# FLIGHTS 2015 DATASET

Data Mining project

Jaume Pérez Medina

Curs 2023 / 2024

MIDA

# ÍNDEX

# 1. DESCRIPCIÓ DE LES DADES ORIGINALS

He decidit analitzar els vols domèstics que es van realitzar als Estats Units d'Amèrica durant l'any 2015, centrant-me a determinar si un vol ha tingut alguna complicació (retard, cancel·lació o desviació a un altre aeroport). He decidit escollir un dataset amb dades reals per a trobar-me amb problemes que poden sorgir a l'analitzar dades no simulades.

L'enllaç al dataset és el següent: https://www.kaggle.com/datasets/usdot/flight-delays. Cal destacar la seva mida, de 5819079 files i 31 columnes.

El dataset escollit té les següents columnes:
**'YEAR'**: Any en que s'ha realitzat el vol
**'MONTH'**: Mes en que s'ha realitzat el vol
**'DAY'**: Dia del mes en que s'ha realitzat el vol
**'DAY_OF_WEEK'**: Dia de la setmana en que s'ha realitzat el vol
**'AIRLINE'**: Codi de l'aerolínia
**'FLIGHT_NUMBER'**: Número identificatiu de la ruta que ha realitzat el vol
**'TAIL_NUMBER'**: Número identificatiu de l'avió que ha realitzat el vol
**'ORIGIN_AIRPORT'**: Aeroport d'origen
**'DESTINATION_AIRPORT'**: Aeroport de destinació
**'SCHEDULED_DEPARTURE'**: Hora programada de sortida
**'DEPARTURE_TIME'**: Hora de sortida
**'DEPARTURE_DELAY'**: Retard en la sortida
**'TAXI_OUT'**: Temps de taxi a la pista de sortida
**'WHEELS_OFF'**: Hora en què l'avió s'enlaira (deixa de tocar el terra)
**'SCHEDULED_TIME'**: Temps programat de vol
**'ELAPSED_TIME'**: Temps transcorregut de vol
**'AIR_TIME'**: Temps d'aire de vol
**'DISTANCE'**: Distància de vol
**'WHEELS_ON'**: Hora en què l'avió aterra
**'TAXI_IN'**: Temps de taxi a la pista d'arribada
**'SCHEDULED_ARRIVAL'**: Hora programada d'arribada
**'ARRIVAL_TIME'**: Hora d'arribada
**'ARRIVAL_DELAY'**: Retard a l'arribada
**'DIVERTED'**: Si l'avió va ser desviat a un altre aeroport de destinació.
**'CANCELLED'**: Si el vol ha estat cancelat.
**'CANCELLATION_REASON'**: Motiu de cancel·lació
**'AIR_SYSTEM_DELAY'**: Retard a causa del sistema d'aire
**'SECURITY_DELAY'**: Retard per motius de seguretat
**'AIRLINE_DELAY'**: Retard a causa de l'aerolínia
**'LATE_AIRCRAFT_DELAY'**: Retard a causa de l'avió
**'WEATHER_DELAY'**: Retard a causa del clima

El target del dataset serà una nova columna anomenada DISRUPTED, que indicarà si el vol ha estat cancel·lat, derivat a un altre aeroport de destinació o ha arribat més tard del que s'indicava en un principi. Explicaré el procés de creació d'aquesta nova columna i com la crearé al següent apartat.

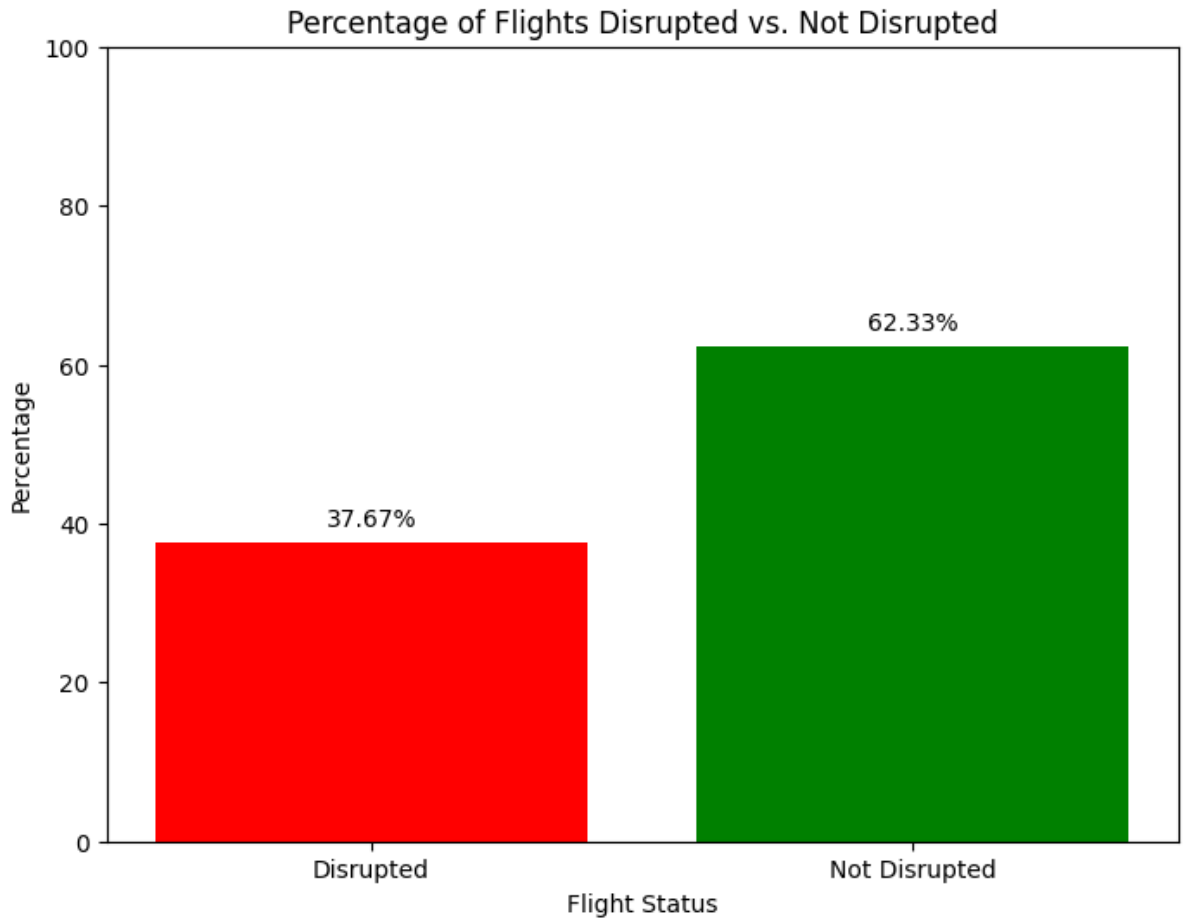
# 2. DESCRIPCIÓ DEL PREPROCESSING

## SELECCIÓ DE LA VARIABLE TARGET

Primer he creat la columna de target, anomenada DISRUPTED. Un vol tindrà la columna DISRUPTED a 1 si compleix alguna de les tres condicions següents:

- Ha estat cancel·lat (CANCELLED = 1)
- Ha estat derivat a un altre aeroport (DIVERTED = 1)
- Ha arribat més tard del que s'esperava (ARRIVAL_DELAY > 0)

Concretament, hi ha 2191967 vols disruptats i 3627112 vols que no ho estan.




Com es pot observar a la imatge el resultat és un dataset desbalancejat, amb un 37.67% de vols disruptats i un 62.33% que no.

Pot semblar contraintuïtiu pensar que la majoria dels vols arriben abans d'hora, ja que no satisfan la tercera norma que hem descrit per a un vol DISRUPTED (ARRIVAL_DELAY > 0). Això és perquè moltes aerolínies planifiquen el temps d'arribada dels seus vols ja comptant amb que hi haurà algun petit retard, i és per això que sovint els vols arriben abans al destí que l'hora planificada d'arribada.

## ELIMINACIÓ DE COLUMNES IRRELLEVANTS

A continuació eliminarem les columnes que no es poden saber en reservar el vol, com per exemple a quina hora ha acabat arribant el vol al seu destí. Aquestes són les següents:

‘'DEPARTURE_TIME'
‘'DEPARTURE_DELAY'
‘'TAXI_OUT’
'WHEELS_OFF'
‘'ELAPSED_TIME'
‘'AIR_TIME'
'WHEELS_ON'
'TAXI_IN'
'ARRIVAL_TIME'
'ARRIVAL_DELAY'
'DIVERTED'
'CANCELLED'
'CANCELLATION_REASON'
'AIR_SYSTEM_DELAY'
'SECURITY_DELAY'
'AIRLINE_DELAY'
'LATE_AIRCRAFT_DELAY'
'WEATHER_DELAY'
'TAIL_NUMBER'

També eliminarem algunes irrellevants, com la columna YEAR. Tots els vols són de l’any 2015, per tant, tots els individus tenen el valor d’aquesta columna a 2015.

Addicionalment eliminarem TAIL_NUMBER. L’objectiu era poder identificar avions que acostumen a tenir falles tècniques i, per tant, més retards. Així i tot, més endavant veurem com reduïm el dataset a 2000 individus, i per aquests pràcticament tots els vols es feien amb diferents avions, i per tant per pràcticament cada individu resultava ser diferent.

Per tant, ens quedem amb un dataset amb 11 columnes:

| | MONTH | DAY | DAY_OF_WEEK | AIRLINE | ORIGIN_AIRPORT | DESTINATION_AIRPORT | SCHEDULED_DEPARTURE | SCHEDULED_TIME | DISTANCE | SCHEDULED_ARRIVAL | DISRUPTED |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | 1 | 4 | AS | ANC | SEA | 5 | 205.0 | 1448 | 430 | 0 |
| **1** | 1 | 1 | 4 | AA | LAX | PBI | 10 | 280.0 | 2330 | 750 | 0 |
| **2** | 1 | 1 | 4 | US | SFO | CLT | 20 | 286.0 | 2296 | 806 | 1 |
| **3** | 1 | 1 | 4 | AA | LAX | MIA | 20 | 285.0 | 2342 | 805 | 0 |
| **4** | 1 | 1 | 4 | AS | SEA | ANC | 25 | 235.0 | 1448 | 320 | 0 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **5819068** | 12 | 31 | 4 | B6 | LAX | BOS | 2359 | 320.0 | 2611 | 819 | 0 |
| **5819069** | 12 | 31 | 4 | B6 | JFK | PSE | 2359 | 227.0 | 1617 | 446 | 0 |
| **5819070** | 12 | 31 | 4 | B6 | JFK | SJU | 2359 | 221.0 | 1598 | 440 | 0 |
| **5819071** | 12 | 31 | 4 | B6 | MCO | SJU | 2359 | 161.0 | 1189 | 340 | 0 |
| **5819072** | 12 | 31 | 4 | B6 | JFK | BQN | 2359 | 221.0 | 1576 | 440 | 1 |

5819073 rows × 11 columns

## ELIMINACIÓ DE VALORS NULS

A continuació ens encarregarem dels valors NaN que queden després d'eliminar les columnes de dalt.

```
Number of rows with any null values: 6
         MONTH  DAY  DAY_OF_WEEK AIRLINE ORIGIN_AIRPORT DESTINATION_AIRPORT  \
483174       2    1            7      NK            FLL                 IAG
619941       2   10            2      NK            FLL                 IAG
1720237      4   20            1      NK            FLL                 LGA
1820960      4   26            7      NK            DEN                 DFW
2031482      5    9            6      NK            MCO                 ACY
2034453      5   10            7      NK            ORD                 BWI

         SCHEDULED_DEPARTURE  SCHEDULED_TIME  DISTANCE  SCHEDULED_ARRIVAL  \
483174                  2215             NaN      1176                107
619941                  2215             NaN      1176                107
1720237                 1602             NaN      1076               1900
1820960                 2059             NaN       641               2350
2031482                 2130             NaN       852               2340
2034453                  825             NaN       622               1118

         DISRUPTED
483174           0
619941           0
1720237          0
1820960          0
2031482          0
2034453          0
```

Podem veure com existeixen 6 individus dels 5,8 milions amb alguns valors a NaN. És un valor ínfim. Si ens fixem en les columnes on no hi ha cap valor establert, podem observar com totes pertanyen a SCHEDULED_TIME. Es tracta d'un error en la recol·lecció de les dades, ja que en teoria totes les columnes que hem deixat haurien de tenir valors definits.

Com tenim molts individus i la representació d'aquests 6 és menyspreable, he decidit eliminar els individus.

## CONVERSIÓ DE DADES CATEGÒRIQUES A NUMERIQUES (AEROPORTS)

Ara ens encarregarem de transformar les dades categòriques a numèriques.
Primerament, transformarem els aeroports, tant d'origen com de destí. Degut al gran nombre d'aquests al dataset (628 aeroports d'origen i 629 de destí), he seleccionat 10 aeroports que són els que analitzaré al meu dataset final. Els aeroports seleccionats són els següents:

```
Total number of unique airports of origin: 628
Total number of unique airports of destination: 629
```

Aeroports seleccionats:

- **ATL**: Aeroport Internacional Hartsfield-Jackson d'Atlanta
- **SFO**: Aeroport Internacional de San Francisco
- **DEN**: Aeroport Internacional de Denver
- **ORD**: Aeroport Internacional O'Hare de Chicago
- **LAX**: Aeroport Internacional de Los Angeles
- **DFW**: Aeroport Internacional de Dallas/Fort Worth

- **SEA**: Aeroport Internacional de Seattle-Tacoma
- **JFK**: Aeroport Internacional John F. Kennedy de Nova York
- **LAS**: Aeroport Internacional McCarran de Las Vegas
- **MIA**: Aeroport Internacional de Miami

Eliminarem la resta d'aeroports del dataset, tant d'origen com de destí, i realitzarem one-hot encoding pels aeroports seleccionats. El dataset ens quedaria així:

```
Index(['MONTH', 'DAY', 'DAY_OF_WEEK', 'AIRLINE', 'SCHEDULED_DEPARTURE',
       'SCHEDULED_TIME', 'DISTANCE', 'SCHEDULED_ARRIVAL', 'DISRUPTED',
       'ORIGIN_AIRPORT_ATL', 'ORIGIN_AIRPORT_DEN', 'ORIGIN_AIRPORT_DFW',
       'ORIGIN_AIRPORT_JFK', 'ORIGIN_AIRPORT_LAS', 'ORIGIN_AIRPORT_LAX',
       'ORIGIN_AIRPORT_MIA', 'ORIGIN_AIRPORT_ORD', 'ORIGIN_AIRPORT_SEA',
       'ORIGIN_AIRPORT_SFO', 'DESTINATION_AIRPORT_ATL',
       'DESTINATION_AIRPORT_DEN', 'DESTINATION_AIRPORT_DFW',
       'DESTINATION_AIRPORT_JFK', 'DESTINATION_AIRPORT_LAS',
       'DESTINATION_AIRPORT_LAX', 'DESTINATION_AIRPORT_MIA',
       'DESTINATION_AIRPORT_ORD', 'DESTINATION_AIRPORT_SEA',
       'DESTINATION_AIRPORT_SFO'],
      dtype='object')
Nombre d'individus: 13007341
Nombre de columnes : 29
```

Com podem observar el nombre d'individus s'ha reduït després d'eliminar els vols amb aeroports que no hem seleccionat, mentre que el nombre de columnes ha augmentat degut al one-hot encoding que hem realitzat.

