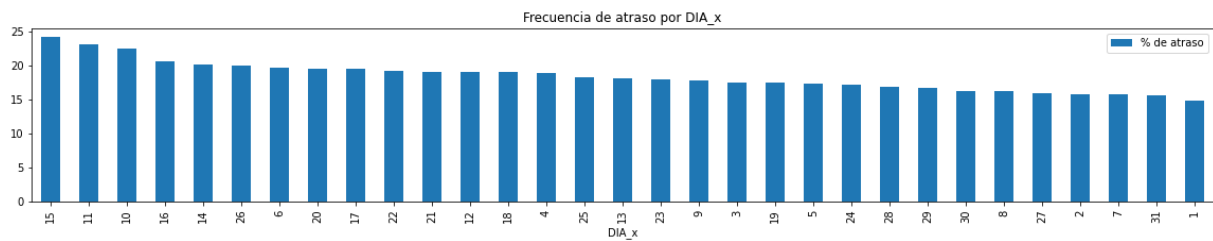
2) Las **aerolíneas** con mayor **índice de retraso** donde el grupo LATAM cuenta con un índice cercano al 20%.



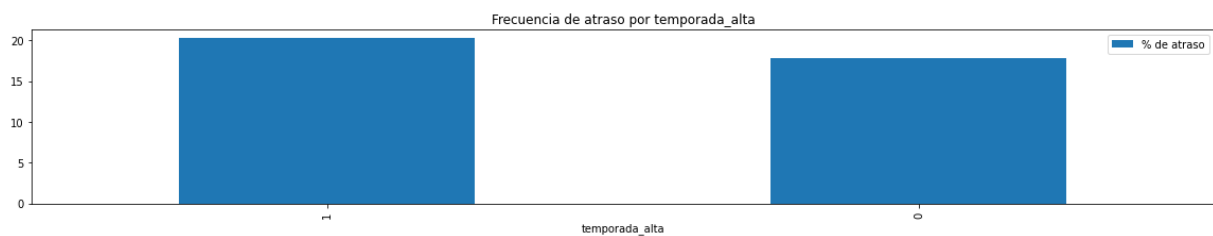
3) Los meses con mayor índice de retraso son Julio y Diciembre. Esto tiene relación con la frecuencia de vuelo en los meses en cuestión. Sin embargo, Siendo Enero, uno de los meses con mayor frecuencia, no registra mismos niveles de retrasos.



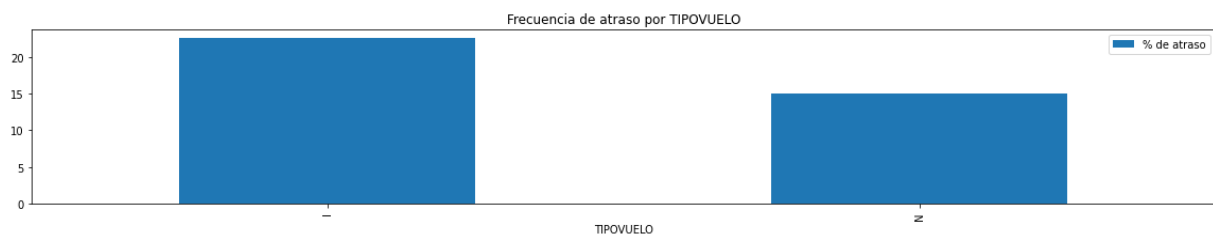
4) La distribución del índice de retraso es mas bien homogénea, donde los días 15, 11 y 10 se percibe un valor superior al resto de los días.



5) Para los vuelos considerados "temporada alta" se tiene un 20% de atraso, mientras que en "temporada baja" se percibe un 18%. Si bien, tiene relación al tipo de temporada, se esperaba contar con mayor retraso en alta que baja.



6) Los vuelos internacionales cuentan con mayor índice de atraso que los vuelos nacionales. Esto tiene relación a la distancia de viaje probablemente, aviones mas grandes, etc.

