S09 T02: Aprenentatge Supervisat - Regressions

Descripció:

Anem a practicar i a familiaritzar-nos amb regressions

NIVELL 1

Exercici 1

Crea almenys tres models de regressió diferents per intentar predir el millor possible l’endarreriment dels vols (ArrDelay) de DelayedFlights.csv.

In [1]:

```
# Crido a les llibreries necessàries
# Faig entrar l'arxiu CSV gràcies a pandas

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

delayedFlightsAmbNaN = pd.read_csv(r'C:\Users\Anna\DataScience\SPRINTS\SPRINT 9\DelayedFli


# Elimino els NaN per fer el dataset algo més petit
delayedFlights = delayedFlightsAmbNaN.dropna()

display(delayedFlights)
```

	Unnamed: 0	Year	Month	DayofMonth	DayOfWeek	DepTime	CRSDepTime	ArrTime	CRSArrTime	Uniqu
3	4	2008	1	3	4	1829.0	1755	1959.0	1925	
5	6	2008	1	3	4	1937.0	1830	2037.0	1940	
7	11	2008	1	3	4	1644.0	1510	1845.0	1725	
9	16	2008	1	3	4	1452.0	1425	1640.0	1625	
11	18	2008	1	3	4	1323.0	1255	1526.0	1510	
...	...	...	...	...	...	...	...	...	...	
1936751	7009705	2008	12	13	6	921.0	830	1112.0	1008	
1936752	7009709	2008	12	13	6	1552.0	1520	1735.0	1718	
1936753	7009710	2008	12	13	6	1250.0	1220	1617.0	1552	
1936754	7009717	2008	12	13	6	657.0	600	904.0	749	
1936755	7009718	2008	12	13	6	1007.0	847	1149.0	1010	

1247486 rows × 30 columns

In [2]:

```
delayedFlights.count()
```

```
Unnamed: 0            1247486
```

Out[2]:
```
Year                   1247486
Month                  1247486
DayofMonth             1247486
DayOfWeek              1247486
DepTime                1247486
CRSDepTime             1247486
ArrTime                1247486
CRSArrTime             1247486
UniqueCarrier          1247486
FlightNum              1247486
TailNum                1247486
ActualElapsedTime      1247486
CRSElapsedTime         1247486
AirTime                1247486
ArrDelay               1247486
DepDelay               1247486
Origin                 1247486
Dest                   1247486
Distance               1247486
TaxiIn                 1247486
TaxiOut                1247486
Cancelled              1247486
CancellationCode       1247486
Diverted               1247486
CarrierDelay           1247486
WeatherDelay           1247486
NASDelay               1247486
SecurityDelay          1247486
LateAircraftDelay      1247486
dtype: int64
```

In [3]:
```
delayedFlights.shape
```

Out[3]:
```
(1247486, 30)
```

In [4]:
```
delayedFlights.describe()
```

Out[4]:

	Unnamed: 0	Year	Month	DayofMonth	DayOfWeek	DepTime	CRSDepTime	ArrTin
count	1.247486e+06	1247486.0	1.247486e+06	1.247486e+06	1.247486e+06	1.247486e+06	1.247486e+06	1.247486e+
mean	3.319515e+06	2008.0	6.065399e+00	1.572542e+01	3.980082e+00	1.558832e+03	1.487949e+03	1.616749e+
std	2.079531e+06	0.0	3.508937e+00	8.793008e+00	1.993270e+00	4.543300e+02	4.211782e+02	5.839416e+
min	4.000000e+00	2008.0	1.000000e+00	1.000000e+00	1.000000e+00	1.000000e+00	0.000000e+00	1.000000e+
25%	1.484624e+06	2008.0	3.000000e+00	8.000000e+00	2.000000e+00	1.232000e+03	1.150000e+03	1.326000e+
50%	3.224052e+06	2008.0	6.000000e+00	1.600000e+01	4.000000e+00	1.618000e+03	1.529000e+03	1.737000e+
75%	4.921396e+06	2008.0	9.000000e+00	2.300000e+01	6.000000e+00	1.924000e+03	1.830000e+03	2.048000e+
max	7.009718e+06	2008.0	1.200000e+01	3.100000e+01	7.000000e+00	2.400000e+03	2.359000e+03	2.400000e+

8 rows × 25 columns

In [5]:
```
delayedFlights.dtypes
```

Out[5]:
```
Unnamed: 0             int64
Year                   int64
Month                  int64
DayofMonth             int64
```

```
DayOfWeek              int64
DepTime              float64
CRSDepTime             int64
ArrTime              float64
CRSArrTime             int64
UniqueCarrier         object
FlightNum              int64
TailNum               object
ActualElapsedTime    float64
CRSElapsedTime       float64
AirTime              float64
ArrDelay             float64
DepDelay             float64
Origin                object
Dest                  object
Distance               int64
TaxiIn               float64
TaxiOut              float64
Cancelled              int64
CancellationCode      object
Diverted               int64
CarrierDelay         float64
WeatherDelay         float64
NASDelay             float64
SecurityDelay        float64
LateAircraftDelay    float64
dtype: object
```