## REDUCCIÓ DEL DATASET

Ara utilitzarem la funció de train_test_split per a reduir la mida del dataset a 2000 línies, utilitzarem el paràmetre de stratify amb DISRUPTED per a mantenir la proporció entre les dues classes. Tot i que utilitzem la funció per a separar entre test i training sets, només crearem un dataset de 2000 línies. He utilitzat aquesta funció per a poder utilitzar el paràmetre stratify.

```
[2000 rows x 29 columns]>
```

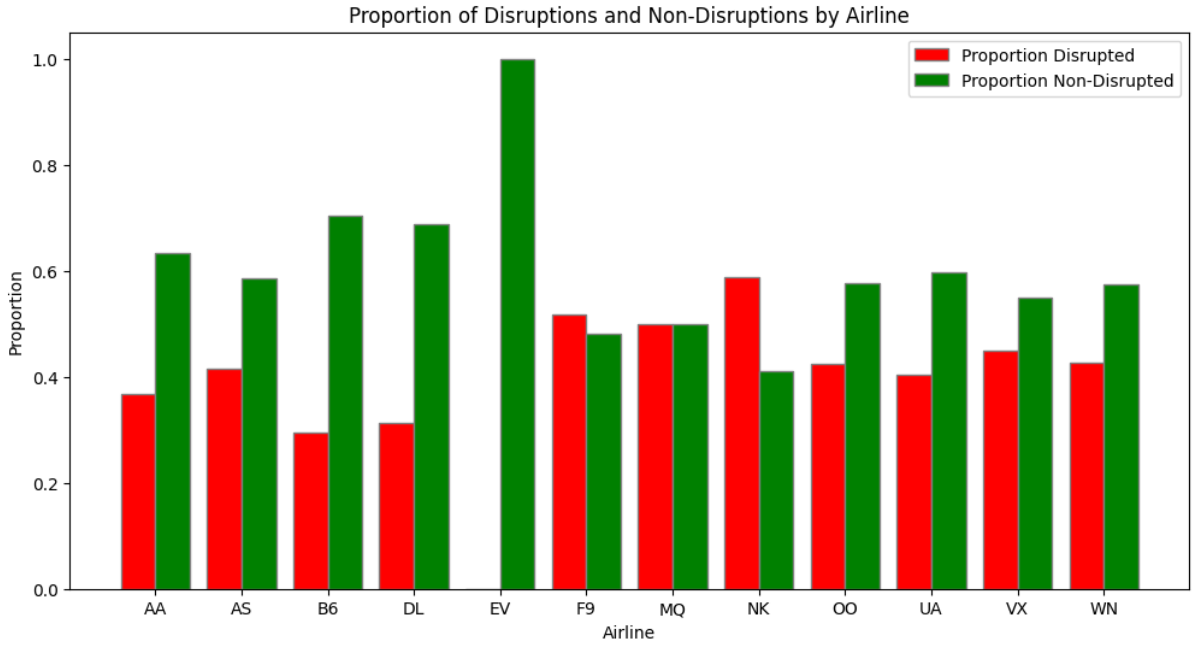
Ens queda, per tant, un dataset de 2000 files i 29 columnes.

## CONVERSIÓ DE DADES CATEGÒRIQUES A NUMERIQUES (AIRLINES)

Una vegada reduït el dataset encara ens queda una columna categòrica: AIRLINE

Per a aquestes inicialment havia decidit agrupar-les per tipus d'aerolínia (low-cost, standard o premium). Tot i això, al mercat dels Estats Units costava segmentar les aerolínies en aquests grups, ja que tenien diferents classes i el mercat és diferent de l'europeu, on clarament es poden identificar les aerolínies low-cost (Vueling, Ryanair, Wizz air), tenint també més vols cancel·lats o retards que la resta.

Un altre inconvenient era que després de segmentar algunes aerolínies segons el seu cost mitjà i els serveis que ofereix els vols disruptats no tenien una correlació amb el tipus d'aerolínia:




Low-cost: B6 (Blue Jet), NK (Spirit Airlines)
Premium: DL (Delta) MQ (Envoy Air)

Per exemple, B6 i DL tenen una ràtio similar, inclús potser B6 té més vols no disruptats que Delta, sent una de baix cost i l'altra premium. Tot i que NK és de baix cost i és la que més vols disruptats té, la segueix de prop MQ, una altra aerolínia que no és de baix cost.

És per això, que he decidit segmentar les aerolínies en si tenen més vols disruptats o no dins el dataset de 2000 files.

Per tant, he creat una columna anomenada AIRLINE_TENDS_TO_DISRUPTIONS que estarà a 1 si l'aerolínia en qüestió té més vols disruptats que de sense disrupcions o a 0 en cas contrari.

```
Airlines that tend to disruptions: ['F9' 'NK']
```

En aquest cas tindran la columna a 1 les aerolínies F9 i NK.

He decidit reduir el nombre d'aerolínies i no fer one-hot encoding per a cada aerolínia degut als meus recursos per a executar els models i així reduir el nombre de columnes.

Finalment eliminem la columna AIRLINE.

## TRACTAMENT COLUMNES CÍCLIQUES

He implementat mitjançant la funció del sinus el tractament de les columnes cícliques, és a dir, que l'extrem superior i inferior realment estaven al costat. Les columnes que he tractat així són les següents:

DAY, DAY_OF_WEEK, MONTH, SCHEDULED_DEPARTURE i SCHEDULED_ARRIVAL.

Després d'aplicar aquest procés les columnes es veurien així:

| | MONTH | DAY | DAY_OF_WEEK | SCHEDULED_DEPARTURE | SC |
|---|---|---|---|---|---|
| **0** | 0.500000 | 0.005766 | 0.283058 | 0.996534 | |
| **1** | 0.066987 | 0.550584 | 0.109084 | 0.313506 | |
| **2** | 0.933013 | 0.968876 | 0.890916 | 0.980510 | |
| **3** | 0.933013 | 0.257349 | 0.716942 | 0.688920 | |
| **4** | 0.750000 | 0.137604 | 0.716942 | 0.917184 | |
| **...** | ... | ... | ... | ... | |
| **1995** | 0.250000 | 0.948902 | 0.012536 | 0.493455 | |
| **1996** | 0.066987 | 0.862396 | 0.987464 | 1.000000 | |
| **1997** | 0.933013 | 0.005766 | 0.012536 | 0.446434 | |
| **1998** | 0.066987 | 0.449416 | 0.500000 | 0.146447 | |
| **1999** | 0.500000 | 0.015961 | 0.890916 | 0.844177 | |

2000 rows × 29 columns

## NORMALITZACIÓ

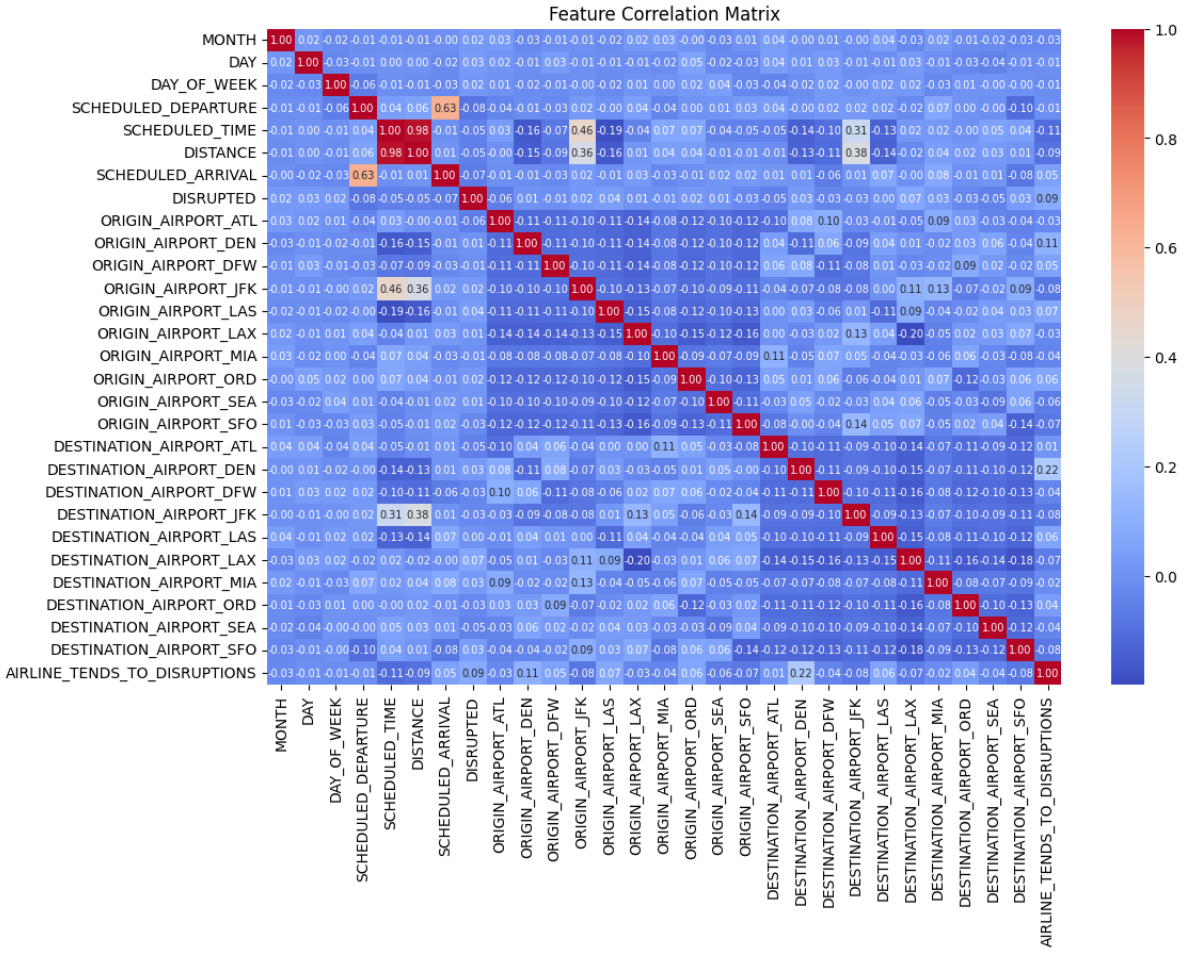
Per a normalitzar només he hagut d'aplicar aquest procediment amb MinMxScaler per a les columnes SCHEDULED_TIME (temps que durarà el vol) i DISTANCE (distància que recorrerà el vol).

| SCHEDULED_TIME | DISTANCE |
|---|---|
| 0.027933 | 0.000000 |
| 0.377095 | 0.387058 |
| 0.779330 | 0.944534 |
| 0.268156 | 0.316720 |
| 0.122905 | 0.071543 |
| ... | ... |
| 0.553073 | 0.572347 |
| 0.175978 | 0.178055 |
| 0.446927 | 0.513666 |
| 0.181564 | 0.143891 |
| 0.251397 | 0.253617 |

Ara ja tenim totes les columnes entre 0 i 1 (cícliques i que indiquen qualitats ( one-hot encoding i les creades per mi DISRUPTED i AIRLINE_TENDS_TO_DISRUPTIONS).

## CORRELACIÓ ENTRE FEATURES

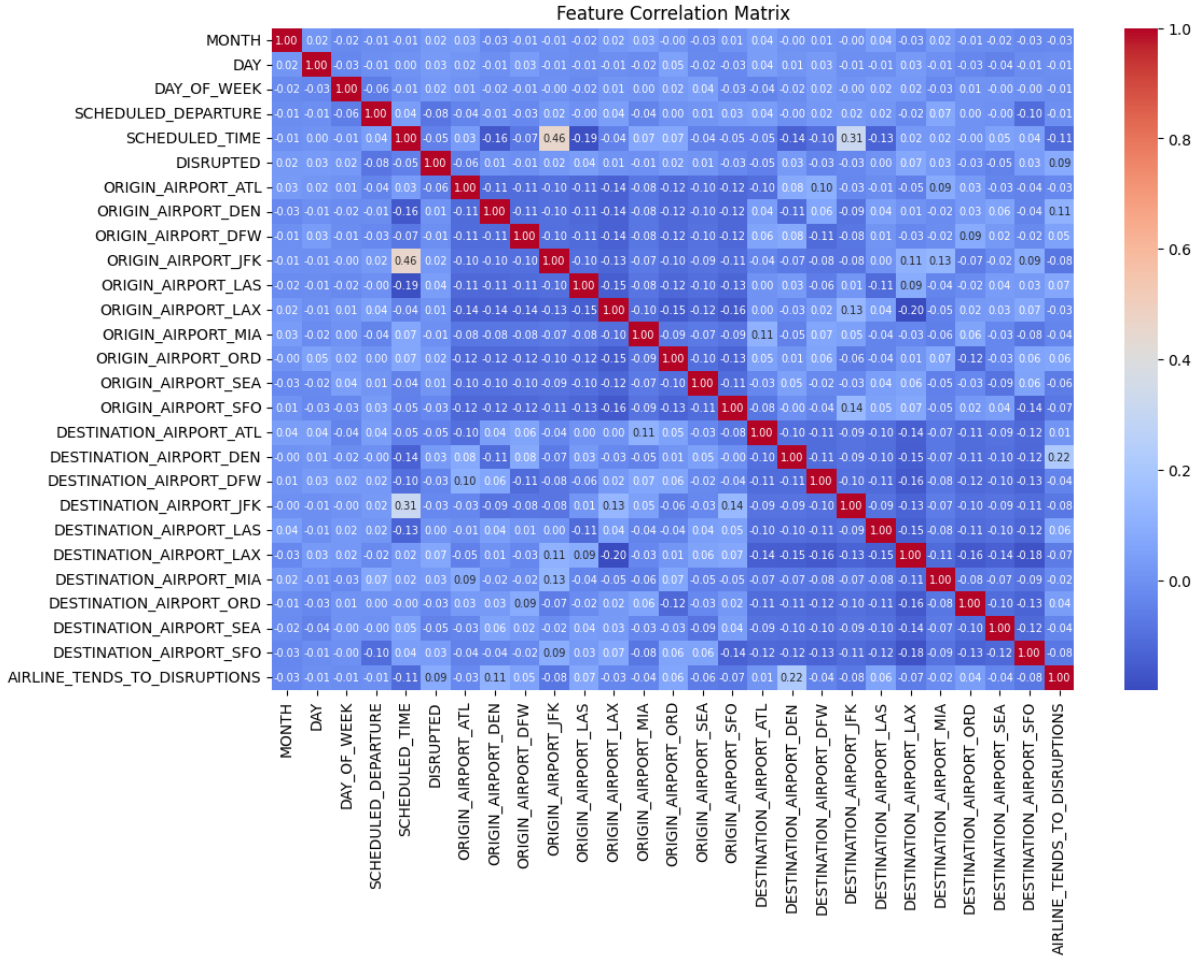
Per a comprovar que no hi ha columnes redundants i, per tant, irrellevants, crearem la matriu de correlació.




D'aquesta matriu destaquen els punts vermells de dalt a l'esquerra. Aquesta enorme correlació, de 0.98 és la causa de les columnes DISTANCE i SCHEDULED_DEPARTURE. Com és lògic, com més distància recorri el vol, més trigarà. A més, com la majoria dels vols comercials volen a la mateixa velocitat, aquesta correlació es fa encara més forta.

També existeix una correlació significativa entre SCHEDULED_DEPARTURE i SCHEDULED_ARRIVAL. Això és degut al fet que com més tard surt avió més tard arriba al seu destí.
He decidit eliminar les columnes DISTANCE i SCHEDULED ARRIVAL, ja que amb les columnes SCHEDULED_DEPARTURE i SCHEDULED_TIME ja es poden deduir les dues anteriors.