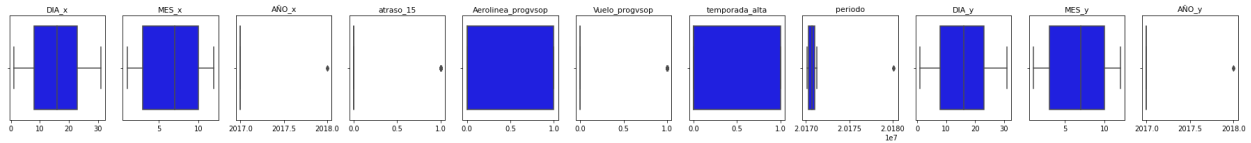


7) De acuerdo a lo estudiado hasta ahora, las variables que tendrían un mayor poder predictivo son:

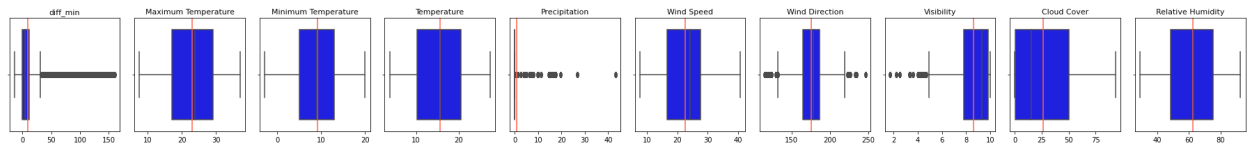
- Aerolínea que opera
- Día de la semana
- Tipo de Vuelo
- Mes del año
- Destino del viaje

f) Una vez que se tiene el dataset completo, utilizando datos tanto de vuelos como de clima, se procede a aplicar la función `"plot_boxplot"` para revisar si las variables cuentan con outliers.

Observando la siguiente imagen, es posible identificar que la data de vuelos se encuentra sin outliers. Por consecuencia de que la data posee datos en su mayoría de año 2017, las variables relacionadas a esto son detectadas como outliers, sin embargo, no son parte del comportamiento atípico.

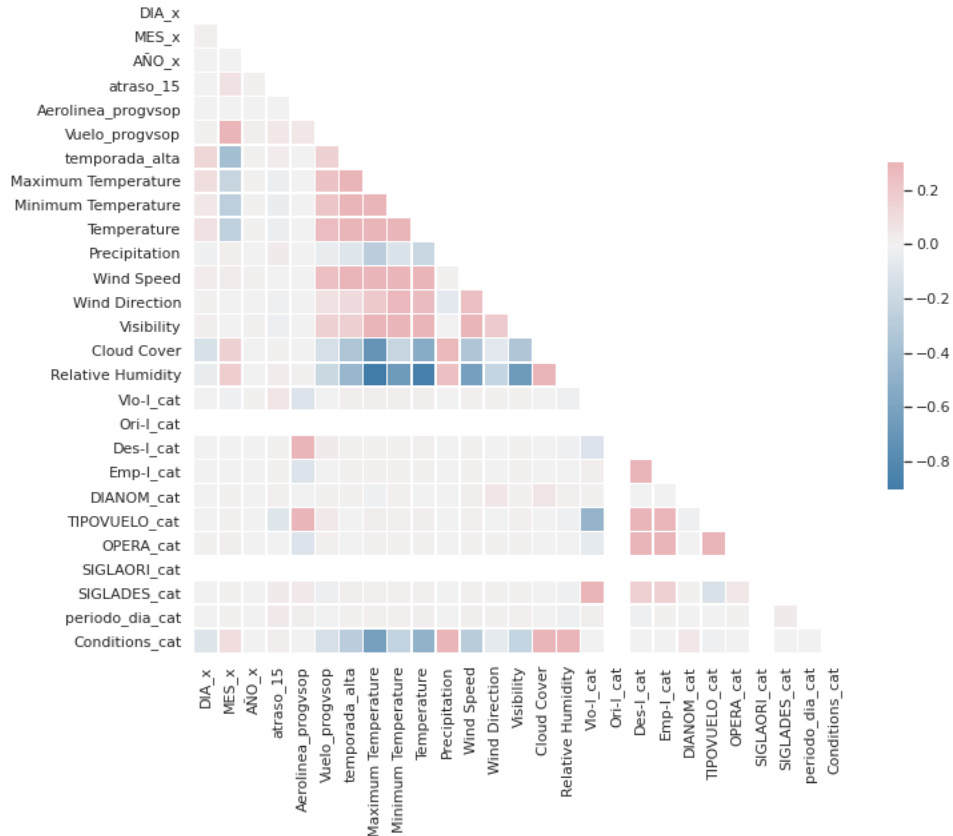


Al revisar variables relacionadas a condiciones meteorológicas (más variable `diff_min`) se tiene que algunas de las variables presentan outliers. Por ejemplo, precipitación debido a que Santiago no es muy común que llueva (al menos los últimos 5 años), dirección del viento y visibilidad. (Se tratarán luego los outliers).