In [6]:

```python
#Primer elimino els atributs que no són numèrics, per facilitar l'estudi de les regressior

numerics = delayedFlights
numerics.drop(["Unnamed: 0","UniqueCarrier", "TailNum", "Origin", "Dest", "CancellationCoc
print(numerics)
```

```
C:\Users\Anna\anaconda3\lib\site-packages\pandas\core\frame.py:4906: SettingWithCopyWarnin
g:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_gu
ide/indexing.html#returning-a-view-versus-a-copy
  return super().drop(
         Year  Month  DayofMonth  DayOfWeek  DepTime  CRSDepTime  ArrTime  \
3        2008      1           3          4   1829.0        1755   1959.0
5        2008      1           3          4   1937.0        1830   2037.0
7        2008      1           3          4   1644.0        1510   1845.0
9        2008      1           3          4   1452.0        1425   1640.0
11       2008      1           3          4   1323.0        1255   1526.0
...       ...    ...         ...        ...      ...         ...      ...
1936751  2008     12          13          6    921.0         830   1112.0
1936752  2008     12          13          6   1552.0        1520   1735.0
1936753  2008     12          13          6   1250.0        1220   1617.0
1936754  2008     12          13          6    657.0         600    904.0
1936755  2008     12          13          6   1007.0         847   1149.0

         CRSArrTime  FlightNum  ActualElapsedTime  ...  Distance  TaxiIn  \
3              1925       3920               90.0  ...       515     3.0
5              1940        509              240.0  ...      1591     3.0
7              1725       1333              121.0  ...       828     6.0
9              1625        675              228.0  ...      1489     7.0
11             1510          4              123.0  ...       838     4.0
...             ...        ...                ...  ...       ...     ...
1936751        1008       1616              111.0  ...       545     8.0
1936752        1718       1620               43.0  ...       151     9.0
1936753        1552       1621              147.0  ...       906     9.0
1936754         749       1631              127.0  ...       481    15.0
```

```
1936755        1010       1631               162.0  ...       689      8.0

         TaxiOut  Cancelled  Diverted  CarrierDelay  WeatherDelay  NASDelay  \
3           10.0          0         0           2.0           0.0       0.0
5            7.0          0         0          10.0           0.0       0.0
7            8.0          0         0           8.0           0.0       0.0
9            8.0          0         0           3.0           0.0       0.0
11           9.0          0         0           0.0           0.0       0.0
...          ...        ...       ...           ...           ...       ...
1936751     21.0          0         0          51.0           0.0      13.0
1936752      7.0          0         0           0.0           0.0       0.0
1936753     18.0          0         0           3.0           0.0       0.0
1936754     34.0          0         0           0.0          57.0      18.0
1936755     32.0          0         0           1.0           0.0      19.0

         SecurityDelay  LateAircraftDelay
3                  0.0               32.0
5                  0.0               47.0
7                  0.0               72.0
9                  0.0               12.0
11                 0.0               16.0
...                ...                ...
1936751            0.0                0.0
1936752            0.0               17.0
1936753            0.0               22.0
1936754            0.0                0.0
1936755            0.0               79.0

[1247486 rows x 24 columns]
```

CREACIÓ TRAIN TEST

In [7]:

```
from sklearn.model_selection import train_test_split
numerics = numerics.sample(100000, random_state=0)
```

In [8]:

```
# Indico quin vull que sigui el meu target o input o y
y = np.array(numerics["ArrDelay"])
x = np.array(numerics)

#Creo el train test
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_state=0)
```

REGRESSIÓ LINEAL MÚLTIPLE

In [9]:

```
#Primer faré una regressió lineal múltiple, importo la llibreria

from sklearn.linear_model import LinearRegression
```

In [10]:

```
# Creo el model de regressió
model = LinearRegression(n_jobs=-1).fit(X_train, y_train)

r_sq = model.score(X_test, y_test)
print(f"coefficient of determination: {r_sq}")

print(f"intercept: {model.intercept_}")

print(f"coefficients: {model.coef_}")
```

```
coefficient of determination: 1.0
intercept: 8.526512829121202e-14
coefficients: [ 0.00000000e+00  2.25514052e-15  4.52383356e-17  3.14819615e-15
```

```
 -8.37004077e-17 -5.11743425e-17  3.12250226e-17  1.33140027e-16
 -9.71445147e-17  1.74418605e-01 -2.32558140e-01  5.81395349e-02
  6.39534884e-01  2.32558140e-01  1.11455983e-16  5.81395349e-02
  5.81395349e-02  0.00000000e+00 -2.77555756e-17  1.27906977e-01
  1.27906977e-01  1.27906977e-01  1.27906977e-01  1.27906977e-01]
```

In [11]:

```
prediccioLineal = model.predict(X_test)
print(prediccioLineal)
```

```
[122.  39.  27. ...  43.  36.  83.]
```

In [12]:

```
# Comparació X-test amb la predicció
pred_lineal = pd.DataFrame({"Actual":y_test, "Predicted":prediccioLineal})
pred_lineal

#No entenc perquè em surt igual
```

Out[12]:

	Actual	Predicted
0	122.0	122.0
1	39.0	39.0
2	27.0	27.0
3	42.0	42.0
4	52.0	52.0
...	...	...
19995	45.0	45.0
19996	18.0	18.0
19997	43.0	43.0
19998	36.0	36.0
19999	83.0	83.0

20000 rows × 2 columns

REGRESSIÓ POLINOMINAL

In [13]:

```
# Importo les llibreries
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures

# Els outputs x i inputs y seran els mateixos que l'anterior regressió

#Creo la instancia necessària
transformer = PolynomialFeatures(degree=2, include_bias=False)
X = transformer.fit_transform(x)
X.reshape((-1,1))
#Creo el train test de nou amb la X correcte
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state=0)
```