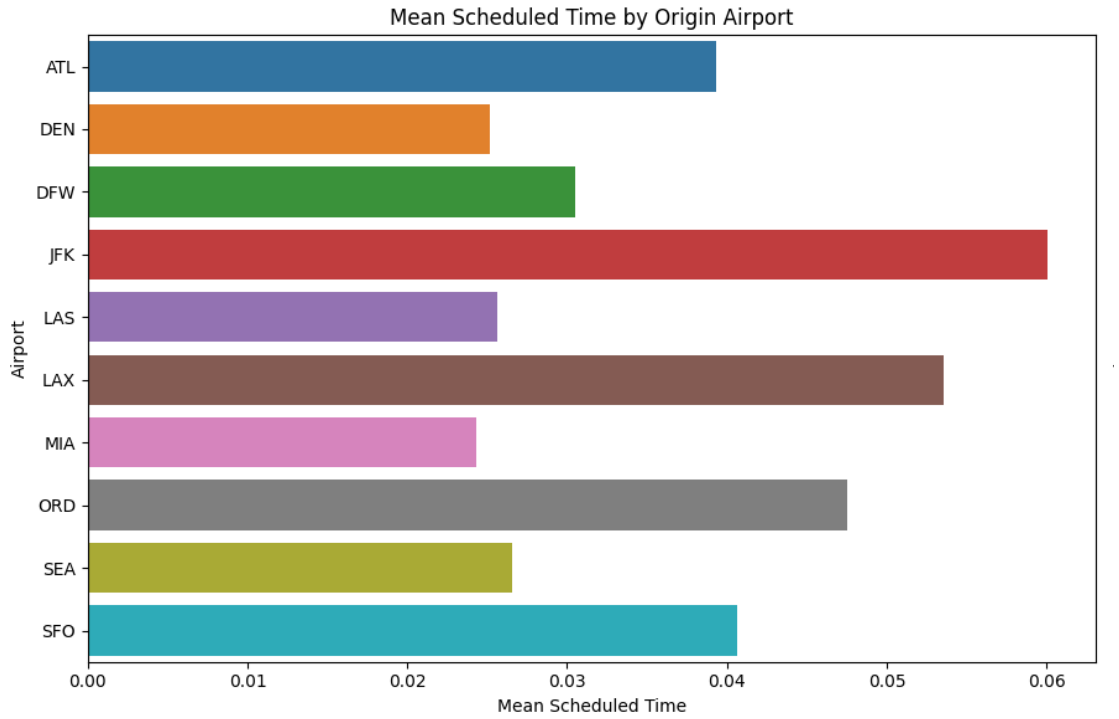
Observem com queda la matriu de correlació després d'eliminar aquestes columnes:



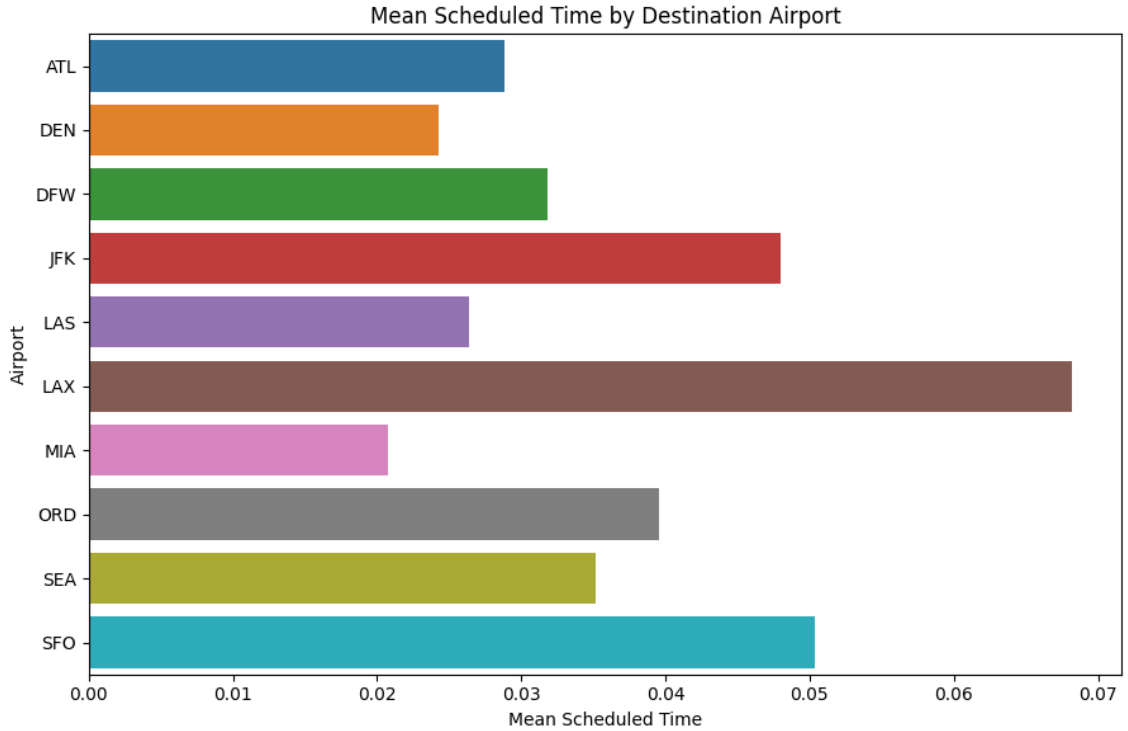

Ara podem observar una correlació més constant entre totes les variables, tot i així es poden observar dues correlacions notables, una entre ORIGIN_AIRPORT_JFK i SCHEDULED_TIME, i una altra entre DESTINATION_AIRPORT_JFK i SCHEDULED_TIME.

Aquestes variables no tenen per què estar correlacionades, inicialment la hipòtesi que ens pot sorgir és que els vols que van o venen de Nova York (Aeroport JFK) acostumen a tenir una durada major. És per això que quan el SCHEDULED_TIME augmenta, també "ho fa" ORIGIN_AIRPORT_JFK i DESTINATION_AIRPORT_JFK, ja que aquests estan a 1 quan el vol té relació amb algun d'aquests aeroports. Així i tot, aalitzarem si aquesta hipòtesi és correcta.

Anem a mirar les mitjanes de duració dels vols pels diferents vols d'origen i destí.



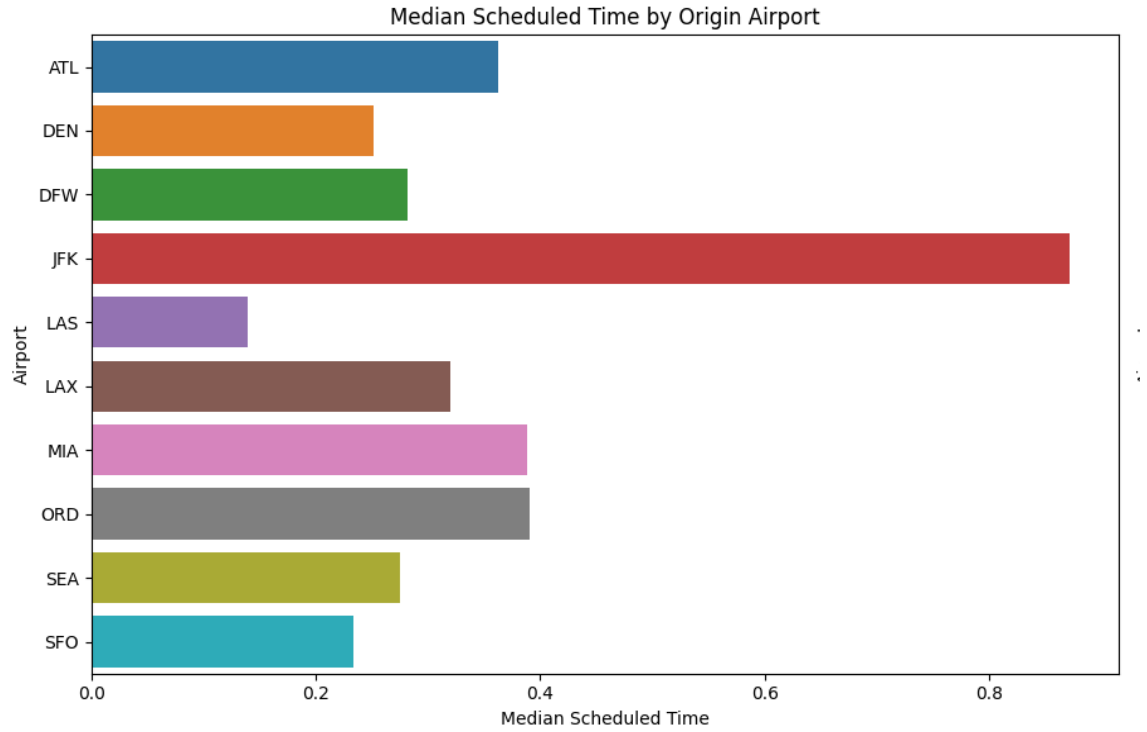

Com es pot observar en aquest gràfic de la mitjana de la duració dels vols per aeroport d'origen, l'aeroport JFK és el que de mitjana té els vols més llargs.




Si observem el mateix gràfic però pels aeroports de destinació, ens trobem amb que LAX (Aeroport de Los Angeles) és el que de mitjana té els vols més llargs.

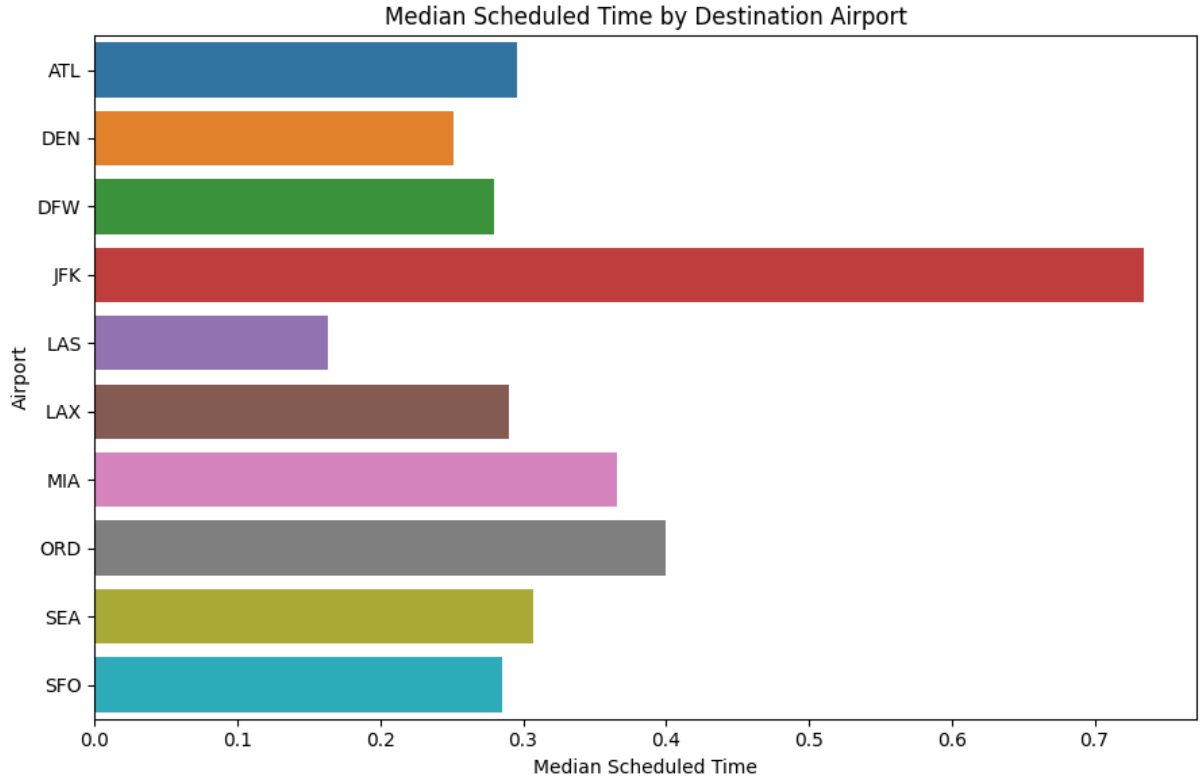
Això no es correspòn amb la matriu de correlació, ja que la correlació entre DESTINATION_AIRPORT_LAX i SCHEDULED_TIME és de 0.02, indicant que no existeix cap relació entre aquests.

Aquest comportament pot ser degut a valors individuals molt alts que fan que la mitjana de la durada dels vols amb destí LAX sigui més gran. Anem a comprovar-ho analitzant les medianes.




Per als aeroports d'origen podem observar com clarament JFK és el que té la mediana més gran, com ja haviem comprovat amb la mitjana es demostra la correlació entre SCHEDULED_TIME i ORIGIN_AIRPORT_JFK.

Ara anem a analitzar els de destí, on LAX tenia la mitjana més alta.




Com podem comprovar es confirma la hipòtesi. LAX tenia pocs valors molt alts que alteraven la mitjana cap adalat. Mirant la mediana de la duració dels vols pels aeroports de destinació podem observar com JFK clarament és el que té la mediana més gran.

Analitzant aquests resultats, podem concloure que la correlació entre JFK i els vols llargs és simplement degut a que per a indicar si un vol té destí JFK es posa la columna DESTINATION_AIRPORT_JFK a 1, el valor màxim després de la normalització, i si arriben vols llargs com en aquest cas a un aeroport o per contra molts vols al final del dia, amb SCHEDULED_DEPARTURE proper a 1, la matriu de correlació indicarà que existeix una correlació entre aquestes dues columnes.

# 3. AVALUACIÓ DEL CRITERI PELS MODELS DE MINERIA DE DADES

## CREACIÓ DE TRAINING I TEST SETS

Després del pre-processing tenim un dataset anomenat flights_clean.csv que ja està preprocessat. Per a separar training i test sets, he utilizat el métode que hem utilitzat al laboratori.

És a dir, he eliminat la target variable (DISRUPTED) del dataset i guardat aquest a la variable X. També he seleccionat el target (DISRUPTED) i guardat a la variable y.

Després he utilitzat la funció train_test_split amb els següents paràmetres:

test_size=0.2: He seleccionat un 80% pel training set i un 20% pel test set.

random_state=42: Per a que es pugui replicar amb el mateix resultat.

stratify = y: He utilitzat stratify per a que es mantingui la proporció de vols DISRUPTED tant al training com al test set.

Finalment he obtingut el següent resultat:

```
X_train shape: (1600, 26)
X_test shape: (400, 26)
y_train shape: (1600,)
y_test shape: (400,)
```

Podem veure com s'han creat correctament els sets d'entrenament i de tests, amb una columna menys (26 en comptes de 27), ja que s'ha eliminat el target i s'ha posat en els sets de y.

## PARÀMETRES D'AVALUACIÓ

Durant el procés d'entrenament de cada model he utilitzat Cross-validation. He considerat tres tipus d'entrenaments, lleugers mitjans i demandants. Per a cadascun he assignat un valor de cv que he escollit basant-me en el compromís entre els recursos del meu ordinador personal i la robustesa dels resultats de cada model.

| Recursos del model | Valor Cross Validation |
|---|---|
| Lleuger | 50 |
| Mitjà | 10 |
| Demandant | 5 |

També he utilitzat K-fold cross validation  amb 10 splits, per a obtenir el millor valor de Cross Validation per a la majoria de models. He escollit aquest nombre de splits ja que crec que és suficient per afirmar una bona robustesa als entrenaments dels models i el meu sistema s'ho pot permetre.

## MÈTRICA D'AVALUACIÓ

Com hem vist al preprocessing, el dataset està desbalancejat, és a dir, hi ha més vols sense disrupcions que vols amb disrupcions. Tot i així també s'ha de considerar la importància de cada classe.

Hem de pensar en casos d'ús del meu model i quin és l'objectiu i els seus usuaris. Un usuari podria ser un empresari que busca prioritzar detectar els vols que seràn disruptius, tot i a costa de crear-ne molts falsos positius, ja que per res del món vol arribar tard a una reunió i prefereix adaptar el seu horari a vols que sàpiga amb certa certesa que tindràn disrupcions.

Un altre podria ser un usuari que busca compensacions de les aerolínies, de pocs recursos possiblement i que, estranyament, no vol haver de pagar per un vol que arribi correctament al destí i no pugui reclamar res a la aerolínea.