Antes de realizar el entrenamiento del modelo, se aplica `"LabelEncoder"` desde sklearn para convertir variables categóricas en numéricas (para la aplicación de los diferentes modelos). En la configuración de los parámetros del modelo, se especifican el tipo de variable a considerar en el entrenamiento.

Calculando la correlación entre las variables:



## Entrenamiento del modelo:

La siguiente línea de código, configura los parámetros necesarios para la aplicación de `Pycaret` a nuestro dataset. En esta configuración se señala cual es la data a utilizar, cual será el target a predecir, aplicando estandarización a las variables y especificando cuales son categóricas y cuales numéricas. Para efecto de entrenamiento, `Pycaret` elimina `outliers` (utilizando rangos inter-cuartiles), elimina variables correlacionadas y utiliza optimización bayesiana de `hyperparametros`.

```
#Aplicacion de Pycaret
mod = setup(df_out, target='atraso_15', normalize = True, categorical_features = ['Aerolinea_progsop', 'Vuelo_progsop', 'temporada_alta',
                                      'DIANOM_cat', 'TIPOVUELO_cat', 'OPERA_cat', 'SIGLAORI_cat', 'W',
                                      numeric_features = ['DIA_x', 'MES_x', 'AÑO_x', 'Maximum Temperature', 'Minimum Temperature', 'Temperature', 'Precipitation', 'Wi',
                                      'Visibility', 'Cloud Cover', 'Relative Humidity'], remove_outliers=True)
```

```
# Comparando modelos
compare_models()
```

`Compare_models()` permite entrenar múltiples modelos de ML utilizando Cross-Validation con 10 folds, separando la data en 80% entrenamiento y 20% para validación.

El resultado del proceso, es el siguiente:

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
<b>lightgbm</b>	Light Gradient Boosting Machine	0.8247	0.7344	0.1206	0.6473	0.2033	0.1540	0.2247	1.378
<b>rf</b>	Random Forest Classifier	0.8204	0.7230	0.1510	0.5583	0.2376	0.1723	0.2204	27.070
<b>gbc</b>	Gradient Boosting Classifier	0.8202	0.6965	0.0582	0.6840	0.1070	0.0802	0.1616	41.601
<b>lr</b>	Logistic Regression	0.8195	0.6964	0.0930	0.5821	0.1602	0.1151	0.1782	31.345
<b>ridge</b>	Ridge Classifier	0.8182	0.0000	0.0821	0.5661	0.1433	0.1012	0.1628	0.689
<b>svm</b>	SVM - Linear Kernel	0.8181	0.0000	0.0583	0.5979	0.1061	0.0757	0.1438	2.771
<b>ada</b>	Ada Boost Classifier	0.8180	0.6856	0.0727	0.5716	0.1290	0.0910	0.1544	9.626
<b>et</b>	Extra Trees Classifier	0.8170	0.7080	0.1802	0.5187	0.2673	0.1899	0.2249	40.441
<b>lda</b>	Linear Discriminant Analysis	0.8148	0.6919	0.1472	0.5021	0.2276	0.1567	0.1953	10.096
<b>knn</b>	K Neighbors Classifier	0.8048	0.6548	0.1854	0.4381	0.2604	0.1687	0.1896	166.315
<b>dt</b>	Decision Tree Classifier	0.7534	0.5819	0.3090	0.3260	0.3172	0.1669	0.1670	2.939
<b>qda</b>	Quadratic Discriminant Analysis	0.6405	0.5073	0.2956	0.1951	0.2302	0.0133	0.0135	7.491
<b>nb</b>	Naive Bayes	0.2903	0.5438	0.9339	0.1989	0.3280	0.0320	0.0898	0.470

Automáticamente los modelos son ordenados de mejor performance al peor. Sin embargo, para efectos de métrica, los modelos indicados tienen bajo `F1 Score`, específicamente debido al bajo `Recall` que capturan los datos. Si bien el `Accuracy`, y el `AUC` son relativamente aceptables, es un error utilizar solo estas métricas para modelos de clasificación (Modelo entrenado para reconocer si un vuelo se atrasara o no), puesto que el 80% de los datos son sin atraso. Dicha razón, explica el motivo por el cual generalmente se utiliza como métricas `Precisión y Recall` o mas bien `F1-Score` que combina ambos. Estas métricas explican cuan preciso o bien generaliza el modelo para identificar ambas clase (precisión) y cuantas de los datos es capaz de captar correctamente (recall).