In [14]:

```
# Creo el model de regressió i l'aplico
model = LinearRegression(n_jobs=-1).fit(X_train, y_train)

r_sq = model.score(X_test, y_test)
print(f"coefficient of determination: {r_sq}")
```

```python
print(f"intercept: {model.intercept_}")

print(f"coefficients: {model.coef_}")
```

```
coefficient of determination: 1.0
intercept: 9.741540907270974e-12
coefficients: [ 1.22421090e-19  2.03820373e-16  1.42432613e-16  3.27200357e-17
  5.40237614e-17 -1.13739584e-17  2.13875819e-17  3.49705493e-18
 -3.68798145e-18  4.32578851e-08 -5.76771802e-08  1.44192951e-08
  1.58612246e-07  5.76771802e-08  1.63107726e-17  1.44192950e-08
  1.44192950e-08  2.12774676e-18  1.75505227e-18  3.17224491e-08
  3.17224491e-08  3.17224491e-08  3.17224491e-08  3.17224491e-08
 -2.08708918e-18  6.03934822e-18  6.40965713e-18  3.99508384e-18
 -3.09336432e-18  2.34448132e-18  2.38651533e-19 -3.05143619e-19
  6.77626358e-21  8.68618334e-05 -1.15815778e-04  2.89539445e-05
  3.18493389e-04  1.15815778e-04  6.85673171e-19  2.89539445e-05
  2.89539445e-05  3.04931861e-20  0.00000000e+00  6.36986778e-05
  6.36986778e-05  6.36986778e-05  6.36986778e-05  6.36986778e-05
  2.82208518e-17  2.05455733e-17 -1.38575807e-16 -4.37608984e-18
 -2.53881748e-18 -5.65164702e-18 -9.35471790e-19 -8.49904918e-19
 -5.38234896e-18  1.11995861e-18 -2.65719983e-17 -9.28677700e-18
 -3.07133733e-18  1.20582155e-17  1.81504146e-17  3.12244262e-18
 -4.74338450e-20 -1.62630326e-19 -5.77870620e-18  2.28248299e-18
  4.33836131e-18 -1.59581453e-17  5.66076544e-18  2.72553241e-17
 -1.50784231e-17 -1.42645642e-19 -5.20109993e-18 -2.49149447e-18
  1.50888332e-18  1.14142984e-18 -2.53824490e-17  1.27352851e-17
  8.50603059e-18 -1.75125964e-17  2.05959469e-17 -1.08409629e-18
 -1.94103622e-17 -1.44414063e-17 -4.06575815e-20 -6.77626358e-21
  4.71785792e-18 -1.39794543e-17 -1.31269096e-18 -8.32183961e-18
  1.47599296e-18  1.67586496e-16 -2.88807067e-17 -8.76882918e-18
 -9.60273311e-18  1.91851639e-18  9.48403600e-18 -3.52015664e-17
 -2.17893543e-17 -6.98386833e-18  1.42651701e-17  2.76868609e-17
  1.43206802e-17 -1.62187754e-17 -1.19628147e-17  1.01643954e-20
 -2.03287907e-20 -2.16487069e-17 -1.61201723e-17 -2.06184489e-17
  3.36960845e-17  3.90262411e-17  1.08843734e-18  1.14974135e-18
 -4.56339000e-19  9.37665473e-19  8.47032947e-21  3.43683618e-18
  2.34151670e-19 -2.77297411e-19 -2.94333361e-18 -1.38320480e-18
 -1.56701095e-20 -1.16167922e-18  3.82162737e-18 -5.92923063e-21
  9.31736242e-21 -2.16586325e-18  1.94993602e-18  2.11610006e-18
 -7.46268876e-18  2.63109609e-19 -2.13918171e-18 -9.59264813e-20
  1.18203448e-18  1.55854062e-19  1.14137690e-19  3.72043340e-18
  3.56389113e-19  4.85138121e-19  1.02745097e-18 -1.06768503e-18
  7.51709977e-18 -3.39924910e-18  3.81164826e-21  1.27054942e-21
  2.16300451e-18 -3.16890907e-18 -3.54398585e-18  8.26756393e-18
 -5.33990746e-18 -2.70203510e-19  4.62056473e-19  6.35274710e-20
  3.89486925e-18 -3.44620649e-18 -3.55383261e-18 -9.24430583e-19
  3.81831865e-18  3.47283508e-19  3.43994638e-18  9.45182890e-19
 -8.47032947e-22  1.05879118e-22  2.79536754e-18 -5.33491790e-19
  1.63540886e-18 -3.01124938e-18 -1.82916765e-18 -1.29087821e-18
 -2.08634803e-19  1.74107622e-18 -2.43746965e-18 -7.82076108e-19
  3.00273180e-19  2.07205435e-19  1.33831206e-19  1.01743215e-18
 -1.90476534e-19 -2.11758237e-22 -4.23516474e-22  1.41510089e-18
 -1.87167812e-19  5.31544938e-18 -2.21330929e-18 -2.49408851e-18
 -1.27054942e-20 -3.91223343e-18  3.91572744e-18  2.02276762e-18
 -8.56456189e-19 -4.01493617e-19 -2.57074499e-19 -4.79281682e-19
  4.37098118e-18  0.00000000e+00  0.00000000e+00 -1.03843592e-18
  1.53499575e-18  8.44280090e-19  2.52222878e-19  6.30510150e-19
 -3.88944956e-18  7.21672071e-18 -8.29059967e-19 -5.90084179e-19
  1.04241896e-17 -1.04269756e-18  1.48870590e-18 -3.28707679e-18
  0.00000000e+00  0.00000000e+00  1.43036402e-18  8.91704588e-18
 -3.69816570e-18 -1.23817974e-18 -5.66796067e-18 -1.72605595e-17
  1.18610619e-17  1.40848013e-17 -1.12414838e-17  7.76199817e-19
  1.30676215e-18 -5.37796438e-18  0.00000000e+00  0.00000000e+00
  1.13340742e-17 -2.34789882e-18  7.53155227e-18 -4.74958704e-18
```

```
  3.09541573e-18 -5.61449172e-18 -4.34478933e-18  8.67766726e-18
  2.54480461e-18  4.75210133e-18 -5.66165425e-19  0.00000000e+00
  0.00000000e+00  2.91338058e-18 -1.28887313e-18 -6.48781246e-18
  7.96039939e-18 -7.19541254e-18  1.58271415e-18  1.58652050e-17
 -1.58686329e-19  8.07144499e-18 -4.10293578e-18  0.00000000e+00
  0.00000000e+00 -1.37020285e-17  1.98412596e-17 -1.40121815e-17
  1.83866432e-17 -9.22547258e-18 -4.91460427e-18 -8.30939321e-19
  7.82879363e-18 -6.20154635e-18  0.00000000e+00  0.00000000e+00
 -3.28141226e-18  8.38103243e-18 -3.03330770e-18  1.48833878e-17
 -1.11313364e-18 -8.55503277e-20 -4.73173284e-18  2.50139417e-21
  0.00000000e+00  0.00000000e+00 -4.45900643e-18 -1.93054691e-18
 -1.64935844e-18  5.47258588e-18  3.50878104e-18 -3.10566926e-18
 -1.69357786e-19  0.00000000e+00  0.00000000e+00  1.03092504e-17
  1.21893057e-17 -2.90364652e-18 -1.50653308e-17  3.54804772e-18
 -2.61318598e-18  0.00000000e+00  0.00000000e+00 -1.16946460e-17
 -2.05857844e-18  5.65819994e-18  5.86578763e-18 -1.92041105e-18
  0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00
  0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00
  0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00
  0.00000000e+00  5.81119196e-18 -1.70146478e-17  1.10809448e-17
 -2.27282661e-17  8.94398839e-18 -3.83768989e-17 -1.99661479e-17
  1.08454145e-16 -1.33826426e-17  4.35599597e-18 -2.20583961e-17
  1.24413961e-17 -2.47035736e-17 -2.05806277e-17  3.44470433e-18]
```