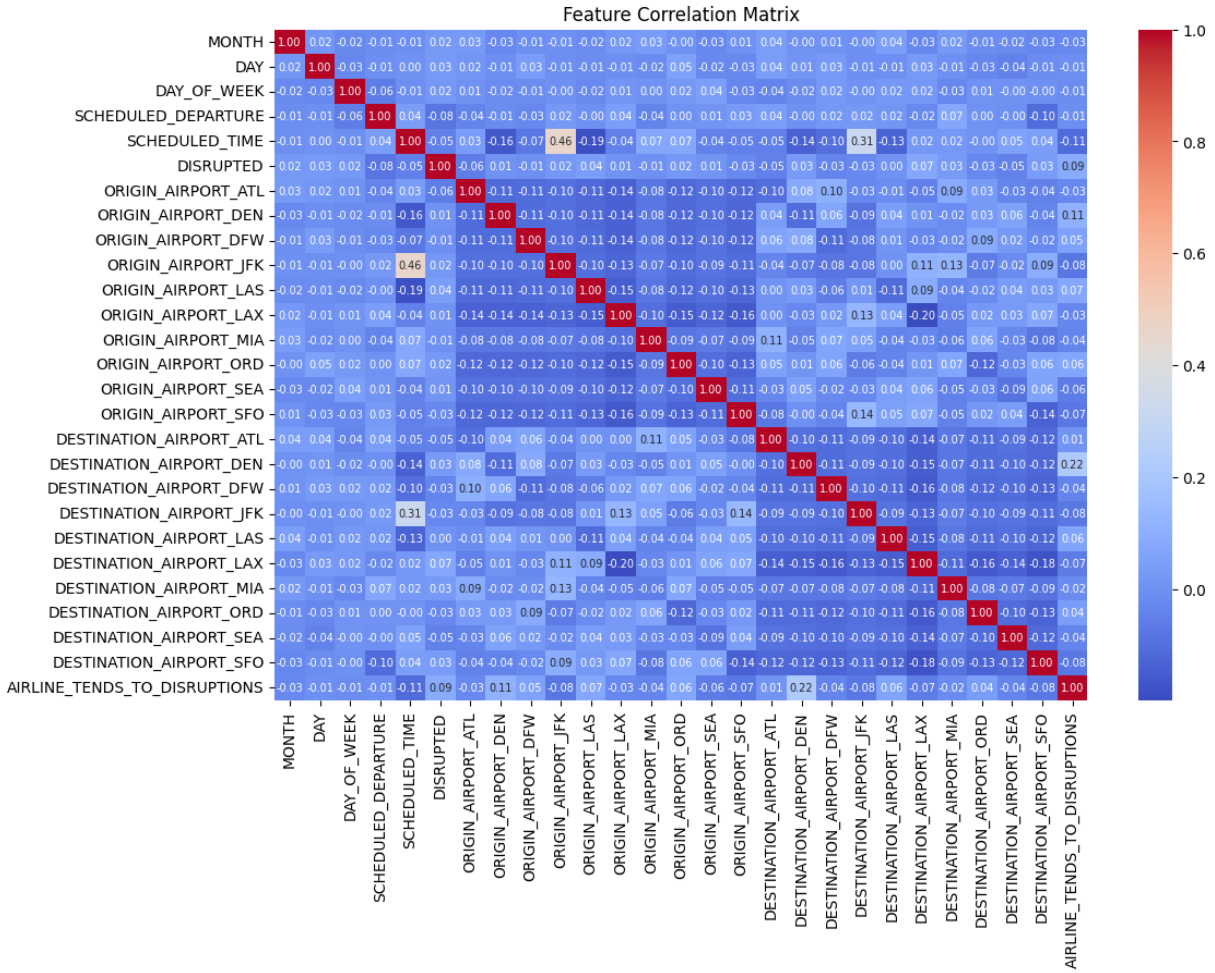
Com no hi ha un cas específic que vulgui prioritzar, volent que sigui un model més general per a la majoria d'usuaris, i el desbalanç no sent molt gran (38% disruptats - 62% no disruptats) he decidit basar-me en la accuracy, anant en compte de no prioritzar un model que tingui molta tendencia a escollir predir molts vols no disruptats per així pujar la accuracy.

És a dir, em bassaré en la accuracy, però també analitzaré el recall i la f1-score per a escollir un model amb alta accuracy però, dins del desbalanç de classes, equilibrat en les seves prediccions.

# 4. EXECUCIÓ DELS DIFERENTS MÈTODES DE MACHINE LEARNING

## NAÏVE BAYES

Com ja he fet al final del preprocessing, podem observar com les columnes són independents entre sí, a excepció del cas que ja he comentat al preprocessing del temps de durada dels viatges i l'aeroport JFK.




Tenim un dataset de 2000 files i 28 columnes, mida suficient per a realitzar Naïve Bayes i obtenir uns resultats vàlids. Apliquem Cross Validation amb un índex de 10, ja que Naïve Bayes no demana molts recursos per a executar-se. Obtenim els següents resultats:

```
Accuracy Score: 0.565
Classification Report:
              precision    recall  f1-score   support

           0       0.65      0.64      0.64      1223
           1       0.44      0.45      0.45       777

    accuracy                           0.56      2000
   macro avg       0.54      0.54      0.54      2000
weighted avg       0.57      0.56      0.57      2000

ROC AUC Score: 0.5441132056013496
```

Tenim una accuracy de 0.565, com veurem més endavant, les accuracies d'aquest dataset no arribaran al 0.7, així que no he interpretat aquest valor baix com que hi hagués algun error en l'aplicació de l'algorisme Naïve Bayes.

Per a la classe 0 (No Disrupció), la precisió és del 65%, el recall és del 64% i la puntuació F1 és del 64%. Per a la classe 1 (Disrupció), la precisió és del 44%, el recall és del 45% i la puntuació F1 és del 45%.

Com es pot observar es detecta amb més precisió els vols que no han tingut cap disrupció, era d'esperar degut al lleuger desbalanç que hi ha al dataset.

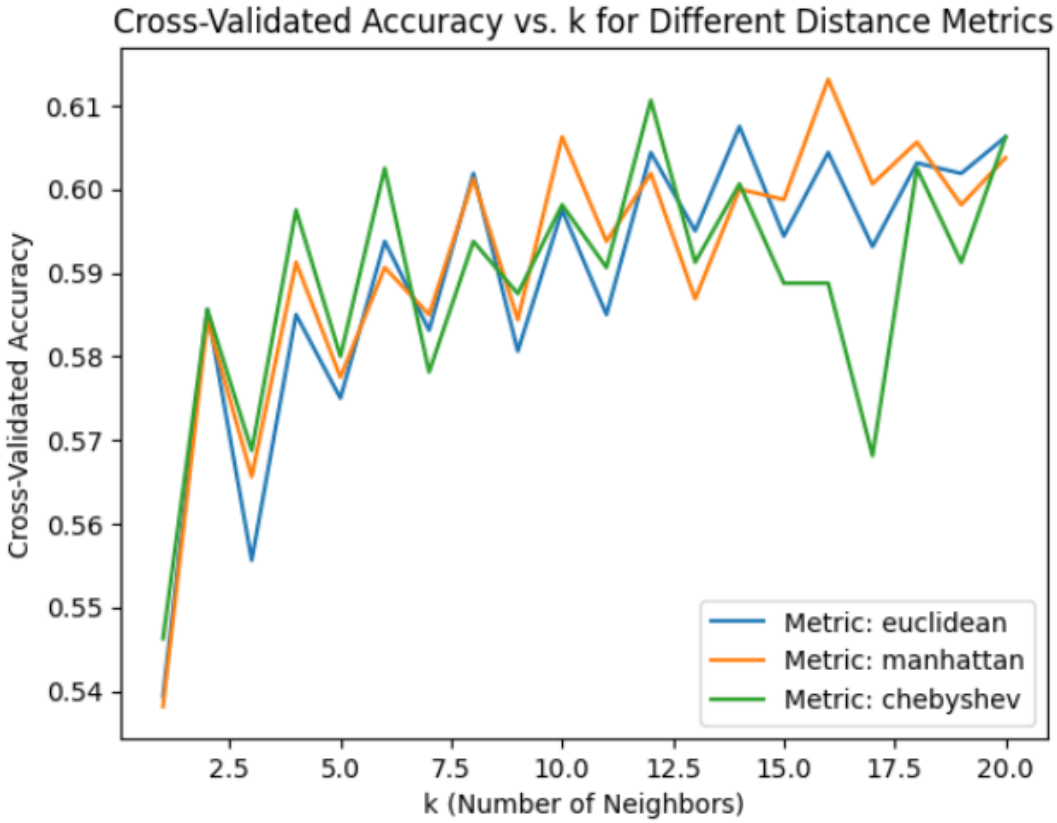
Per a aquest model també he inclòs la AUC (Area Under the Curve). És del 0.544, bastant baixa, però millor que l'aleatorietat.

Són uns resultats d'esperar de Naïve Bayes, són resultats positius, però en ser un algorisme relativament bàsic, no són resultats molt bons.

## KNN

Per a l'algorisme KNN he determinat valors de k de 1 a 40. També he aplicat cross validation (5 folds) i considerat tres mètriques diferents de distància: Manhattan, Euclidiana i Chebyshev. Per a cadascuna de les diferents mètriques he iterat sobre els valors de k per a trobar l'òptim i l'accuracy més alta.

```
Optimal k for euclidean: 14, Max Accuracy: 0.6075
Optimal k for manhattan: 16, Max Accuracy: 0.6131
Optimal k for chebyshev: 12, Max Accuracy: 0.6106
```




Com es pot observar he obtingut els següents resultats:

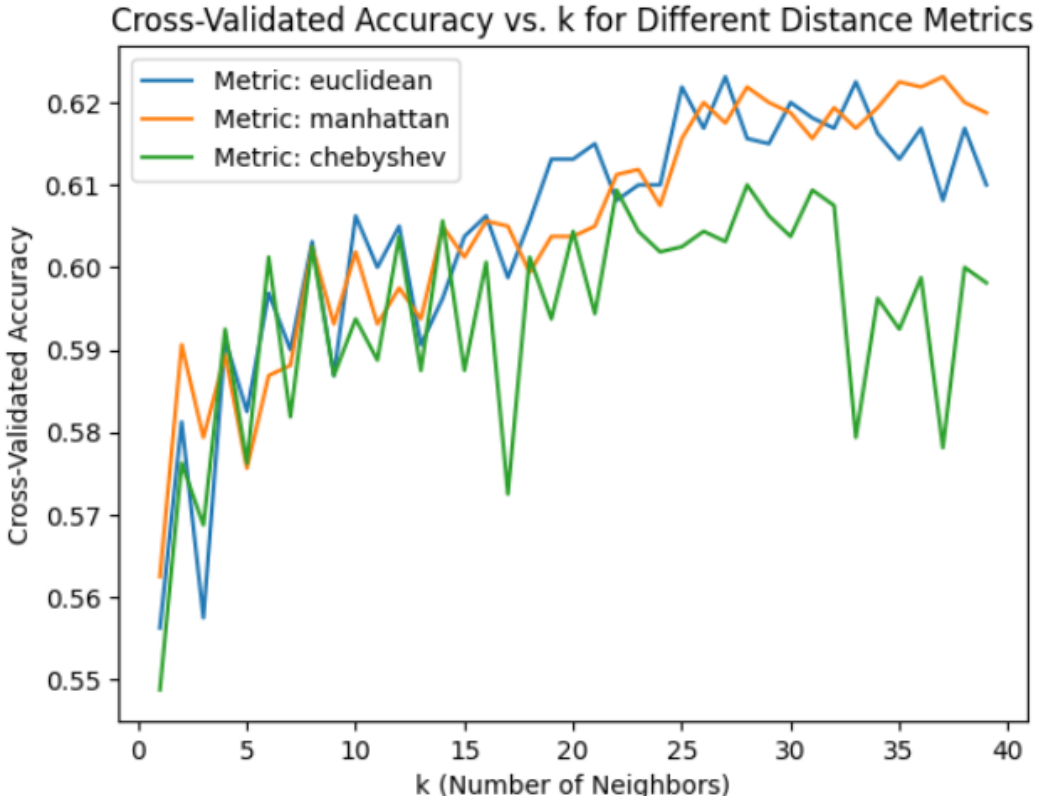
| Tipus de distància | Valor òptim de K | Accuracy |
|---|---|---|
| Euclidiana | 22 | 0.6256 |
| Manhattan | 38 | 0.6269 |
| Chebyshev | 33 | 0.6312 |

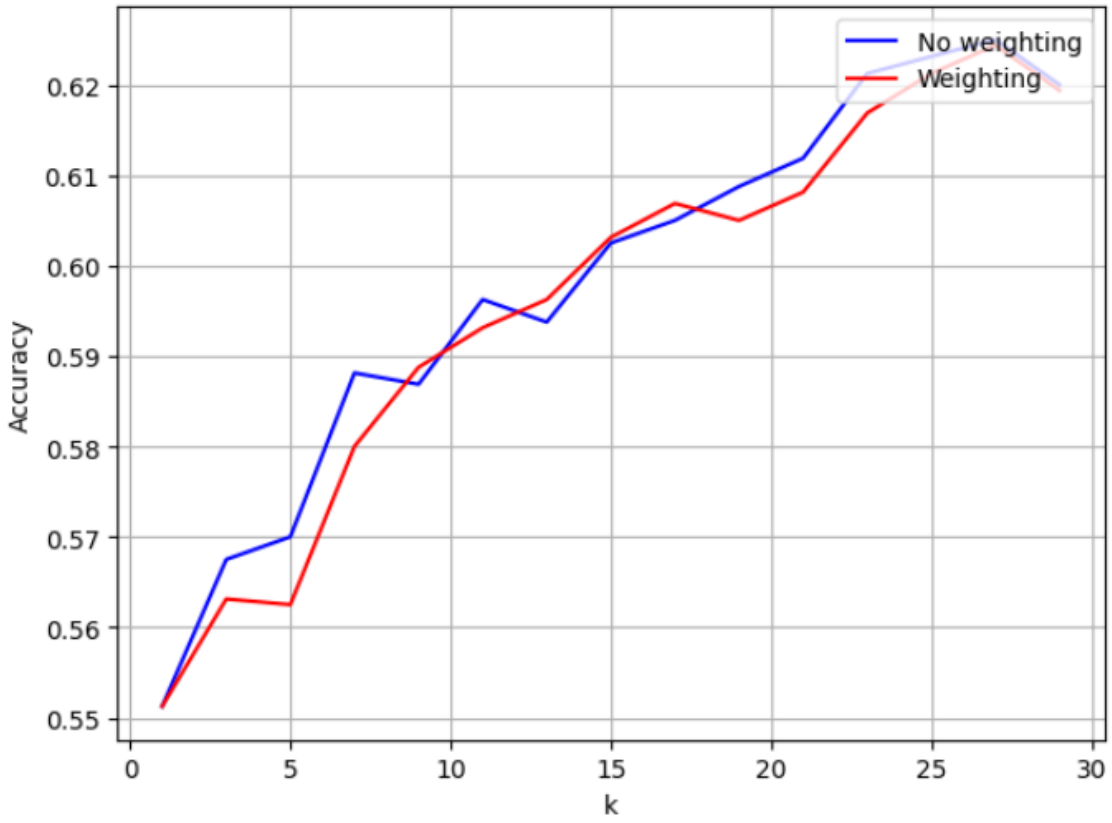
Són resultats bastant bons, però m'agradaria comentar una cosa interessant sobre l'aplicació de KNN amb el meu dataset.