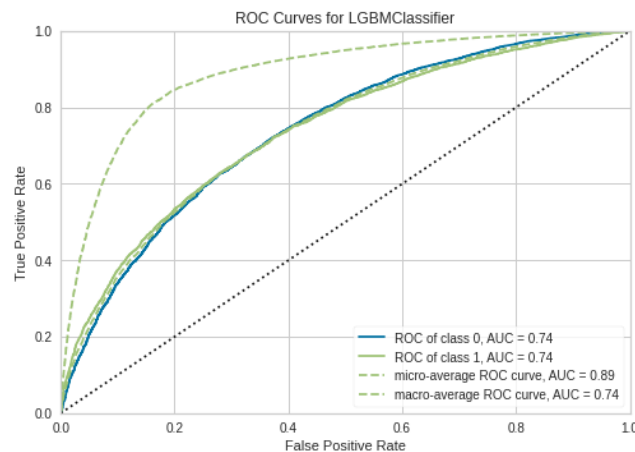
Buscando una mejora en los resultados, se procede a aplicar la función "`tune_model`" de `pycaret`, la cual optimiza los `hyperparametros` del modelo elegido (lightgbm) utilizando optimización bayesiana para evitar el `overfitting` de parámetros. Luego de aplicar, los resultados tienden a mejorar debido a una mejora del `Recall`. El siguiente gráfico muestra el K-Fold 10 para el dataset de entrenamiento.

	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	0.8272	0.7376	0.2090	0.5968	0.3096	0.2362	0.2790
1	0.8258	0.7287	0.2065	0.5865	0.3055	0.2312	0.2727
2	0.8228	0.7479	0.2054	0.5617	0.3008	0.2237	0.2611
3	0.8255	0.7232	0.2043	0.5858	0.3029	0.2289	0.2708
4	0.8205	0.7417	0.2056	0.5417	0.2981	0.2184	0.2523
5	0.8249	0.7290	0.2034	0.5788	0.3010	0.2264	0.2672
6	0.8232	0.7272	0.1955	0.5672	0.2908	0.2163	0.2567
7	0.8211	0.7350	0.1955	0.5492	0.2883	0.2116	0.2488
8	0.8215	0.7427	0.2000	0.5514	0.2935	0.2162	0.2529
9	0.8215	0.7113	0.1898	0.5545	0.2828	0.2079	0.2472
Mean	0.8234	0.7324	0.2015	0.5674	0.2973	0.2217	0.2609
SD	0.0022	0.0102	0.0058	0.0177	0.0079	0.0086	0.0105

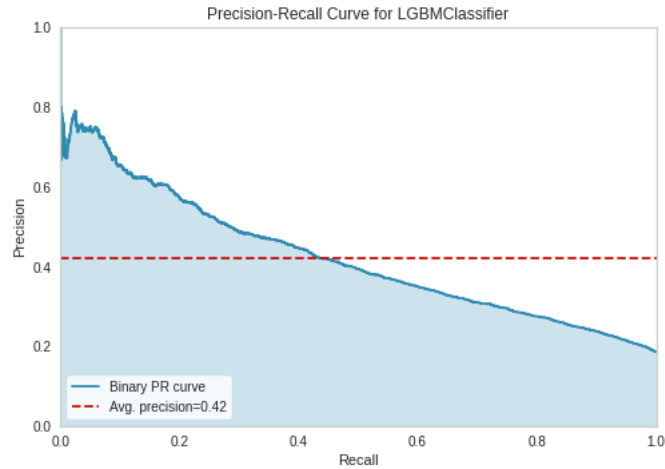
Para poder comprender mejor el resultado del modelo, se debe utilizar una función de costo para obtener el real potencial de predicción. En la práctica, modelos con 50% de **accuracy** suelen generar impactos significativos en ahorros de costo o mayores márgenes. En general, todo depende del caso de estudio y las variables propias del modelo de negocio. Por ejemplo, es totalmente diferente predecir un fraude a predecir el churn rate de un cliente. En el primero, nuestro modelo debe tener excelente precisión puesto que no puedo dictar de fraude a un cliente real por un falso positivo, sin embargo, en el segundo, nuestro modelo debe tener buena **precisión** y **accuracy** para detectar clientes que hoy se fugan de la compañía (para generar un plan de retención), pero no con la misma precisión que el modelo de fraudes. No obstante, como mencione anteriormente, todo depende de la función de coste.