In [15]:

```python
prediccioPolinominal = model.predict(X_test)
print(f"predicted response:\n{prediccioPolinominal}")
```

```
predicted response:
[122.  39.  27. ...  43.  36.  83.]
```

In [16]:

```python
# Comparació X-test amb la predicció
pred_polinominal = pd.DataFrame({"Actual":y_test, "Predicted":prediccioPolinominal})
pred_polinominal

#No entenc perquè em surt igual
```

Out[16]:

	Actual	Predicted
0	122.0	122.0
1	39.0	39.0
2	27.0	27.0
3	42.0	42.0
4	52.0	52.0
...	...	...
19995	45.0	45.0
19996	18.0	18.0
19997	43.0	43.0
19998	36.0	36.0
19999	83.0	83.0

20000 rows × 2 columns

DECISION TREE REGRESSOR

In [17]:

```python
# Importo les llibreries
```

```
from sklearn.tree import DecisionTreeRegressor

# Els outputs x i inputs y seran els mateixos que l'anterior regressió

#Creo la instancia necessària
regressor = DecisionTreeRegressor(random_state=0)
regressor.fit(X_train,y_train)
```

Out[17]:

```
▾        DecisionTreeRegressor
DecisionTreeRegressor(random_state=0)
```

In [18]:

```
predliccioTree = regressor.predict(X_test)
print(f"predicted response:\n{predliccioTree}")
```

```
predicted response:
[122.  39.  27. ...  43.  36.  83.]
```

In [19]:

```
# Comparació X-test amb la predicció
pred_Tree = pd.DataFrame({"Actual":y_test, "Predicted":predliccioTree})
pred_Tree

#No entenc perquè em surt igual
```

Out[19]:

	Actual	Predicted
0	122.0	122.0
1	39.0	39.0
2	27.0	27.0
3	42.0	42.0
4	52.0	52.0
...	...	...
19995	45.0	45.0
19996	18.0	18.0
19997	43.0	43.0
19998	36.0	36.0
19999	83.0	83.0

20000 rows × 2 columns

Exercici 2

Compara’ls en base al MSE i al R2 .

In [20]:

```
import statsmodels.api as sm
from sklearn import metrics
```

In [21]:

```
print("REGRESSIÓ LINEAL")
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, predliccioLineal))
print('Mean Squared Error:', metrics.mean_squared_error(y_test, predliccioLineal))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test, predliccioLine
```

```
REGRESSIÓ LINEAL
Mean Absolute Error: 1.7859509426898511e-13
Mean Squared Error: 4.907812483092654e-26
Root Mean Squared Error: 2.2153583193453502e-13
```

In [22]:

```
print("REGRESSIÓ POLINOMINAL")
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, prediccioPolinominal))
print('Mean Squared Error:', metrics.mean_squared_error(y_test, prediccioPolinominal))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test, prediccioPoli
```

```
REGRESSIÓ POLINOMINAL
Mean Absolute Error: 3.4243665680833146e-13
Mean Squared Error: 2.746917523940014e-25
Root Mean Squared Error: 5.241104391194678e-13
```

In [23]:

```
print("DECISION TREE REGRESSOR")
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, prediccioTree))
print('Mean Squared Error:', metrics.mean_squared_error(y_test, prediccioTree))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test, prediccioTree
```

```
DECISION TREE REGRESSOR
Mean Absolute Error: 0.01635
Mean Squared Error: 0.41935
Root Mean Squared Error: 0.6475723897758459
```

Exercici 3

Entrena'ls utilitzant els diferents paràmetres que admeten.

CANVIEM PARÀMETRES DE LA REGRESSIÓ LINEAL

In [24]:

```
# Creo el model de regressió
model = LinearRegression(fit_intercept = False, normalize=True, n_jobs=-1).fit(X_train, y_

r_sq = model.score(X_test, y_test)
print(f"coefficient of determination: {r_sq}")

print(f"intercept: {model.intercept_}")

print(f"coefficients: {model.coef_}")
```

```
C:\Users\Anna\anaconda3\lib\site-packages\sklearn\linear_model\_base.py:141: FutureWarnin
g: 'normalize' was deprecated in version 1.0 and will be removed in 1.2.
If you wish to scale the data, use Pipeline with a StandardScaler in a preprocessing stag
e. To reproduce the previous behavior:

from sklearn.pipeline import make_pipeline

model = make_pipeline(StandardScaler(with_mean=False), LinearRegression())

If you wish to pass a sample_weight parameter, you need to pass it as a fit parameter to e
ach step of the pipeline as follows:

kwargs = {s[0] + '__sample_weight': sample_weight for s in model.steps}
model.fit(X, y, **kwargs)


  warnings.warn(
coefficient of determination: 1.0
intercept: 0.0
coefficients: [-4.84628442e-12 -2.48677398e-08 -2.04300821e-07 -1.19744257e-07
  1.42077538e-07 -1.71605380e-07 -1.85155991e-07  1.47736012e-10
```