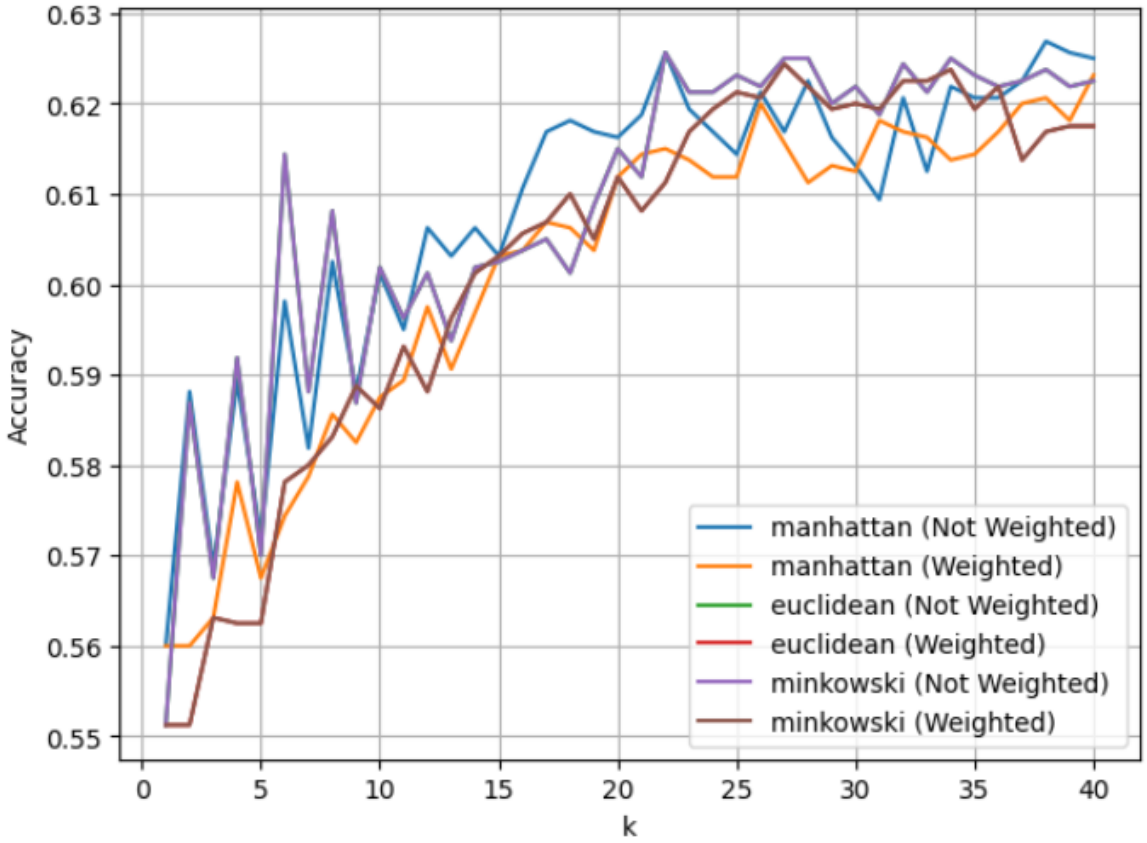
Com vam comentar a classe, a l'algorisme KNN l'afecten les columnes que depenen d'altres i, per tant, són irrellevants. En una fase inicial del projecte no havia avaluat la correlació

entre columnes, al dataset hi havia les columnes de DISTANCE i SCHEDULED_ARRIVAL que s'han eliminat a l'última part del preprocessing degut a la seva redundància amb SCHEDULED_TIME i SCHEDULED_DEPARTURE respectivament.
Amb aquest dataset anterior vaig executar KNN i em va donar el següent resultat:

```
Optimal k for euclidean: 27, Max Accuracy: 0.6231
Optimal k for manhattan: 37, Max Accuracy: 0.6231
Optimal k for chebyshev: 28, Max Accuracy: 0.6100
```




No weighting
Weighting
Accuracy
k
0.55
0.56
0.57
0.58
0.59
0.60
0.61
0.62
0
5
10
15
20
25
30


manhattan (Not Weighted)
manhattan (Weighted)
euclidean (Not Weighted)
euclidean (Weighted)
minkowski (Not Weighted)
minkowski (Weighted)
Accuracy
k
0.55
0.56
0.57
0.58
0.59
0.60
0.61
0.62
0.63
0
5
10
15
20
25
30
35
40

Com es pot observar vaig obtenir els següents resultats:

| Tipus de distància | Valor òptim de K | Accuracy |
|---|---|---|
| Euclidiana | 27 | 0.6231 |
| Manhattan | 37 | 0.6231 |
| Chebyshev | 28 | 0.6100 |

I comparant amb la taula anterior veiem el següent:

| Tipus de distància | Accuracy (Amb columnes irrellevants) | Accuracy (Sense columnes Irrelevants) | Millora d'accuracy |
|---|---|---|---|
| Euclidiana | 0.6231 | 0.6256 | 0.0025 |
| Manhattan | 0.6231 | 0.6269 | 0.0038 |
| Chebyshev | 0.6100 | 0.6312 | 0.0212 |

Com es pot observar, simplement eliminant dues columnes irrellevants (el dataset era el mateix, agafant els mateixos individus), els millors resultats amb les diferents distàncies milloren. M'ha semblat interessant poder experimentar en primera persona el que vam veure a classe i com les columnes irrellevants afecten negativament el resultat de KNN.

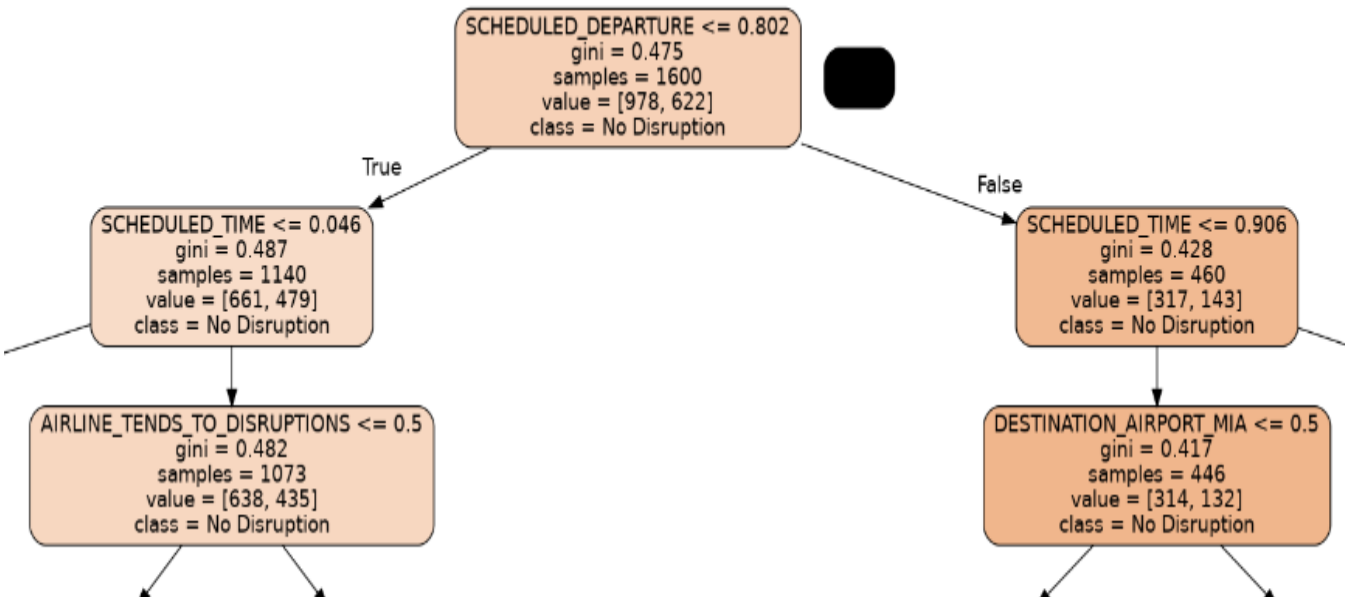
## DECISION TREE

Per al Decision Tree he escollit com a rang per a avaluar les diferents profunditats de 1 a 10. Després de realitzar l'algorisme amb diferents profunditats he obtingut els següents resultats:

```
Optimal max depth: 4, Max Accuracy: 0.6288
Classification Report:
                precision    recall  f1-score   support

No Disruption       0.64      0.92      0.75       245
   Disruption       0.59      0.17      0.27       155

     accuracy                           0.63       400
    macro avg       0.61      0.55      0.51       400
 weighted avg       0.62      0.63      0.57       400
```

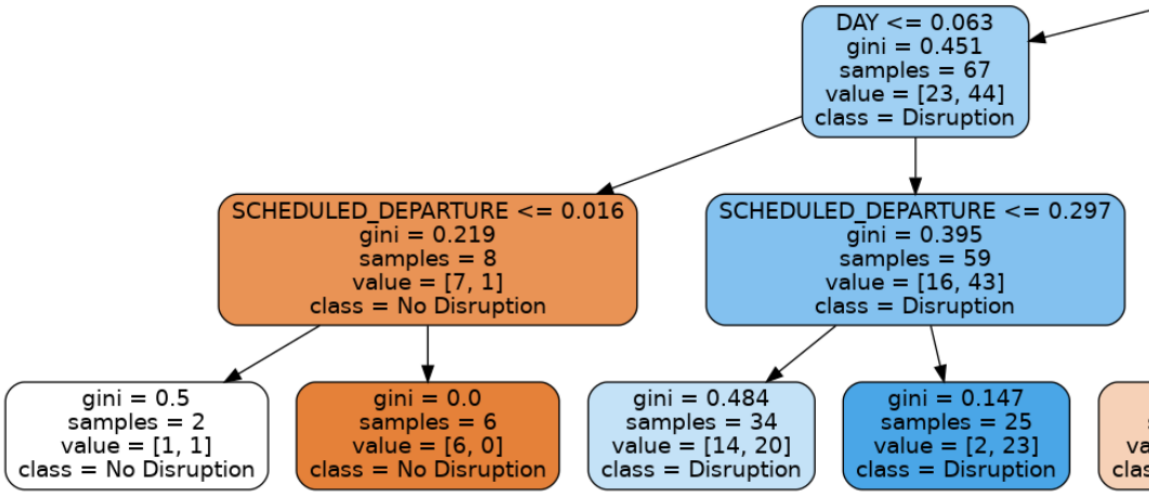
La profunditat òptima ha resultat ser de 4 nivells, amb una accuracy màxima de 0.6288. Com és d'esperar i com els mètodes que hem vist fins ara, prediu millor els vols no disruptats dels que si, pel fet que tenen una major presència al dataset. Ho podem veure en fixar-nos en el recall, sent molt alt per la classe on els vols no tenen disrupció (0.92) i més petit en els disruptats (0.17).

El que més m'agrada de Decision Tree és que tens una manera d'entendre visualment què és el que fa l'algorisme i quines regles han de seguir certs vols per a predir si disruptats o no.




Al principi de l'arbre podem veure com comença discriminant per l'hora de sortida de l'avió, diferenciant entre aquells que surten durant les hores finals del dia i aquells que surten abans.

Ens centrarem en les fulles amb gini = 0, indicant que segur que un vol amb les característiques que portin fins la fulla en qüestió pertanyerà a una classe en concret.



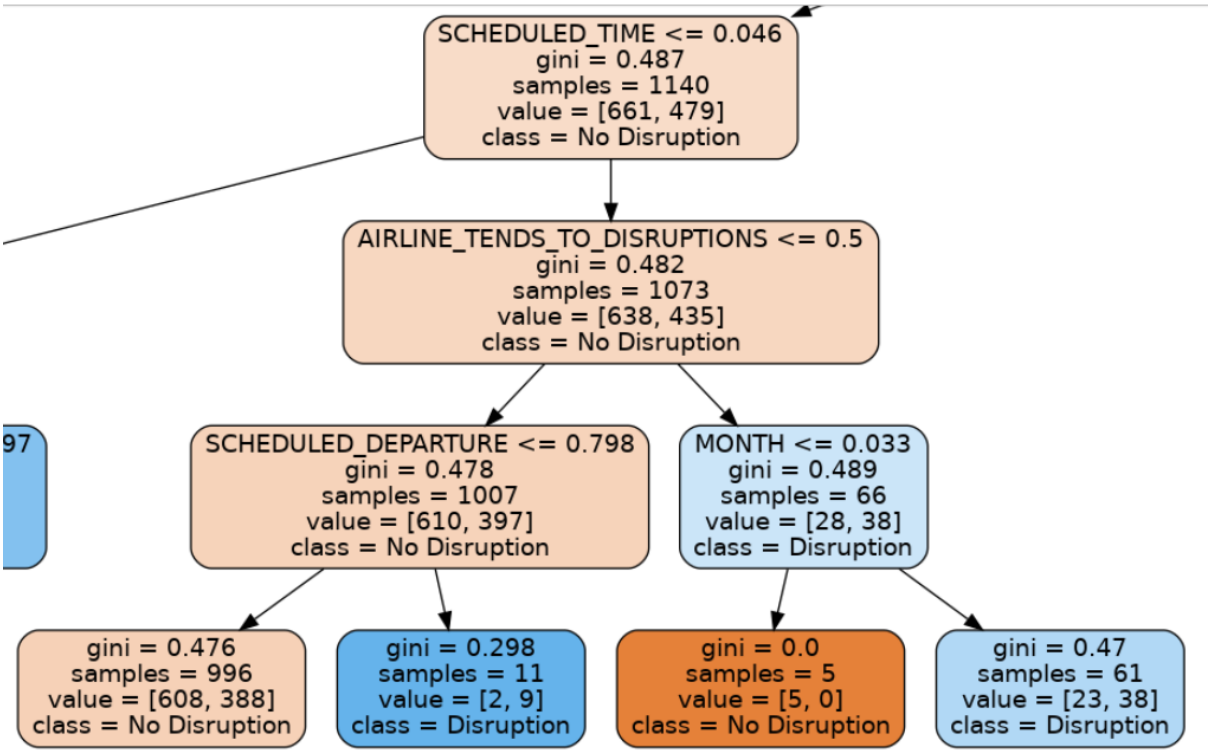

En la part esquerra de l'arbre tenim a una mateixa branca dues fulles amb ginis oposats; una amb 0,5 i l'altre amb 0.

La de 0.5 són aquells vols que no surten a última hora del dia (SCHEDULED_DEPARTURE <= 0.802 ->TRUE ), duren molt poc (SCHEDULED_TIME <= 0.046 -> TRUE), es realitzen a principis de mes (DAY <= 0.063 -> True) i surten a primeríssima hora del dia (SCHEDULED_DEPARTURE <= 0.016 -> True). En aquest cas només hi ha un individu de cada clase, no sent gens clar el resultat i amb pocs elements per a arribar a alguna conclusió prou sòlida.

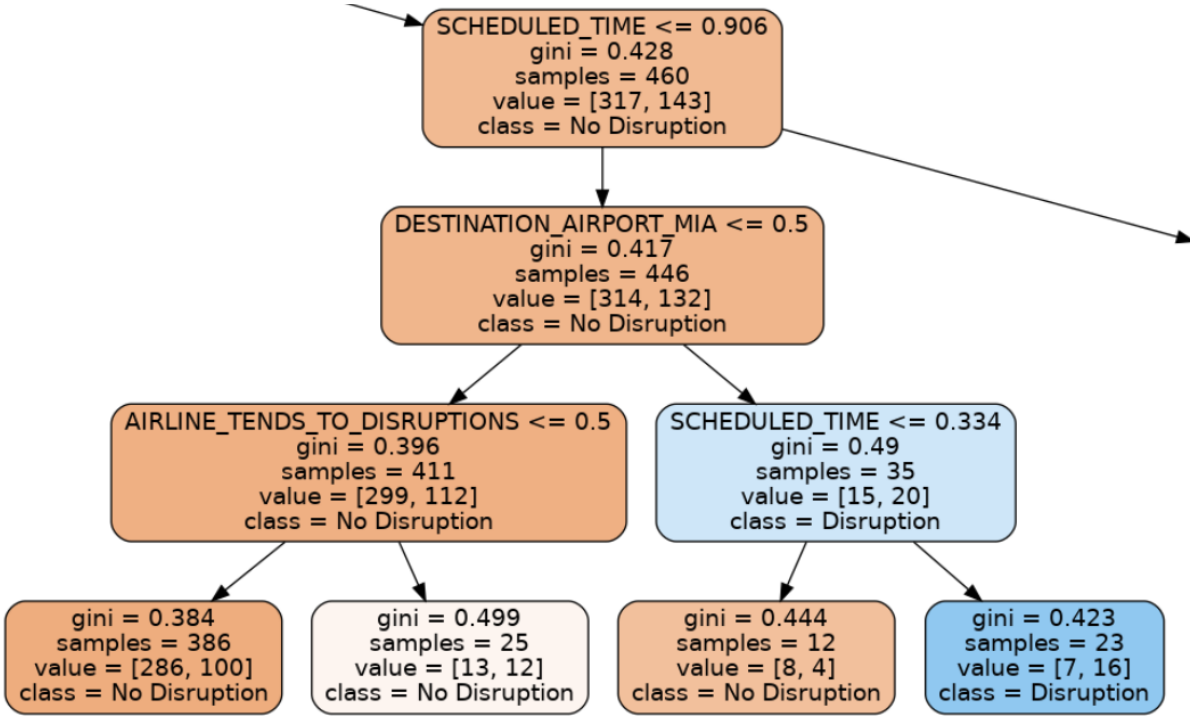
La de 0.0 en canvi, decidint els vols no disruptats, són aquells vols que no surten a última hora del dia (SCHEDULED_DEPARTURE <= 0.802 ->TRUE), duren molt poc (SCHEDULED_TIME <= 0.046 -> TRUE), es realitzen a principis de mes (DAY <= 0.063 -> True) i no surten a primeríssima hora del dia (SCHEDULED_DEPARTURE <= 0.016 -> False). En aquest cas la gini és 0 i amb 6 individus l'algorisme de Decision Tree pot afirmar que el vol no serà disruptat.