#### Importancia de variables y otros gráficos (pycaret):

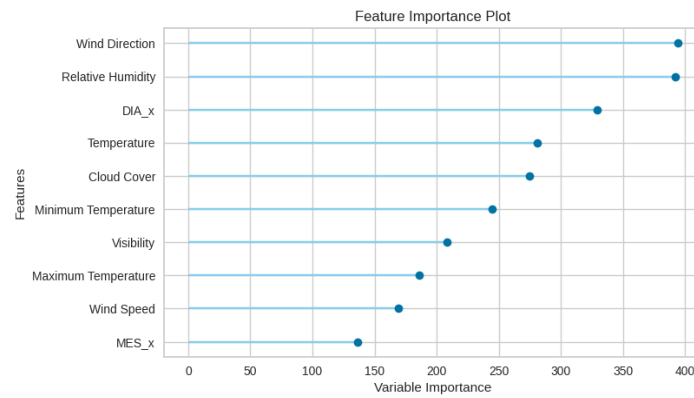
**AUC:** Es posible apreciar, que el área bajo la curva ROC para cada clase es homogénea.



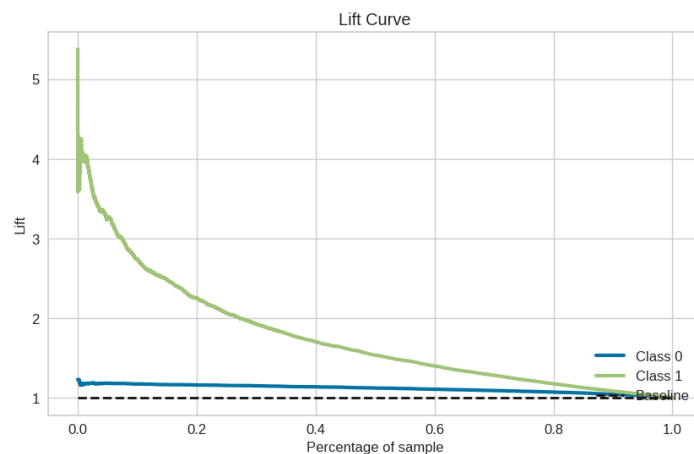
**Precision-Recall Tradeoff:** Se evidencia que el corte optimo entre precision y recall se encuentra alrededor 40%, es decir, se recomienda definir como "atraso" todo lo que se encuentre sobre el 40% de probabilidad.



**Feature Importance:** Se evidencia que las variables climatológicas son de gran importancia para la generalización del modelo.

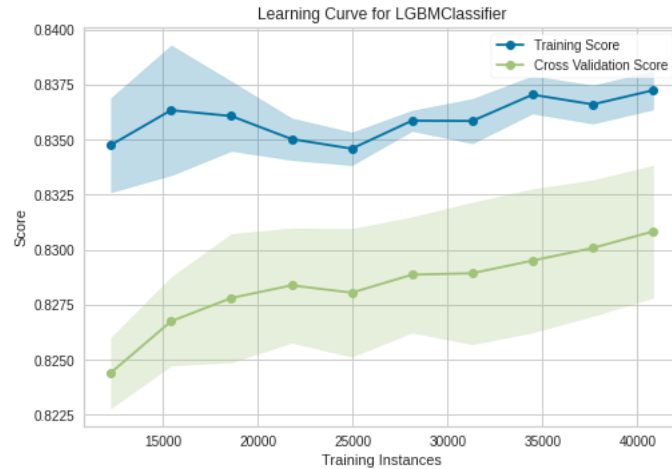


**Lift Curve:** Este grafico permite demostrar que hasta el 2o decil (de acuerdo a la probabilidad predicha), el modelo logra generalizar correctamente lo que permite identificar aquellos vuelos con mayor propensión a atrasarse.



**Learning Curve:** El siguiente grafico permite conocer la curva de aprendizaje del modelo utilizado, cuyo training y testing demuestran resultado muy similares los cuales permiten indicar que el modelo no se encuentre en overfitting.





**Shap Value:**


Un grafico interesante para observar, es el adjunto. **SHAP value** permite conocer el impacto tanto positivo como negativo en la interacción de cada variable con la variable a predecir. Es un grafico que acompaña a la importancia de variables que permiten interpretar de mejor manera el porque de la selección de cada variable y su poder predictivo.