-2.31554081e-07	-2.14349746e-07	1.49193218e-07	-5.25919096e-08
2.20749683e-07	-2.74822614e-07	1.17568035e-07	-9.41946106e-09
4.29526142e-08	-7.08787719e-08	-2.68606276e-09	3.13684329e-08
1.71670345e-08	2.55645243e-08	3.26464843e-08	3.30702208e-08
2.41393802e-15	1.23842642e-11	1.01743443e-10	5.96335962e-11
-7.07557453e-11	8.54608426e-11	9.22091605e-11	-7.35736473e-14
1.15315777e-10	8.68244811e-05	-1.15745099e-04	2.89206761e-05
3.18574500e-04	1.15745161e-04	-5.85498175e-11	2.89206546e-05
2.89206285e-05	4.68126480e-10	-2.19455400e-09	6.36883181e-05
6.36883252e-05	6.36883210e-05	6.36883175e-05	6.36883173e-05
7.28355358e-16	4.49678964e-16	6.30406246e-16	1.85881943e-17
1.79464940e-17	9.05300708e-18	4.14002408e-17	5.92506164e-19
1.38699192e-10	-5.69000202e-11	-8.17995070e-11	-4.36870553e-11
5.69000691e-11	2.35197359e-17	-8.17990497e-11	-8.17992213e-11
3.50041170e-10	-2.91553600e-10	-1.32132715e-11	-1.32132260e-11
-1.32131972e-11	-1.32128794e-11	-1.32131207e-11	-1.01499353e-17
-1.31957096e-15	1.09192142e-17	-4.54467124e-17	2.00463847e-17
1.38258276e-17	4.48940697e-18	1.87193843e-10	-2.61276527e-10
7.40822240e-11	-3.73267487e-10	2.61276281e-10	6.77896184e-17
7.40822339e-11	7.40821544e-11	-1.10928571e-10	2.26747870e-11
1.11991101e-10	1.11991145e-10	1.11991226e-10	1.11991189e-10
1.11991087e-10	-2.34380393e-15	8.91995028e-17	-3.53917480e-17
-3.53097513e-17	-1.40590429e-17	1.17515895e-17	4.90264964e-11
-2.97813764e-11	-1.92460535e-11	-8.58231367e-11	2.97807562e-11
1.52860918e-16	-1.92460178e-11	-1.92454928e-11	1.03605603e-10
1.13426630e-11	5.60420425e-11	5.60422612e-11	5.60422184e-11
5.60426605e-11	5.60421307e-11	9.01878331e-19	-7.14895807e-19
-1.27838448e-18	-1.35991140e-18	1.11278953e-19	1.20256913e-08
1.85673336e-08	-3.05930249e-08	1.51259467e-08	-1.85673336e-08
3.01861367e-18	-3.05930249e-08	-3.05930249e-08	-1.22083109e-11
1.38996097e-11	3.44138685e-09	3.44138685e-09	3.44138685e-09
3.44138671e-09	3.44138685e-09	2.54914565e-18	-1.68136040e-19
1.15143541e-19	-3.00114361e-19	-2.05464368e-08	3.35166432e-08
-1.29702063e-08	3.42254682e-08	-3.35166432e-08	-5.51693734e-18
-1.29702063e-08	-1.29702063e-08	1.00287797e-11	-3.35850167e-11
-7.08825007e-10	-7.08825038e-10	-7.08825027e-10	-7.08824890e-10
-7.08825008e-10	-2.25946039e-19	5.21984054e-19	-1.95241094e-19
1.85870949e-08	-8.12833412e-09	-1.04587608e-08	-5.95469704e-09
8.12833412e-09	-1.30866590e-18	-1.04587608e-08	-1.04587608e-08
7.78148622e-12	-4.18300832e-13	-2.17363708e-09	-2.17363708e-09
-2.17363708e-09	-2.17363708e-09	-2.17363708e-09	4.41092407e-19
2.63003730e-19	2.81308094e-08	3.52677583e-08	-6.33985677e-08
3.07299661e-08	-3.52677583e-08	7.41789104e-19	-6.33985677e-08
-6.33985677e-08	-9.32932560e-13	-2.05205568e-12	4.53779221e-09
4.53779224e-09	4.53779222e-09	4.53779218e-09	4.53779221e-09
4.34104385e-21	1.03338350e-08	3.17994449e-08	-4.21332799e-08
4.05394047e-08	-3.17994449e-08	-5.63276910e-19	-4.21332799e-08
-4.21332799e-08	0.00000000e+00	0.00000000e+00	-8.73995980e-09
-8.73995980e-09	-8.73995979e-09	-8.73995981e-09	-8.73995980e-09
3.42989000e-09	-2.25063584e-09	2.73781279e-09	-2.17858523e-09
1.13434994e-09	3.11887719e-08	-8.89631876e-10	-9.32566838e-10
0.00000000e+00	0.00000000e+00	-3.13095991e-10	1.79645619e-10
-1.67522196e-10	-1.26729713e-11	-3.35898660e-10	-2.18565149e-09
3.86733656e-09	-2.37586021e-09	3.52219700e-09	-3.35652587e-08
6.15424301e-10	7.60550625e-10	0.00000000e+00	0.00000000e+00
-5.97316461e-11	-5.04627588e-10	-7.43891254e-10	-4.36179224e-10
1.43697042e-10	-5.59875214e-09	2.04592364e-09	-1.90194499e-09
2.37648680e-09	-4.31814751e-09	-4.42033846e-09	0.00000000e+00
0.00000000e+00	1.26747459e-10	7.89017981e-11	6.65333071e-10
2.02771825e-10	-5.38785163e-11	-4.15807123e-10	2.46920373e-09
-2.95641756e-08	-8.52561912e-10	-7.76932002e-10	0.00000000e+00
0.00000000e+00	3.68590904e-11	-1.36957073e-10	-7.46981759e-10
-9.10872421e-11	6.77211639e-11	-1.33654586e-09	3.35652587e-08
1.34996711e-09	1.20484102e-09	0.00000000e+00	0.00000000e+00
-8.82717328e-10	-4.37821406e-10	-1.98557687e-10	-5.06269744e-10
-1.08614608e-09	-1.91852963e-18	2.37648684e-09	2.37648678e-09

```
 0.00000000e+00  0.00000000e+00 -4.00108310e-09 -4.00108311e-09
-4.00108309e-09 -4.00108312e-09 -4.00108311e-09  1.28060454e-09
 2.45901806e-09  0.00000000e+00  0.00000000e+00 -2.26679189e-10
-2.74524818e-10  3.11906653e-10 -1.50654856e-10 -4.07305178e-10
 1.17841364e-09  0.00000000e+00  0.00000000e+00 -1.57183022e-10
-2.05028701e-10  3.81402779e-10 -8.11583573e-11 -3.37809107e-10
 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00
 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00
 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00
 0.00000000e+00  1.29007492e-10 -1.30647781e-11  3.57696252e-10
 9.51391937e-12  4.30581667e-10 -1.42072265e-10  8.66164939e-11
-2.61564835e-10  1.59501890e-10  2.28688713e-10  1.09195085e-10
 5.30262915e-10 -1.19493701e-10  1.82080375e-10  3.01574154e-10]
```