Un altre exemple d'una situació prou clara en aquest cas per afirmar que el vol serà disruptat és el següent tipus de vol: El vol que no surt a última hora del dia (SCHEDULED_DEPARTURE <= 0.802 ->TRUE), dura molt poc (SCHEDULED_TIME <= 0.046 -> TRUE), no es realitza a principis de mes (DAY <= 0.063 -> False) i no surt abans que hagi passat un terç del temps entre que surt el primer i últim avió del dia aproximadament (SCHEDULED_DEPARTURE <= 0.297 -> FALSE).

Aquest, però, és una mica més incert, ja que té una gini = 0.147. Així i tot, podríem estar confiats que el vol segurament seria disruptat, ja que n'hi ha 23 amb aquestes situacions que ho han estat davant dels 2 que no.

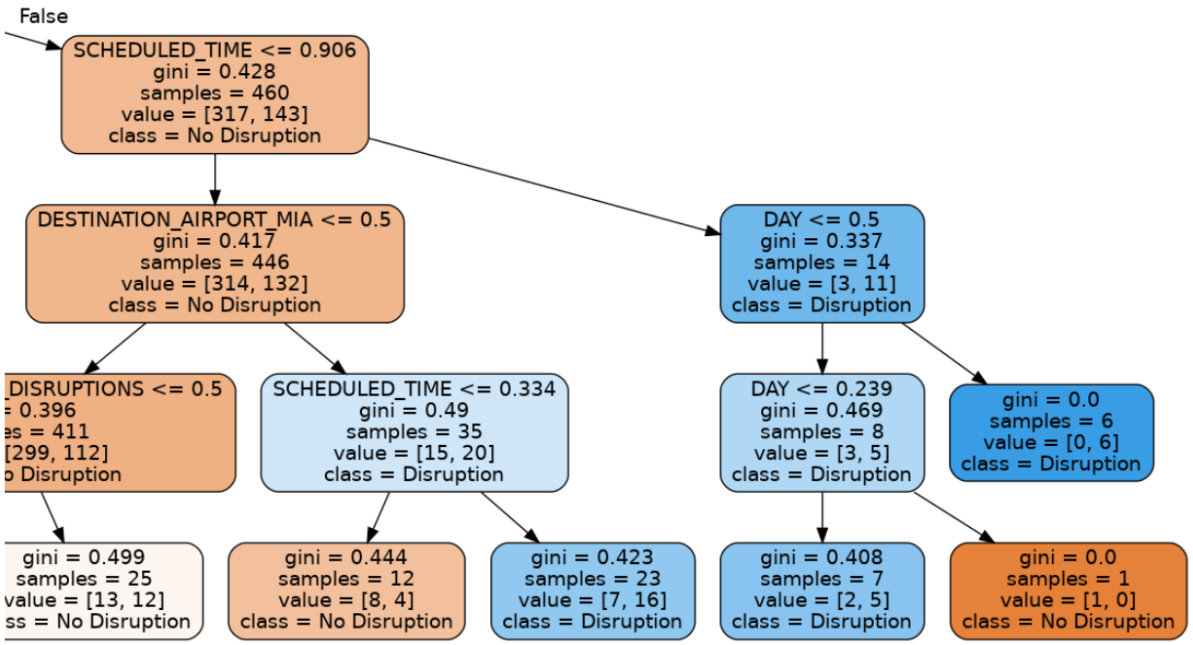



També tenim un altre cas on es podria assegurar que el vol no seria disruptat. El vol hauria de sortir a última hora del dia (SCHEDULED_DEPARTURE <= 0.802 ->TRUE), no durar molt poc (SCHEDULED_TIME <= 0.046 -> FALSE), que l'aerolínia no tingui durant l'any 2015 més vols disruptats que de no disruptats (AIRLINE_TENDS_TO_DISRUPTIONS <= 0.5 -> FALSE) i es facin durant els primers mesos de l'any (MONTH <= 0.033 -> TRUE).



Un exemple curiós és de la fulla on hi ha vols 13 no disruptats i 12 disruptats. És curiós perquè té un nombre relativament alt d'elements per ser una fulla i els valors estan molt parells.

És el cas dels vols que surten a les últimes hores del dia (SCHEDULED_DEPARTURE <= 0.802 -> False), no són d'una durada molt alta (SCHEDULED_TIME <= 0.906 -> TRUE) tenen com a destí l'aeroport de Miami (DESTINATION_AIRPORT_MIA <= 0.5 -> True) i l'aerolínia no ha realitzat més vols disruptats que no (AIRLINE_TENDS_TO_DISRUPTIONS <= 0.5 ->FALSE). En aquest cas hi ha 25 samples dels quals com he comentat 13 són sense disrupció i 12 sí que són disruptius. Degut a aquesta semblança tan gran, la gini és igual a 0.499.




Finalment, tenim una fulla que inicialment pot donar a pensar que està clar que un vol no serà disruptat.

És el cas dels vols que surten a les últimes hores del dia (SCHEDULED_DEPARTURE <= 0.802 -> FALSE), duren molt (SCHEDULED_TIME <= 0.906 -> FALSE), surten durant la primera quinzena del mes (DAY <= 0.5 -> TRUE), i concretament durant la segona setmana del mes aproximadament (DAY <= 0.239 -> FALSE).

El motiu és que té una gini = 0, però només un individu d'un vol no disruptat. És per exemples com aquest que no només ens hem de fixar en el gini, sinó també en el nombre d'elements que hi ha de cada classe, per fer-nos una millor idea de la confiança amb què ho podem dir.

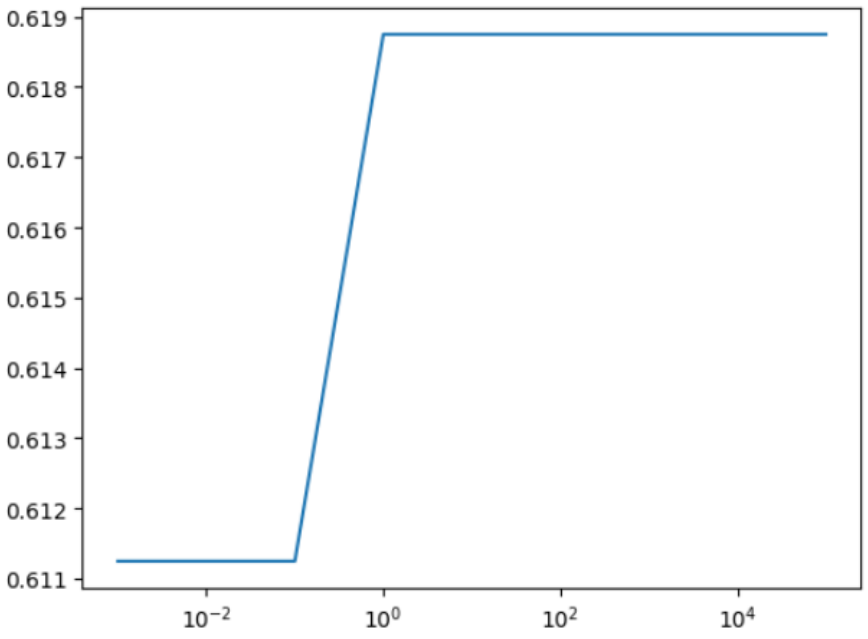
## SUPPORT VECTOR MACHINES (SVM)

### SVM Lineal

Per a una SVM lineal obtenim els seguents resultats:

```
Confusion matrix on test set:
 [[228  17]
 [138  17]]

 Classification Report:
               precision    recall  f1-score   support

            0       0.62      0.93      0.75       245
            1       0.50      0.11      0.18       155

     accuracy                           0.61       400
    macro avg       0.56      0.52      0.46       400
 weighted avg       0.58      0.61      0.53       400
```

Com podem observar aquí el recall és més pronunciat que en l'apartat anterior de Decision Tree, tenint preferència per identificar correctament els vols no disruptats i identificant positivament pocs vols disruptats.




Si analitzem els millors valors per a C podem observar com el millor és C = 1, que és el per defecte i, per tant, el mateix que hem calculat a dalt.

```
Confusion matrix on the test set:
 [[228  17]
 [138  17]]

Accuracy on the test set:  0.6125

Best value of parameter C found:  {'C': 1.0}

Number of supports:  1280 ( 1167 of them have slacks)
Proportion of supports:  0.8
```

Com podem observar hi ha 1280 suports, 1167 amb slacks. Hi ha un gran nombre de suports que s'han deixat estar al costat erroni de la línia recta (en aquest cas estem amb un SVM lineal) per a obtenir els millors resultats per al dataset. Amb la matriu de confusió podem veure que només ha detectat 17 vols disruptats correctament, i 138 els ha marcat com a negatius quan eren positius. El model té una forta inclinació a marcar els vols com a no disruptats, ja que existeix una població més gran d'aquests.

**SVM Polinòmica**

Amb una SVM Polinòmica de grau 2 obtenim els següents resultats:

```
Confusion matrix on test set:
 [[213  32]
 [111  44]]

Accuracy on test set:  0.6425

Classification Report:
              precision    recall  f1-score   support

           0       0.66      0.87      0.75       245
           1       0.58      0.28      0.38       155

    accuracy                           0.64       400
   macro avg       0.62      0.58      0.56       400
weighted avg       0.63      0.64      0.61       400
```

Podem observar com l'accuracy és major, però no només això. Aquest model no tendeix tant a marcar tots els vols com a no disruptats, ja que el recall és menor en la classe 0 i per tant major en la 1, tot i que la classe 0 segueix sent la preferida a l'hora d'escollir un resultat per a un vol.

Si realitzem els models polinòmics per a diferents graus obtenim el següent:

```
Degree 2:


Confusion matrix on test set:
 [[213  32]
 [111  44]]

Accuracy on test set:  0.6425

Classification Report:
              precision    recall  f1-score   support

           0       0.66      0.87      0.75       245
           1       0.58      0.28      0.38       155

    accuracy                           0.64       400
   macro avg       0.62      0.58      0.56       400
weighted avg       0.63      0.64      0.61       400


-----------------------------------

Degree 3:

Confusion matrix on test set:
 [[203  42]
 [101  54]]

Accuracy on test set:  0.6425

Classification Report:
              precision    recall  f1-score   support

           0       0.67      0.83      0.74       245
           1       0.56      0.35      0.43       155

    accuracy                           0.64       400
   macro avg       0.62      0.59      0.58       400
weighted avg       0.63      0.64      0.62       400
-----------------------------------
```

```
Degree 4:

Confusion matrix on test set:
 [[194  51]
 [104  51]]

Accuracy on test set:  0.6125

Classification Report:
              precision    recall  f1-score   support

           0       0.65      0.79      0.71       245
           1       0.50      0.33      0.40       155

    accuracy                           0.61       400
   macro avg       0.58      0.56      0.56       400
weighted avg       0.59      0.61      0.59       400


-------------------------------------

Degree 5:

Confusion matrix on test set:
 [[188  57]
 [105  50]]

Accuracy on test set:  0.595

Classification Report:
              precision    recall  f1-score   support

           0       0.64      0.77      0.70       245
           1       0.47      0.32      0.38       155

    accuracy                           0.59       400
   macro avg       0.55      0.54      0.54       400
weighted avg       0.57      0.59      0.58       400


-------------------------------------
```

```
Degree 6:

Confusion matrix on test set:
 [[187  58]
 [103  52]]

Accuracy on test set:  0.5975

Classification Report:
              precision    recall  f1-score   support

           0       0.64      0.76      0.70       245
           1       0.47      0.34      0.39       155

    accuracy                           0.60       400
   macro avg       0.56      0.55      0.55       400
weighted avg       0.58      0.60      0.58       400


-----------------------------------

Degree 7:

Confusion matrix on test set:
 [[191  54]
 [101  54]]

Accuracy on test set:  0.6125

Classification Report:
              precision    recall  f1-score   support

           0       0.65      0.78      0.71       245
           1       0.50      0.35      0.41       155

    accuracy                           0.61       400
   macro avg       0.58      0.56      0.56       400
weighted avg       0.59      0.61      0.59       400


-----------------------------------
```

```
Degree 8:

Confusion matrix on test set:
 [[195  50]
 [104  51]]

Accuracy on test set:  0.615

Classification Report:
              precision    recall  f1-score   support

           0       0.65      0.80      0.72       245
           1       0.50      0.33      0.40       155

    accuracy                           0.61       400
   macro avg       0.58      0.56      0.56       400
weighted avg       0.60      0.61      0.59       400


-------------------------------------

Degree 9:

Confusion matrix on test set:
 [[190  55]
 [104  51]]

Accuracy on test set:  0.6025

Classification Report:
              precision    recall  f1-score   support

           0       0.65      0.78      0.71       245
           1       0.48      0.33      0.39       155

    accuracy                           0.60       400
   macro avg       0.56      0.55      0.55       400
weighted avg       0.58      0.60      0.58       400


-------------------------------------
```

```
Degree 10:

Confusion matrix on test set:
 [[186  59]
 [107  48]]

Accuracy on test set:  0.585

Classification Report:
              precision    recall  f1-score   support

           0       0.63      0.76      0.69       245
           1       0.45      0.31      0.37       155

    accuracy                           0.58       400
   macro avg       0.54      0.53      0.53       400
weighted avg       0.56      0.58      0.57       400


-------------------------------------
Degree 11:

Confusion matrix on test set:
 [[186  59]
 [106  49]]

Accuracy on test set:  0.5875

Classification Report:
              precision    recall  f1-score   support

           0       0.64      0.76      0.69       245
           1       0.45      0.32      0.37       155

    accuracy                           0.59       400
   macro avg       0.55      0.54      0.53       400
weighted avg       0.57      0.59      0.57       400


-------------------------------------
```

```
Degree 12:

Confusion matrix on test set:
 [[182  63]
 [107  48]]

Accuracy on test set:  0.575

Classification Report:
              precision    recall  f1-score   support

           0       0.63      0.74      0.68       245
           1       0.43      0.31      0.36       155

    accuracy                           0.57       400
   macro avg       0.53      0.53      0.52       400
weighted avg       0.55      0.57      0.56       400

-------------------------------------

Degree 13:

Confusion matrix on test set:
 [[185  60]
 [107  48]]

Accuracy on test set:  0.5825

Classification Report:
              precision    recall  f1-score   support

           0       0.63      0.76      0.69       245
           1       0.44      0.31      0.37       155

    accuracy                           0.58       400
   macro avg       0.54      0.53      0.53       400
weighted avg       0.56      0.58      0.56       400


-------------------------------------
```

```
Degree 14:

Confusion matrix on test set:
 [[184  61]
 [107  48]]

Accuracy on test set:  0.58

Classification Report:
              precision    recall  f1-score   support

           0       0.63      0.75      0.69       245
           1       0.44      0.31      0.36       155

    accuracy                           0.58       400
   macro avg       0.54      0.53      0.53       400
weighted avg       0.56      0.58      0.56       400

-------------------------------------

Degree 15:

Confusion matrix on test set:
 [[181  64]
 [103  52]]

Accuracy on test set:  0.5825

Classification Report:
              precision    recall  f1-score   support

           0       0.64      0.74      0.68       245
           1       0.45      0.34      0.38       155

    accuracy                           0.58       400
   macro avg       0.54      0.54      0.53       400
weighted avg       0.56      0.58      0.57       400


-------------------------------------
```

```
Degree 16:

Confusion matrix on test set:
 [[180  65]
 [104  51]]

Accuracy on test set:  0.5775

Classification Report:
              precision    recall  f1-score   support

           0       0.63      0.73      0.68       245
           1       0.44      0.33      0.38       155

    accuracy                           0.58       400
   macro avg       0.54      0.53      0.53       400
weighted avg       0.56      0.58      0.56       400

-------------------------------------
Degree 17:

Confusion matrix on test set:
 [[178  67]
 [105  50]]

Accuracy on test set:  0.57

Classification Report:
              precision    recall  f1-score   support

           0       0.63      0.73      0.67       245
           1       0.43      0.32      0.37       155

    accuracy                           0.57       400
   macro avg       0.53      0.52      0.52       400
weighted avg       0.55      0.57      0.56       400


-------------------------------------
```

```
Degree 18:

Confusion matrix on test set:
 [[174  71]
 [104  51]]

Accuracy on test set:  0.5625

Classification Report:
              precision    recall  f1-score   support

           0       0.63      0.71      0.67       245
           1       0.42      0.33      0.37       155

    accuracy                           0.56       400
   macro avg       0.52      0.52      0.52       400
weighted avg       0.55      0.56      0.55       400


-----------------------------------

Degree 19:

Confusion matrix on test set:
 [[173  72]
 [104  51]]

Accuracy on test set:  0.56

Classification Report:
              precision    recall  f1-score   support

           0       0.62      0.71      0.66       245
           1       0.41      0.33      0.37       155

    accuracy                           0.56       400
   macro avg       0.52      0.52      0.51       400
weighted avg       0.54      0.56      0.55       400


-----------------------------------
```

```
Degree 20:

Confusion matrix on test set:
 [[170  75]
 [104  51]]

Accuracy on test set:  0.5525

Classification Report:
              precision    recall  f1-score   support

           0       0.62      0.69      0.66       245
           1       0.40      0.33      0.36       155

    accuracy                           0.55       400
   macro avg       0.51      0.51      0.51       400
weighted avg       0.54      0.55      0.54       400


-------------------------------------
```