Para comprender mejor este indicador, dejo la siguiente URL:

### Explain Your Model with the SHAP Values

Better Interpretability Leads to Better Adoption Is your highly-trained model easy to understand? A sophisticated machine learning algorithms usually can produce accurate predictions, but its notorious "black box" nature does not help adoption at all. Think about this: If you ask me to swallow a black pill without telling me what's in it, I

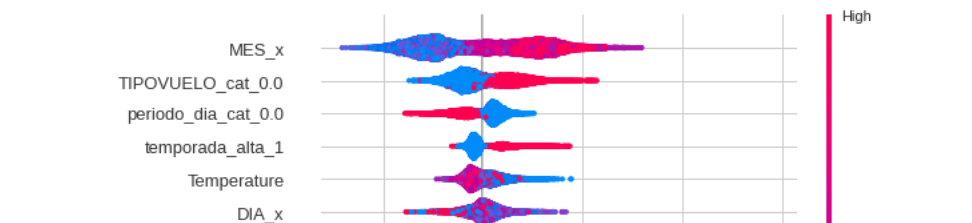
<https://towardsdatascience.com/explain-your-model-with-the-shap-values-bc36aac4de3d>



**No Longer a Black Box**

higher · 27 lower  
5.326 5.376 5.426 5.476 5.526 5.576 5.611 5.626 5.676 5.726 5.776 5.826 5.876 5.926

Shy = 0.58 / density = 0.8951 / total sulfur dioxide = 10 / accuracy = 11 / nitrogen = 0.58 / pH = 2.55 / nitric acid = 0.01



Feature	SHAP Value Range (approx.)
MES_x	-0.4 to 0.8
TIPOVUELO_cat_0.0	-0.3 to 0.6
periodo_dia_cat_0.0	-0.2 to 0.4
temporada_alta_1	-0.1 to 0.5
Temperature	-0.2 to 0.5
DIA_x	-0.3 to 0.4
Minimum Temperature	-0.2 to 0.6
Wind Direction	-0.2 to 0.4
Relative Humidity	-0.2 to 0.4
Visibility	-0.2 to 0.6
OPERA_cat_12.0	-0.2 to 0.3
DIANOM_cat_6.0	-0.1 to 0.4
OPERA_cat_17.0	-0.1 to 1.5
Cloud Cover	-0.2 to 0.3
OPERA_cat_21.0	-0.2 to 0.4
Wind Speed	-0.2 to 0.4
Des-I_cat_37.0	-0.4 to 0.2
SIGLADES_cat_8.0	-0.1 to 0.4
Maximum Temperature	-0.2 to 0.3
periodo_dia_cat_1.0	-0.2 to 0.3

### Mejoras del modelo:

A continuación se indican diferentes formas de mejorar el modelo:

**a) Aumentar cantidad de datos:** Al utilizar mayor cantidad de ejemplos, permite al modelo poder adquirir patrones y generalizar mejor durante el entrenamiento.

**b) Agregar variables:** Las variables del dataset entregado, parecen no ser suficientes para mejorar el poder predictivo de nuestro modelo. Es por esta razón que se podría reentrenar con variables como:

- 1) Marca y Modelo del avion
- 2) Cantidad de pasajeros disponibles
- 3) Cantidad de motores y potencia
- 4) Velocidad promedio del avión durante los vuelos
- 5) Días a la ultima mantención del avión según manuales técnicos.
- 6) Tripulación, Piloto.
- 7) Millas recorridas del avión
- 8) Experiencia en años de los pilotos.
- 9) Distancia entre origen y destino
- 10) Utilizar condiciones climáticas en tiempo real o por hora (actualmente el dataset obtenido solo permite ser por día)

**c) Utilizar diferente metodología:** Para poder adquirir data de comportamiento, una manera de poder aumentar el poder predictivo en función del tipo de caso de estudio, es utilizar "series de tiempo".

**d) Utilizar librerías de feature engineering:** Hacer uso de librerías como "featuretools" para así utilizar Deep Feature Synthesis generando n cantidad de variables de comportamiento que permitan aumentar el rendimiento del modelo.

**e) Redes neuronales:** Hacer uso de redes neuronales recurrentes LSTM podría ser una buena manera de comparar resultados actuales y evidenciar si al hacer uso de Deep Learning el modelo mejora.