In [25]:

```
prediccioLineal = model.predict(X_test)
print(prediccioLineal)
```

```
[122.  39.  27. ...  43.  36.  83.]
```

In [26]:

```
# Comparació X-test amb la predicció
pred_lineal = pd.DataFrame({"Actual":y_test, "Predicted":prediccioLineal})
pred_lineal

#No entenc perquè em surt igual
```

Out[26]:

	Actual	Predicted
0	122.0	122.0
1	39.0	39.0
2	27.0	27.0
3	42.0	42.0
4	52.0	52.0
...	...	...
19995	45.0	45.0
19996	18.0	18.0
19997	43.0	43.0
19998	36.0	36.0
19999	83.0	83.0

20000 rows × 2 columns

CANVIEM PARÀMETRES DE LA REGRESSIÓ POLINOMINAL

In [27]:

```
#Creo la instancia necessària
transformer = PolynomialFeatures(degree=3, include_bias=True, interaction_only=True)
X = transformer.fit_transform(x)
X.reshape((-1,1))
#Creo el train test de nou amb la X correcte
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state=0)
```

In [28]:

```
# Creo el model de regressió i l'aplico
model = LinearRegression(n_jobs=-1).fit(X_train, y_train)
```

```
r_sq = model.score(X_test, y_test)
print(f"coefficient of determination: {r_sq}")

print(f"intercept: {model.intercept_}")

print(f"coefficients: {model.coef_}")
```

```
coefficient of determination: 1.0
intercept: 0.00016865601577364941
coefficients: [-7.17138572e-17 -8.39919234e-08  4.53184445e-08 ... -3.44700174e-16
  0.00000000e+00  3.77823947e-15]
```

In [29]:

```
prediccioPolinominal = model.predict(X_test)
print(f"predicted response:\n{prediccioPolinominal}")
```

```
predicted response:
[122.  39.  27. ...  43.  36.  83.]
```

In [30]:

```
# Comparació X-test amb la predicció
pred_polinominal = pd.DataFrame({"Actual":y_test, "Predicted":prediccioPolinominal})
pred_polinominal

#No entenc perquè em surt igual
```

Out[30]:

	Actual	Predicted
0	122.0	122.0
1	39.0	39.0
2	27.0	27.0
3	42.0	42.0
4	52.0	52.0
...	...	...
19995	45.0	45.0
19996	18.0	18.0
19997	43.0	43.0
19998	36.0	36.0
19999	83.0	83.0

20000 rows × 2 columns

CANVIEM PARÀMETRES DE LA REGRESSIÓ D'ARBRE (DECISION TREE REGRESSION)

In [ ]:

```

```

Exercici 4

Compara el seu rendiment utilitzant l'aproximació traint/test o utilitzant totes les dades (validació interna)

REGRESSIÓ LINEAL MÚLTIPLE

In [31]:

```
# Creo el model de regressió
model = LinearRegression(n_jobs=-1).fit(x, y)
```

```
r_sq = model.score(x, y)
print(f"coefficient of determination: {r_sq}")

print(f"intercept: {model.intercept_}")

print(f"coefficients: {model.coef_}")
```

```
coefficient of determination: 1.0
intercept: 5.826450433232822e-13
coefficients: [ 0.00000000e+00  1.27051147e-14  1.39862080e-15 -3.82415318e-14
 -3.03576608e-16  1.01915004e-16 -1.90819582e-16  1.26634814e-16
  0.00000000e+00  1.74418605e-01 -2.32558140e-01  5.81395349e-02
  6.39534884e-01  2.32558140e-01  2.79290480e-16  5.81395349e-02
  5.81395349e-02 -1.21430643e-17  2.77555756e-17  1.27906977e-01
  1.27906977e-01  1.27906977e-01  1.27906977e-01  1.27906977e-01]
```

In [32]:

```
prediccioLineal = model.predict(x)
print(prediccioLineal)
```

```
[ 16.   99.   29. ... 110.   54.   18.]
```

In [33]:

```
# Comparació X-test amb la predicció
pred_lineal = pd.DataFrame({"Actual":y, "Predicted":prediccioLineal})
pred_lineal

#No entenc perquè em surt igual
```

Out[33]:

	Actual	Predicted
0	16.0	16.0
1	99.0	99.0
2	29.0	29.0
3	32.0	32.0
4	23.0	23.0
...	...	...
99995	23.0	23.0
99996	26.0	26.0
99997	110.0	110.0
99998	54.0	54.0
99999	18.0	18.0