Resumint en una taula tots els resultats trobats la taula pot quedar així:

| Degree | True Negative (TN) | False Negative (FN) | True Positive (TP) | False Positive (FP) | Accuracy | Recall (Class 0) | Recall (Class 1) | F1-Score (Class 0) | F1-Score (Class 1) |
|---|---|---|---|---|---|---|---|---|---|
| 2 | 213 | 32 | 44 | 111 | 0.6425 | 0.87 | 0.28 | 0.75 | 0.38 |
| 3 | 203 | 42 | 54 | 101 | 0.6425 | 0.83 | 0.35 | 0.74 | 0.43 |
| 4 | 194 | 51 | 51 | 104 | 0.6125 | 0.79 | 0.33 | 0.71 | 0.40 |
| 5 | 188 | 57 | 50 | 105 | 0.595 | 0.77 | 0.32 | 0.70 | 0.38 |
| 6 | 187 | 58 | 52 | 103 | 0.5975 | 0.76 | 0.34 | 0.70 | 0.39 |
| 7 | 191 | 54 | 54 | 101 | 0.6125 | 0.78 | 0.35 | 0.71 | 0.41 |
| 8 | 195 | 50 | 51 | 104 | 0.615 | 0.80 | 0.33 | 0.72 | 0.40 |
| 9 | 190 | 55 | 51 | 104 | 0.6025 | 0.78 | 0.33 | 0.71 | 0.39 |
| 10 | 186 | 59 | 48 | 107 | 0.585 | 0.76 | 0.31 | 0.69 | 0.37 |
| 11 | 186 | 59 | 49 | 106 | 0.5875 | 0.76 | 0.32 | 0.69 | 0.37 |
| 12 | 182 | 63 | 48 | 107 | 0.575 | 0.74 | 0.31 | 0.68 | 0.36 |
| 13 | 185 | 60 | 48 | 107 | 0.5825 | 0.76 | 0.31 | 0.69 | 0.37 |
| 14 | 184 | 61 | 48 | 107 | 0.58 | 0.75 | 0.31 | 0.69 | 0.36 |
| 15 | 181 | 64 | 52 | 103 | 0.5825 | 0.74 | 0.34 | 0.68 | 0.38 |
| 16 | 180 | 65 | 51 | 104 | 0.5775 | 0.73 | 0.33 | 0.68 | 0.38 |
| 17 | 178 | 67 | 50 | 105 | 0.57 | 0.73 | 0.32 | 0.67 | 0.37 |
| 18 | 174 | 71 | 51 | 104 | 0.5625 | 0.71 | 0.33 | 0.67 | 0.37 |
| 19 | 173 | 72 | 51 | 104 | 0.56 | 0.71 | 0.33 | 0.66 | 0.37 |
| 20 | 170 | 75 | 51 | 104 | 0.5525 | 0.69 | 0.33 | 0.66 | 0.36 |

D'aquestes dades ara més recollides i visuals amb la taula podem obtenir les següents conclusions:

Com més augmenta els graus i, per tant, la complexitat de la SVM polinòmica, les accuracies acostumen a tendir a la baixa. Així i tot, a partir del grau 10 el recall de la classe 1 tendeix a pujar. Podria ser interessant si pretenem fixar-nos en detectar correctament els vols disruptats, però hi ha graus entre el 2 i el 6 que tenen un millor recall per a la classe 1.

El que sí sembla ser constant és una baixada de la detecció correcta dels casos negatius (TN True Negative) a mesura que augmenta el grau del polinomi.

Com a conclusió, si ens centrem en l'accuracy tant amb grau 2 i 3 obtenim el mateix, concretament 0.6425. Ens fixarem en els detalls dels dos graus.

| Degree | True Negative (TN) | False Negative (FN) | True Positive (TP) | False Positive (FP) | Accuracy | Recall (Class 0) | Recall (Class 1) | F1-Score (Class 0) | F1-Score (Class 1) |
|---|---|---|---|---|---|---|---|---|---|
| 2 | 213 | 32 | 44 | 111 | 0.6425 | 0.87 | 0.28 | 0.75 | 0.38 |
| 3 | 203 | 42 | 54 | 101 | 0.6425 | 0.83 | 0.35 | 0.74 | 0.43 |

El SVM polinòmic de grau 3 detecta pitjor els casos negatius i, en canvi, es fixa una mica més amb els positius comparat amb els models que hem vist fins ara. No obstant això, també es fixa més en els casos on els vols no s'han disruptat com era d'esperar.

El SVM polinòmic de grau 2, en canvi, es centra més en la classe predominant, vols no disruptats (classe 2).

Aquest em sembla un bon exemple per comentar el que vam parlar a classe, que dos models amb accuracies molt similars (o idèntiques com en aquest cas), poden tenir comportaments diferents.

Tot i tenir la mateixa accuracy i ser un grau més complex, escolliré el de grau 3 com millor grau per la SVM polinòmica. Ho faré perquè és un model més general, que és el que busco, que detecti bé la classe 0, però no deixi enrere la 1 tot i ser menys predominant. També, si ens fixem en les f1-score dels dos graus, el de tercer grau baixa un punt en la classe 0 però, en canvi, augmenta 5 en la classe 1.

Pel que fa als millors paràmetres per a C, he intentat executar el codi del laboratori per a executar el GridSearch, però després d'hores d'execució no l'he pogut obtenir. Així i tot al notebook es pot veure que he provat amb una C de 10 i una altra de 0.1. No he obtingut millors resultats que amb la per defecte (C=1) que és amb la que he dut a terme els tests anteriors.

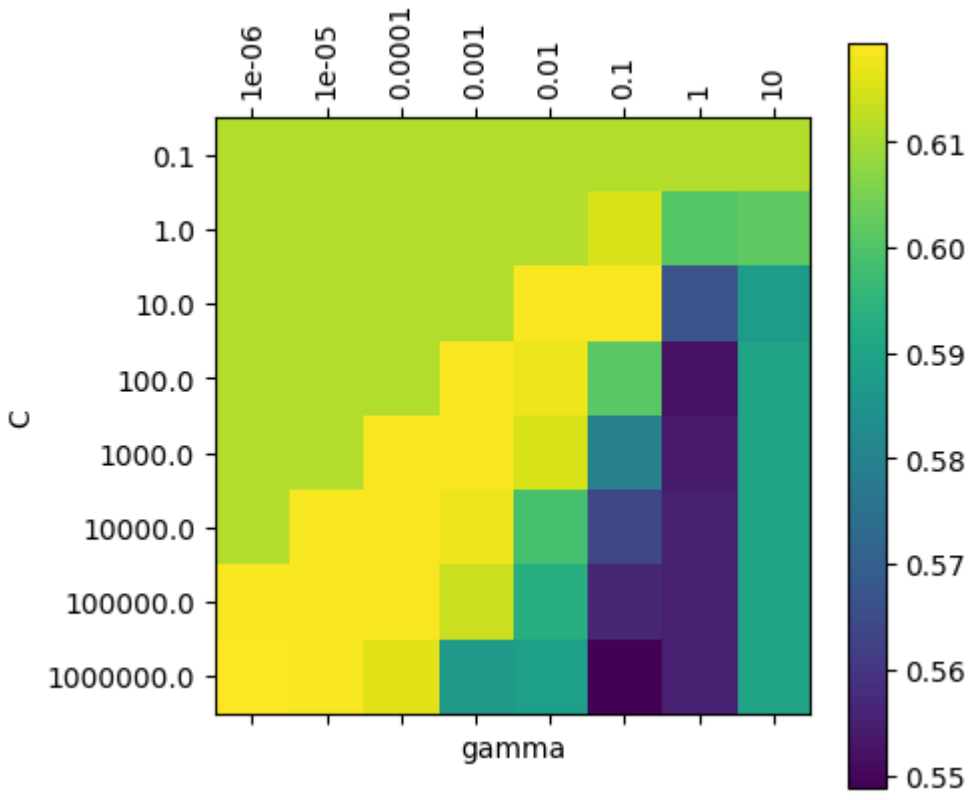
**SVM Radial**

Per a l'execució de la SVM Radial per defecte tenim el següent resultat:

```
Confusion matrix on test set:
 [[214  31]
 [113  42]]

Accuracy on test set:  0.64

Classification Report:
              precision    recall  f1-score   support

           0       0.65      0.87      0.75       245
           1       0.58      0.27      0.37       155

    accuracy                           0.64       400
   macro avg       0.61      0.57      0.56       400
weighted avg       0.62      0.64      0.60       400
```

Amb una accuracy de 0.64 no es separa molt de la polinòmica de tercer i segon grau. Així i tot, el seu comportament s'assembla més al de segon grau, amb un recall superior per als vols no disruptats que pels que si han tingut alguna complicació.

Amb aquest kernel de SVM sí que he pogut calcular el gridsearch, analitzem-lo:




```
Best combination of parameters found:  {'C': 1000000.0, 'gamma': 1e-06}
```

Podem observar una diagonal entre com més petita és la gamma més grans han de ser els valors de C per a obtenir els millors resultats. Els millors valors ens indica que és per una C=1000000 i una gamma de 1e-06 (0.000001).
Analitzem els resultats de la SVM Radial amb els valors de C i gamma òptims segons GridSearch.

```
Confusion matrix on test set:
 [[224  21]
 [134  21]]

Accuracy on test set:  0.6125

Classification Report:
              precision    recall  f1-score   support

           0       0.63      0.91      0.74       245
           1       0.50      0.14      0.21       155

    accuracy                           0.61       400
   macro avg       0.56      0.52      0.48       400
weighted avg       0.58      0.61      0.54       400
```

En aquest cas l'accuracy és menor, de 0.6125 respecte als valors per defecte. Si ens centrem en el recall i la f1-score, podem veure com aquest model té una predisposició major per a la classe 0. Per aquest model no m'interessa tant, ja que no vull que es centri tant en la classe predominant (vols no disruptats).

## META METHODS

### Votació per majoria

Per a la votació sense pesos primerament creem els tres models: KNN, Naive Bayes i Decision Tree. Ens donen els següents resultats:

```
Best Params for Knn = {'n_neighbors': 27, 'weights': 'uniform'} Accuracy = 0.6235
Accuracy: 0.562 [Naive Bayes]
Accuracy: 0.624 [Knn (3)]
Accuracy: 0.576 [Dec. Tree]
Classification Report for Naive Bayes:
                precision    recall  f1-score   support

No Disruption       0.67      0.69      0.68       245
   Disruption       0.48      0.46      0.47       155

     accuracy                           0.60       400
    macro avg       0.58      0.57      0.57       400
 weighted avg       0.60      0.60      0.60       400

Classification Report for Knn (3):
                precision    recall  f1-score   support

No Disruption       0.64      0.84      0.72       245
   Disruption       0.49      0.25      0.33       155

     accuracy                           0.61       400
    macro avg       0.57      0.54      0.53       400
 weighted avg       0.58      0.61      0.57       400

Classification Report for Dec. Tree:
                precision    recall  f1-score   support

No Disruption       0.65      0.67      0.66       245
   Disruption       0.46      0.44      0.45       155

     accuracy                           0.58       400
    macro avg       0.56      0.56      0.56       400
 weighted avg       0.58      0.58      0.58       400
```

Creem els tres models que participaran a la votació. En aquest cas decision Tree i Naive Bayes obtenen uns resultats similars, mentre que el que té un comportament diferent és KNN, tendint més cap a la classe predominant.

Si utilitzem VotingClassClassifier amb el voting a hard, és a dir, indicant que serà votació per majoria i amb un cross-validation de 50, obtenim el següent resultat.

```
Accuracy: 0.608 [Majority Voting]
```

Seguint les instruccions que vam treballar al laboratori, obtenim una accuracy del 0.608. Aquesta accuracy és major que la del Decision Tree i Naive Bayes per individual, però inferior a la de KNN per solitari.

**Votació per pesos**

Per a executar la votació per pesos i obtenir els pesos òptims, he iterat pels 3 models assignant models del 1 al 3, realitzant així cada combinació possible per a trobar els millors pesos.

```
Best Weights: (3, 3, 2) with Accuracy: 0.6150000000000001
```

La millor combinació és assignar un pes de 3 a les votacions de Naive Bayes i KNN, mentre assignar un valor inferior al vot de Decision Tree. Amb aquest mètode obtenim una accuracy de 0.615, major al vot per majoria i únicament una lleugerament inferior al de KNN per solitari, mentre que millora molt l'accuracy de Naive Bayes i Decision Tree individualment.

**Bagging**

Seguint els passos que vam fer al laboratori i aplicant com a estimador el classificador de Decision Tree amb un cross validation de 50, obtenim els següents resultats:

```
Accuracy: 0.564 [1]
Accuracy: 0.586 [2]
Accuracy: 0.582 [5]
Accuracy: 0.607 [10]
Accuracy: 0.614 [20]
Accuracy: 0.619 [50]
Accuracy: 0.628 [100]
Accuracy: 0.629 [200]

Accuracy: 0.570 [1]
Accuracy: 0.574 [2]
Accuracy: 0.588 [5]
Accuracy: 0.589 [10]
Accuracy: 0.597 [20]
Accuracy: 0.613 [50]
Accuracy: 0.624 [100]
Accuracy: 0.627 [200]
```

Com podem observar a mesura que augmentant el nombre d'estimadors l'accuracy es va incrementant, tot i que cada vegada ho fa de manera més lenta. Obtenim relativament bones accuracies, però cap a destacar.

**Random Forest**

Ara comentarem els resultats similars però amb RandomForests. Amb el Random Forest

```
Accuracy: 0.546 [1]
Accuracy: 0.589 [2]
Accuracy: 0.574 [5]
Accuracy: 0.609 [10]
Accuracy: 0.609 [20]
Accuracy: 0.621 [50]
Accuracy: 0.630 [100]
Accuracy: 0.621 [200]
```

Classifier obtenim els següents resultats:
I aplicant el classificador de ExtraTreesClassifier obtenim els següents resultats:

```
Accuracy: 0.565 [1]
Accuracy: 0.593 [2]
Accuracy: 0.591 [5]
Accuracy: 0.589 [10]
Accuracy: 0.607 [20]
Accuracy: 0.593 [50]
Accuracy: 0.602 [100]
Accuracy: 0.604 [200]
```

Com és normal obtenim valors cada vegada millors, però podem observar com aquest també redueixen en la seva millora respecte al nombre anterior. També podem veure com els resultats amb el Random Forest Classifier tenen una accuracy més alta que els d'Extra Trees Classifier. També hi ha resultats independents que no són millors que l'anterior.

**Boosting**

Ara repetirem el mateix però amb aplicant Boosting. Els classificadors amb els seus resultats són els següents:

**AdaBoostClassifier**

```
Accuracy: 0.605 [1]
Accuracy: 0.619 [2]
Accuracy: 0.624 [5]
Accuracy: 0.623 [10]
Accuracy: 0.620 [20]
Accuracy: 0.614 [50]
Accuracy: 0.612 [100]
Accuracy: 0.610 [200]
```

**AdaBoostClassifier amb DecisionTreeClassifier**

```
Accuracy: 0.611 [1]
Accuracy: 0.608 [2]
Accuracy: 0.597 [5]
Accuracy: 0.604 [10]
Accuracy: 0.573 [20]
Accuracy: 0.581 [50]
Accuracy: 0.583 [100]
Accuracy: 0.589 [200]
```

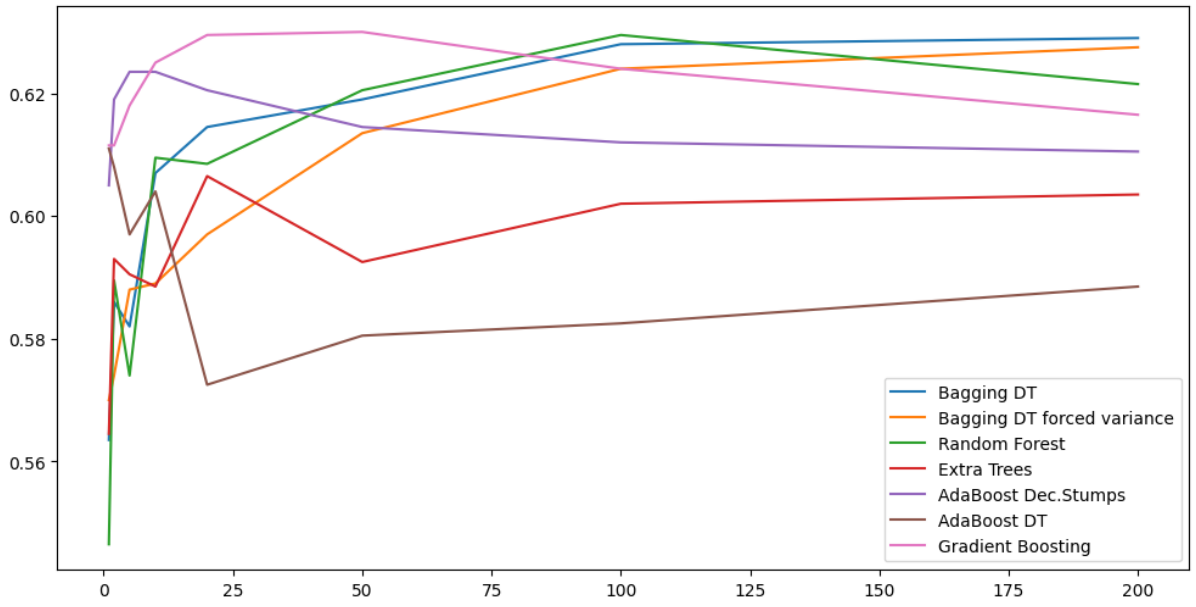
**Gradient Boosting Classifier**

```
Accuracy: 0.612 [1]
Accuracy: 0.612 [2]
Accuracy: 0.618 [5]
Accuracy: 0.625 [10]
Accuracy: 0.630 [20]
Accuracy: 0.630 [50]
Accuracy: 0.624 [100]
Accuracy: 0.616 [200]
```

Com podem observar aquests classificadors de Boosting deixen de seguir la tendència dels anteriors, deixant de tenir més accuracy com més gran és el nombre d'estimadors.

**Comparació dels classificadors**

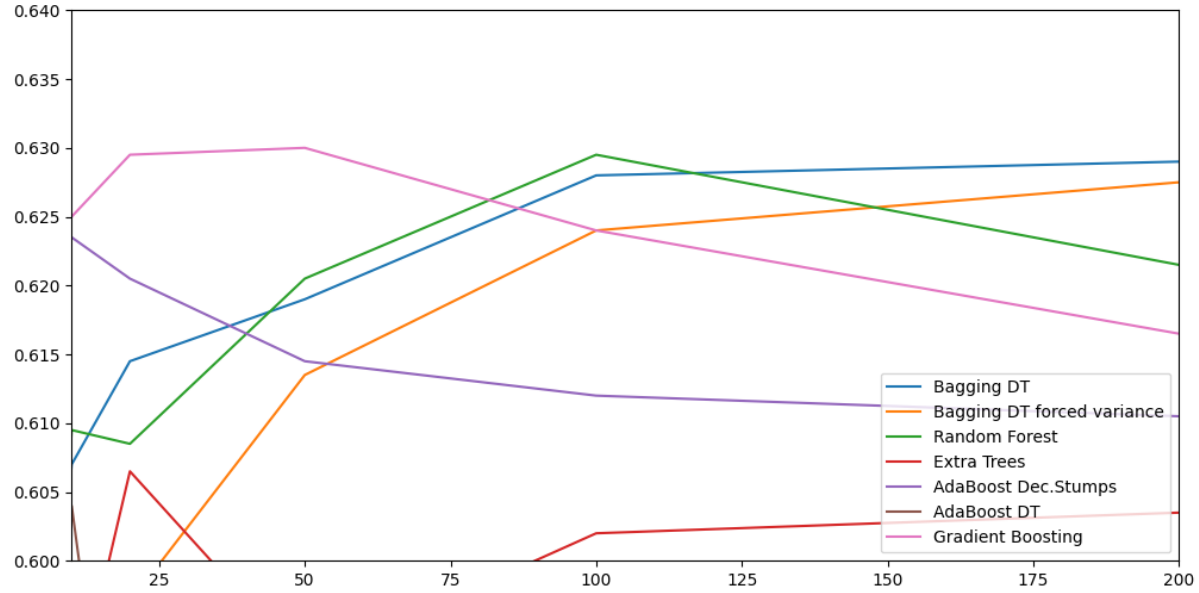
Una vegada creats tots, els compararem amb un gràfic per a fer-ho més visual.




Com podem observar a fins als 50 estimadors aproximadament, tots segueixen diferents comportaments. A partir del 50 estimadors tots comencen a estabilitzar-se i seguir un comportament estable, els primers que hem analitzat, com hem comentat en aquesta

secció, acostumen a incrementar a un ritme més lent, mentre que els de Boosting tendeixen a la baixa.

Ens fixarem en la part superior del gràfic.




En aquest gràfic focalitzat a la part amb accuracies més altes, podem veure com el que millor rendeix és el Gradient Boosting per a 20 i 50 features (Accuracy = 0.63) juntament amb Random Forest per a 100 features (Accuracy = 0.63).

**Random Forest amb feature selection**

Primerament mostrem les importàncies de les columnes juntament amb el seu nom:

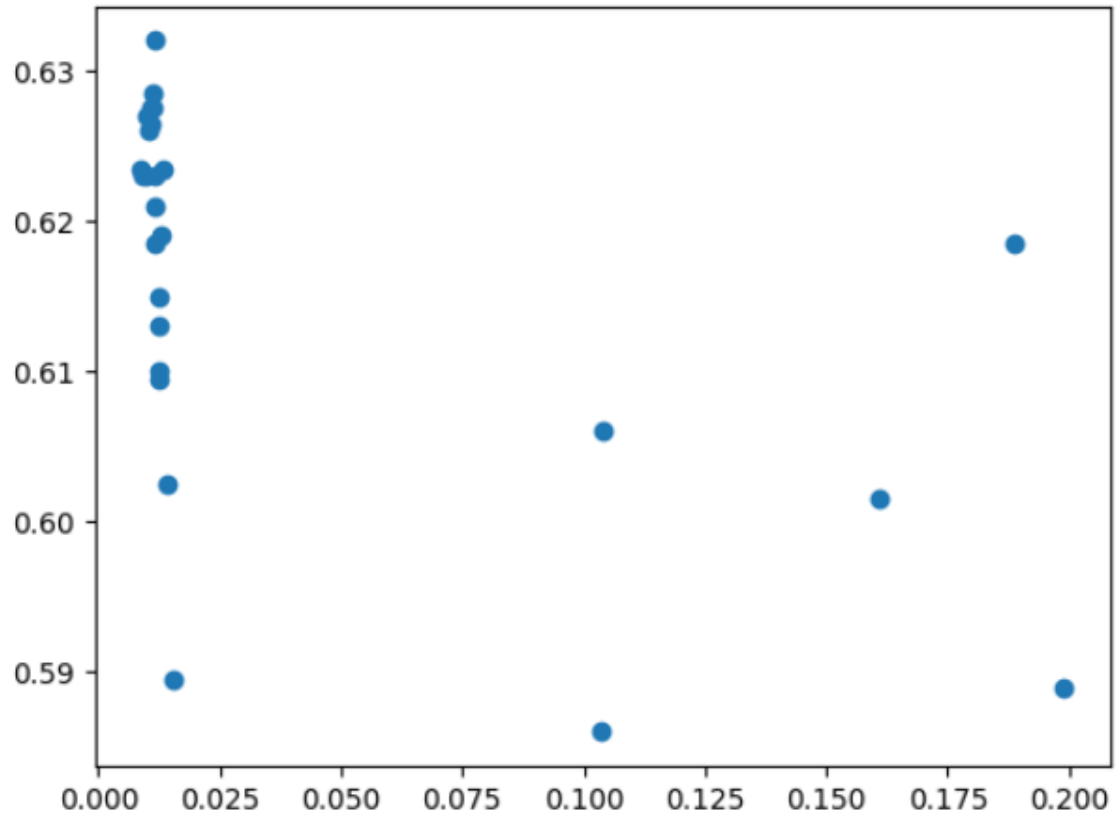
```
[0.10390358 0.16095448 0.10363059 0.19889265 0.18870057 0.01141529
 0.01225421 0.0126302  0.01027489 0.01162322 0.01548555 0.00914593
 0.012584   0.01266476 0.01262885 0.00970932 0.01179226 0.01071874
 0.00871104 0.01120035 0.01403084 0.01008417 0.01119654 0.01070066
 0.01170837 0.01335893]
Index(['MONTH', 'DAY', 'DAY_OF_WEEK', 'SCHEDULED_DEPARTURE', 'SCHEDULED_TIME',
       'ORIGIN_AIRPORT_ATL', 'ORIGIN_AIRPORT_DEN', 'ORIGIN_AIRPORT_DFW',
       'ORIGIN_AIRPORT_JFK', 'ORIGIN_AIRPORT_LAS', 'ORIGIN_AIRPORT_LAX',
       'ORIGIN_AIRPORT_MIA', 'ORIGIN_AIRPORT_ORD', 'ORIGIN_AIRPORT_SEA',
       'ORIGIN_AIRPORT_SFO', 'DESTINATION_AIRPORT_ATL',
       'DESTINATION_AIRPORT_DEN', 'DESTINATION_AIRPORT_DFW',
       'DESTINATION_AIRPORT_JFK', 'DESTINATION_AIRPORT_LAS',
       'DESTINATION_AIRPORT_LAX', 'DESTINATION_AIRPORT_MIA',
       'DESTINATION_AIRPORT_ORD', 'DESTINATION_AIRPORT_SEA',
       'DESTINATION_AIRPORT_SFO', 'AIRLINE_TENDS_TO_DISRUPTIONS'],
      dtype='object')
```

Filtrem els features només amb una importància major a 0.05 i eliminem la resta del nou dataset. Ens quedem amb 5 columnes només.

```
(2000, 5)
0.5855
0.5414999999999999
```

Com podem observar els resultats per a KNN són pitjors per al nou dataset de 5 features si el comparem amb el que té totes les columnes.

A continuació trobem el valor de threshold òptim. Obtenim el següent:




```
Best threshold:  0.011415292104470233
```

Seguint el que vam fer al laboratori obtenim el nou valor òptim de threshold, 0.011415 aproximadament.

Ara tornem a seleccionar els features que estiguin per sobre del threshold d'importància, però en comptes de seleccionar un arbitrari com abans (0.05), assignarem el valor òptim que hem trobat (0.011415 ).

```
(2000, 17)
Original:  0.5579999999999999
With FS:  0.632
```

En aquest cas ens quedem amb 17 columnes, és a dir, la eliminació de features no és tan gran com en el cas del threshold arbitrari. Podem observar com es produeix una millora molt gran simplement seleccionant el threshold òptim, el model passa d'una accuracy de 0.557999 a 0.632 amb el Feature Selection amb el valor òptim de Threshold.

# 5. COMPARACIÓ I CONCLUSIONS

Resumint l'apartat anterior, podem extreure la següent informació dels models en una taula:

| Model | Accuracy | Recall (Class 0) | Recall (Class 1) | F1-Score (Class 0) | F1-Score (Class 1) |
|---|---|---|---|---|---|
| **Naive Bayes** | 0.565 | 0.64 | 0.45 | 0.64 | 0.45 |
| **SVM Linear** | 0.6125 | 0.93 | 0.11 | 0.75 | 0.18 |
| **SVM Poly 2** | 0.6425 | 0.87 | 0.28 | 0.75 | 0.38 |
| **SVM Poly 3** | 0.6425 | 0.83 | 0.35 | 0.74 | 0.43 |
| **SVM Radial** | 0.64 | 0.87 | 0.27 | 0.75 | 0.37 |
| **KNN Best** | 0.63 | 0.87 | 0.25 | 0.74 | 0.34 |
| **Decision Tree** | 0.63 | 0.92 | 0.17 | 0.75 | 0.27 |
| **Majority Voting** | 0.608 | N/A | N/A | N/A | N/A |
| **Weighted Voting** | 0.615 | N/A | N/A | N/A | N/A |
| **Bagging** | 0.63 | 0.91 | 0.19 | 0.75 | 0.29 |

De tots m'agradaria destacar els dos models que han obtingut una millor accuracy; les SVM polinòmiques de grau 2 i 3.

| Grau | True Negative (TN) | False Negative (FN) | True Positive (TP) | False Positive (FP) | Accuracy | Recall (Class 0) | Recall (Class 1) | F1-Score (Class 0) | F1-Score (Class 1) |
|---|---|---|---|---|---|---|---|---|---|
| 2 | 213 | 32 | 44 | 111 | 0.6425 | 0.87 | 0.28 | 0.75 | 0.38 |
| 3 | 203 | 42 | 54 | 101 | 0.6425 | 0.83 | 0.35 | 0.74 | 0.43 |

Entre aquests dos, com he comentat al seu apartat, escolliria el de grau 3. Aquest, en comparació a tota la resta de models, és el que obté una accuracy més alta i a més, tot i mostrar tendència a la classe més poblada (DISRUPTED = 0,Class 0), també es centra en detectar de manera correcta els casos on els vols sí tenen disrupcions.

La SVM Radial també té una accuracy i comportament similars als del lineal de grau 2. En general tots segueixen el mateix comportament, a diferència del Naïve Bayes.

Naïve Bayes és el model que li dona més prioritat a la classe 1, inclús més que el percentatge del dataset (0.38 DISRUPTED i 0.62 NO DISRUPTED) amb un recall de 0.45 i 0.65. Tot i fixar-se en la classe 1, crec que ho fa massa, ja que veient la f1-score i comparant-la amb el polinòmic de grau 3, aquest darrer té 10 punts més per la classe 0 i només 2 punts menys per la classe 1.

Com a conclusió, diria que tots els models tenen un comportament similar, a excepció de Naïve Bayes. El model SVM polinòmic de grau 3 és el que més accuracy té i, a més, entre els models amb millor accuracy, és el que menys es centra en la classe predominant i té un recall més parell amb la proporció del dataset, juntament amb uns millors valors per a f1-score.