100000 rows × 2 columns

In [34]:

```
print("REGRESSIÓ LINEAL")
print('Mean Absolute Error:', metrics.mean_absolute_error(y, prediccioLineal))
print('Mean Squared Error:', metrics.mean_squared_error(y, prediccioLineal))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y, prediccioLineal)))
```

```
REGRESSIÓ LINEAL
Mean Absolute Error: 1.5488041071876069e-13
Mean Squared Error: 4.109286176381836e-26
Root Mean Squared Error: 2.0271374340142397e-13
```

REGRESSIÓ POLINOMINAL

In [35]:
```python
#Creo la instancia necessària
transformer = PolynomialFeatures(degree=2, include_bias=False)
X = transformer.fit_transform(x)
X.reshape((-1,1))
```

Out[35]:
```
array([[2.008e+03],
       [2.000e+00],
       [2.600e+01],
       ...,
       [0.000e+00],
       [0.000e+00],
       [1.690e+02]])
```

In [36]:
```python
# Creo el model de regressió i l'aplico
model = LinearRegression(n_jobs=-1).fit(X, y)

r_sq = model.score(X, y)
print(f"coefficient of determination: {r_sq}")

print(f"intercept: {model.intercept_}")

print(f"coefficients: {model.coef_}")
```

```
coefficient of determination: 1.0
intercept: 6.181721801112872e-13
coefficients: [-6.93341280e-20 -4.04747918e-16  1.34813764e-17  3.03989960e-16
 -5.40949121e-17 -1.23267011e-16  1.11930322e-16  5.73644593e-17
 -3.87305815e-17  4.32578851e-08 -5.76771802e-08  1.44192951e-08
  1.58612246e-07  5.76771802e-08  2.92375259e-17  1.44192951e-08
  1.44192951e-08  2.76471554e-18 -7.18283939e-18  3.17224491e-08
  3.17224491e-08  3.17224491e-08  3.17224491e-08  3.17224491e-08
 -3.59141970e-19  1.69347297e-17 -6.29091370e-18 -3.07197674e-18
  1.45350854e-18 -1.18923426e-18  7.42000862e-19 -5.09913834e-19
  6.09863722e-20  8.68618334e-05 -1.15815778e-04  2.89539445e-05
  3.18493389e-04  1.15815778e-04 -3.37119113e-19  2.89539445e-05
  2.89539445e-05  1.86347248e-19 -1.67712524e-19  6.36986778e-05
  6.36986778e-05  6.36986778e-05  6.36986778e-05  6.36986778e-05
  8.81399787e-17  2.96756868e-17 -5.67131835e-16  4.24448210e-18
  3.23905399e-18  6.48191963e-19  1.60131579e-18 -5.78099987e-19
  5.08510936e-18  1.53552780e-17 -6.41548048e-18 -1.06271798e-17
 -4.30517730e-19 -3.36843827e-18  1.33053077e-17 -1.72733179e-18
 -3.55753838e-20  9.48676901e-20  6.88929615e-18  2.21084384e-17
  1.97428160e-19 -3.66663344e-17 -2.63361072e-18 -4.82837897e-18
 -6.32664956e-18 -3.24667729e-18 -1.01474547e-18  1.99052743e-19
  1.22396261e-18  2.75031598e-18  2.76662136e-19  1.52571810e-19
 -1.78872183e-18 -6.75085259e-19 -1.20851418e-18 -2.83756037e-19
 -6.55927756e-19  2.99978042e-18 -3.72694497e-20 -1.49077799e-19
 -1.67780021e-18  5.38908169e-20 -3.59071825e-18  5.77838913e-18
 -1.04172645e-18 -1.82383532e-16 -2.81214938e-19  1.93557616e-18
 -1.57526952e-18 -1.70317150e-18  4.73618472e-18 -5.93206290e-18
  1.46300457e-17  9.62319425e-18 -1.07470085e-17  9.71882957e-18
 -3.32640426e-18 -2.30296078e-17  7.53706626e-18  0.00000000e+00
 -3.38813179e-20 -1.34122903e-17 -1.46349803e-17  9.64857546e-18
  1.29302377e-17 -5.35444598e-18  6.03087458e-19 -1.19939865e-18
 -1.55854062e-19 -1.35525272e-20 -1.32137140e-19 -1.94912869e-18
  9.41900637e-19 -1.39336920e-18  3.21872520e-19 -3.22296036e-18
 -5.28548559e-19 -9.58523659e-19 -5.50359657e-19 -4.23516474e-21
  0.00000000e+00  2.09217138e-19  9.52488549e-19  1.82959117e-19
 -2.78223192e-18 -2.64274280e-19  8.23316025e-19 -3.09167026e-19
 -2.93073400e-19  8.47032947e-22 -6.44592073e-19  1.01898064e-18
 -5.48877350e-19  1.70761842e-18 -3.38813179e-21  6.50521303e-19
 -1.23243294e-18 -3.23143069e-19 -7.94093388e-22 -8.47032947e-22
  2.58345049e-19 -4.13775595e-19 -8.77526133e-19  2.48336164e-18
  9.19030748e-20  3.77776694e-19 -5.69206141e-19 -2.79520873e-20
  1.51788304e-18 -2.63935466e-18 -6.80802731e-20 -1.76352260e-18
```

```
  1.53651777e-18 -1.44842634e-19  1.75441699e-19  1.80841534e-19
  8.99972506e-22 -5.29395592e-23 -1.32645360e-18 -1.51830656e-19
 -7.62329653e-19 -8.20139651e-19  4.40457133e-20  4.47233396e-19
  1.23666810e-19 -3.33540399e-18  3.16536212e-18  1.72037686e-18
  1.93123512e-19 -7.11507676e-19  9.69852725e-20  1.53360609e-18
  9.06325254e-19 -7.94093388e-23  5.29395592e-22  2.01847951e-18
  2.34924588e-18  2.13791116e-18  1.68078468e-18  1.45096744e-18
 -1.93229391e-20  1.62884436e-18 -1.62905612e-18 -2.96461532e-21
 -1.19177536e-18  2.05405490e-18  7.11507676e-20 -3.26213564e-19
 -4.94667241e-19  0.00000000e+00  0.00000000e+00 -1.40438063e-18
 -1.75844040e-18 -8.88114045e-19 -1.24641560e-18 -6.87578995e-19
 -2.40387950e-18 -1.00648690e-18 -1.58162227e-18 -2.24781368e-18
 -1.01548662e-18 -5.95464162e-19 -1.81713465e-18 -5.04619878e-19
  0.00000000e+00  0.00000000e+00  3.43694206e-18 -1.74793190e-18
  4.84989890e-18 -1.25982411e-17  4.19429540e-18  7.01766797e-19
 -2.35464571e-18 -3.32036915e-18  9.88593329e-19  7.87740641e-20
  3.21752744e-18 -2.85542747e-18  0.00000000e+00  0.00000000e+00
 -5.55526558e-18  3.97614471e-18 -4.93401986e-18  7.74852009e-18
 -3.98285480e-18 -2.37402159e-18  3.49062278e-18  1.68411326e-18
  2.54787511e-18 -6.86561563e-19  2.86847708e-18  0.00000000e+00
  0.00000000e+00 -1.91101221e-18 -6.37746988e-18 -2.53321085e-18
  1.68652929e-17 -2.65237780e-18 -1.00352228e-18 -2.40705588e-18
 -1.29426634e-18 -6.28835937e-18  9.09131050e-19  0.00000000e+00
  0.00000000e+00  1.10902818e-17 -1.57370648e-17  1.13212306e-17
 -1.79509509e-17  9.99827103e-18 -4.33045594e-20 -3.15096256e-19
 -1.22447215e-18 -1.20368676e-18  0.00000000e+00  0.00000000e+00
  2.04883373e-18 -9.95853989e-18  1.12779790e-18  2.41378520e-18
  1.62365628e-18 -4.06575815e-20  1.08896673e-18  2.47206566e-18
  0.00000000e+00  0.00000000e+00  1.22735074e-18  2.12658209e-18
  1.93843490e-18 -1.46215754e-19  1.93843490e-18  9.76106210e-19
 -2.18885721e-18  0.00000000e+00  0.00000000e+00  3.40917196e-18
  5.02481037e-18  4.57604587e-18 -2.29027454e-17  3.64615258e-18
 -1.28619306e-18  0.00000000e+00  0.00000000e+00  2.07508514e-18
 -4.66576188e-19  2.92669736e-18 -6.56432202e-18  3.04160267e-18
  0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00
  0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00
  0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00
  0.00000000e+00 -7.91687285e-18  2.27926845e-17 -1.47270111e-17
  2.52788183e-17 -1.39593644e-17  3.11622429e-17  1.91683060e-17
 -1.13057515e-16  2.42270572e-17 -8.96782898e-18  3.00478952e-17
 -1.45487967e-17  2.09414392e-17  1.88373528e-17 -4.69256253e-18]
```

In [37]:

```python
prediccioPolinominal = model.predict(X)
print(f"predicted response:\n{prediccioPolinominal}")
```

```
predicted response:
[ 16.   99.   29.  ... 110.   54.   18.]
```

In [38]:

```python
# Comparació X-test amb la predicció
pred_polinominal = pd.DataFrame({"Actual":y, "Predicted":prediccioPolinominal})
pred_polinominal

#No entenc perquè em surt igual
```

Out[38]:

	Actual	Predicted
0	16.0	16.0
1	99.0	99.0
2	29.0	29.0
3	32.0	32.0
4	23.0	23.0

	Actual	Predicted
...	...	...
99995	23.0	23.0
99996	26.0	26.0
99997	110.0	110.0
99998	54.0	54.0
99999	18.0	18.0

100000 rows × 2 columns

In [39]:

```
print("REGRESSIÓ POLINOMINAL")
print('Mean Absolute Error:', metrics.mean_absolute_error(y, prediccioPolinominal))
print('Mean Squared Error:', metrics.mean_squared_error(y, prediccioPolinominal))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y, prediccioPolinomir
```

```
REGRESSIÓ POLINOMINAL
Mean Absolute Error: 2.3955818662102503e-13
Mean Squared Error: 1.0287020560438499e-25
Root Mean Squared Error: 3.207338547836586e-13
```

DECISION TREE REGRESSOR

In [40]:

```
#Creo la instancia necessària
regressor = DecisionTreeRegressor(random_state=0)
regressor.fit(x,y)
```

Out[40]:

```
▾ DecisionTreeRegressor
DecisionTreeRegressor(random_state=0)
```

In [41]:

```
prediccioTree = regressor.predict(x)
print(f"predicted response:\n{prediccioTree}")
```

```
predicted response:
[ 16.  99.  29. ... 110.  54.  18.]
```

In [42]:

```
# Comparació X-test amb la predicció
pred_Tree = pd.DataFrame({"Actual":y, "Predicted":prediccioTree})
pred_Tree

#No entenc perquè em surt igual
```

Out[42]:

	Actual	Predicted
0	16.0	16.0
1	99.0	99.0
2	29.0	29.0
3	32.0	32.0
4	23.0	23.0
...	...	...

	Actual	Predicted
99995	23.0	23.0
99996	26.0	26.0
99997	110.0	110.0
99998	54.0	54.0
99999	18.0	18.0

100000 rows × 2 columns

In [43]:
```
print("DECISION TREE REGRESSOR")
print('Mean Absolute Error:', metrics.mean_absolute_error(y, prediccioTree))
print('Mean Squared Error:', metrics.mean_squared_error(y, prediccioTree))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y, prediccioTree)))
```

```
DECISION TREE REGRESSOR
Mean Absolute Error: 0.0
Mean Squared Error: 0.0
Root Mean Squared Error: 0.0
```

NIVELL 2

Exercici 5

Realitza algun procés d'enginyeria de variables per millorar-ne la predicció

In [ ]:

NIVELL 3

Exercici 6

No utilitzis la variable DepDelay a l'hora de fer prediccions

In [ ]: