```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

```
df = pd.read_csv("airlines_delay.csv")
```

```
df
```

	Flight	Time	Length	Airline	AirportFrom	AirportTo	DayOfWeek	Class
0	2313.0	1296.0	141.0	DL	ATL	HOU	1	0
1	6948.0	360.0	146.0	OO	COS	ORD	4	0
2	1247.0	1170.0	143.0	B6	BOS	CLT	3	0
3	31.0	1410.0	344.0	US	OGG	PHX	6	0
4	563.0	692.0	98.0	FL	BMI	ATL	4	0
...	...	...	...	...	...	...	...	...
539377	6973.0	530.0	72.0	OO	GEG	SEA	5	1
539378	1264.0	560.0	115.0	WN	LAS	DEN	4	1
539379	5209.0	827.0	74.0	EV	CAE	ATL	2	1
539380	607.0	715.0	65.0	WN	BWI	BUF	4	1
539381	6377.0	770.0	55.0	OO	CPR	DEN	2	1

539382 rows × 8 columns

```
df.info()
```

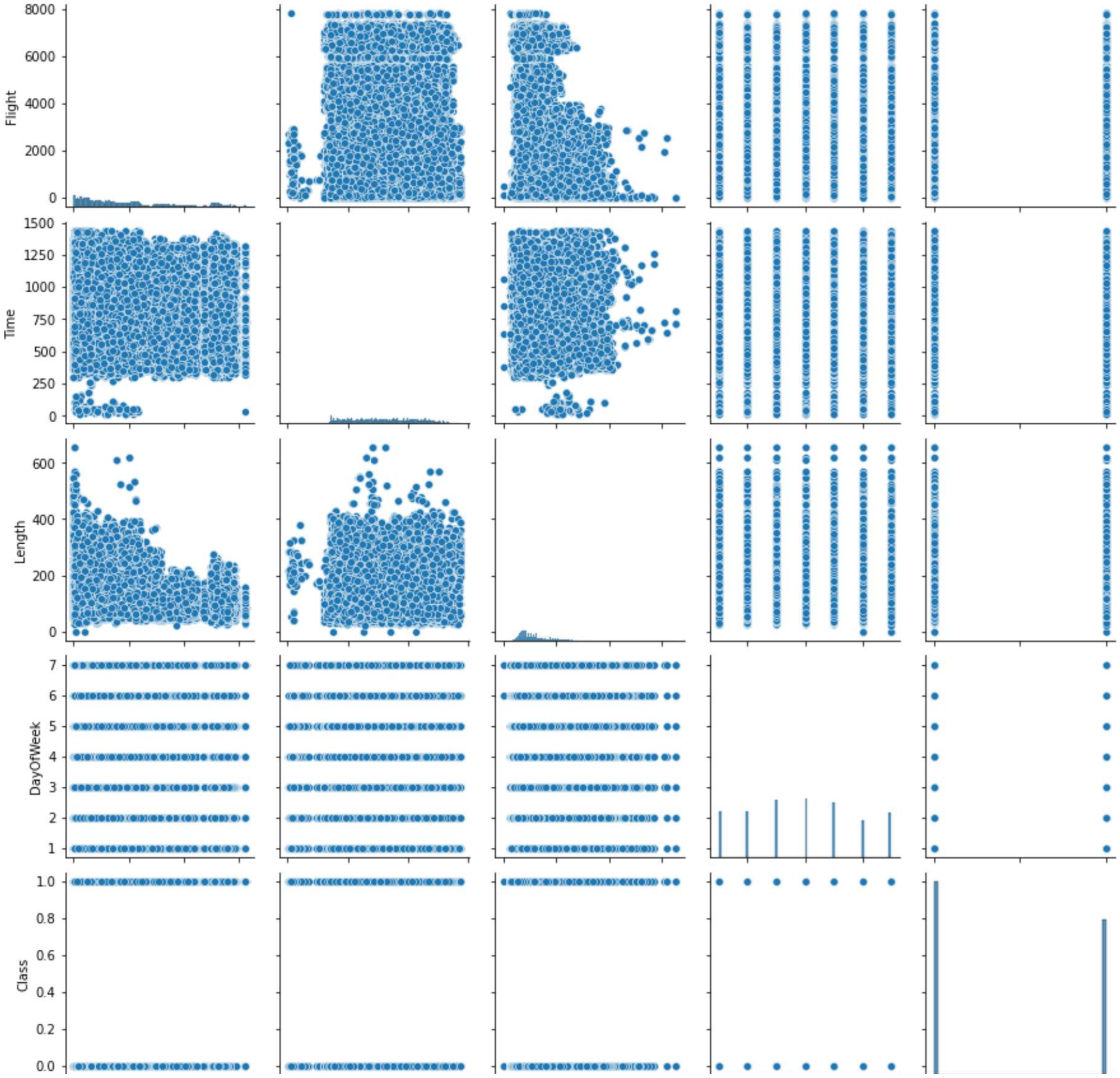
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 539382 entries, 0 to 539381
Data columns (total 8 columns):
 #   Column       Non-Null Count   Dtype  
---  ------       --------------   -----  
 0   Flight       539382 non-null  float64
 1   Time         539382 non-null  float64
 2   Length       539382 non-null  float64
 3   Airline      539382 non-null  object 
 4   AirportFrom  539382 non-null  object 
 5   AirportTo    539382 non-null  object 
 6   DayOfWeek    539382 non-null  int64  
 7   Class        539382 non-null  int64  
dtypes: float64(3), int64(2), object(3)
memory usage: 32.9+ MB
```

```
df.describe()
```

	Flight	Time	Length	DayOfWeek	Class
count	539382.000000	539382.000000	539382.000000	539382.000000	539382.000000
mean	2427.927988	802.728161	132.202104	3.929666	0.445443
std	2067.431700	278.045546	70.117045	1.914666	0.497015
min	1.000000	10.000000	0.000000	1.000000	0.000000
25%	712.000000	565.000000	81.000000	2.000000	0.000000
50%	1809.000000	795.000000	115.000000	4.000000	0.000000
75%	3745.000000	1035.000000	162.000000	5.000000	1.000000
max	7814.000000	1439.000000	655.000000	7.000000	1.000000

```
sns.pairplot(df)
```

```
<seaborn.axisgrid.PairGrid at 0x7f85484b6fa0>
```



```
df.Class.value_counts()
```

```
0    299118
1    240264
Name: Class, dtype: int64
```

```
df.AirportFrom.value_counts()
```

```
ATL    34449
ORD    24822
DFW    22153
DEN    19843
LAX    16657
       ...
MMH       16
SJT       15
GUM       10
ADK        9
ABR        2
Name: AirportFrom, Length: 293, dtype: int64
```

```
df.AirportTo.value_counts()
```

```
ATL    34440
ORD    24871
DFW    22153
DEN    19848
LAX    16656
       ...
MMH       16
SJT       15
```

```
GUM       10
ADK        9
ABR        2
Name: AirportTo, Length: 293, dtype: int64
```

```
from sklearn.preprocessing import OrdinalEncoder
```

```
oe = OrdinalEncoder()
```

```
df.iloc[:,3:6] = oe.fit_transform(df.iloc[:,3:6])
```

```
df.head()
```

	Flight	Time	Length	Airline	AirportFrom	AirportTo	DayOfWeek	Class
0	2313.0	1296.0	141.0	5.0	16.0	129.0	1	0
1	6948.0	360.0	146.0	12.0	65.0	208.0	4	0
2	1247.0	1170.0	143.0	3.0	35.0	60.0	3	0
3	31.0	1410.0	344.0	14.0	203.0	217.0	6	0
4	563.0	692.0	98.0	8.0	32.0	16.0	4	0

```
x = df.iloc[:,0:-1]
```

```
x
```

	Flight	Time	Length	Airline	AirportFrom	AirportTo	DayOfWeek
0	2313.0	1296.0	141.0	5.0	16.0	129.0	1
1	6948.0	360.0	146.0	12.0	65.0	208.0	4
2	1247.0	1170.0	143.0	3.0	35.0	60.0	3
3	31.0	1410.0	344.0	14.0	203.0	217.0	6
4	563.0	692.0	98.0	8.0	32.0	16.0	4
...	...	...	...	...	...	...	...
539377	6973.0	530.0	72.0	12.0	111.0	252.0	5
539378	1264.0	560.0	115.0	15.0	153.0	79.0	4
539379	5209.0	827.0	74.0	6.0	47.0	16.0	2
539380	607.0	715.0	65.0	15.0	45.0	43.0	4
539381	6377.0	770.0	55.0	12.0	67.0	79.0	2

539382 rows × 7 columns

```
y = df.iloc[:,-1]
```

```
y
```

```
0         0
1         0
2         0
3         0
4         0
         ..
539377    1
539378    1
539379    1
539380    1
539381    1
Name: Class, Length: 539382, dtype: int64
```

```
from sklearn.preprocessing import StandardScaler
```

```
sc = StandardScaler()
```

```
x = sc.fit_transform(x)
```

```
x
```

```
array([[-0.05558979,  1.77407013,  0.12547455, ..., -1.61396926,
        -0.1908345 , -1.53012021],
       [ 2.18632432, -1.59228795,  0.19678395, ..., -0.99668319,
         0.80471616,  0.03673449],
       [-0.57120585,  1.32090654,  0.15399831, ..., -1.37461344,
        -1.06036608, -0.48555041],
       ...,
       [ 1.3451833 ,  0.08729455, -0.83007146, ..., -1.22344134,
        -1.61484999, -1.00783531],
       [-0.88076896, -0.31551753, -0.95842839, ..., -1.24863669,
        -1.2745985 ,  0.03673449],
       [ 1.91013598, -0.11770802, -1.10104719, ..., -0.97148784,
        -0.82092985, -1.00783531]])
```

```
from sklearn.model_selection import train_test_split
```

```
xtrain,xtest,ytrain,ytest = train_test_split(x,y,test_size=0.3,random_state=1)
```

```
xtrain
```

```
array([[-0.61570555,  1.83161471, -0.8728571 , ..., -1.61396926,
         0.03599983,  1.08130429],
       [-0.9605782 ,  0.67352999, -0.0599299 , ...,  0.80478431,
        -1.53923855, -1.53012021],
       [-1.16904961,  1.23099313, -0.67319077, ...,  0.11191219,
         1.43481151, -0.48555041],
       ...,
       [ 2.01993415,  0.40378976, -0.18828683, ..., -0.92109714,
        -1.53923855,  0.03673449],
       [-0.39610921,  0.9612529 , -0.88711898, ...,  1.24570293,
         0.11161127,  1.60358918],
       [ 1.50238332, -1.32254772, -0.47352444, ..., -1.29902739,
        -1.61484999,  1.08130429]])
```

```
xtest
```

```
array([[-0.10105687,  0.22396294,  0.45349781, ...,  0.66620988,
        -1.58964618, -0.48555041],
       [-0.70712341, -1.16070358,  0.41071216, ...,  1.06933548,
        -1.06036608,  1.60358918],
       [-0.18570304,  0.59080965, -0.77302394, ...,  0.64101453,
        -1.0855699 ,  1.60358918],
       ...,
       [-0.45125014,  0.05492572,  0.36792652, ...,  0.35126801,
         0.36364941,  1.60358918],
       [-0.05607348,  0.85335681,  1.46609134, ...,  0.12450987,
         0.38885323,  1.08130429],
       [-0.31726736, -0.84061184,  0.6103785 , ..., -1.05967157,
        -0.80832794, -1.53012021]])
```

```
ytrain
```

```
186108    0
245254    0
369689    1
282188    0
120575    0
         ..
371403    1
491263    0
470924    1
491755    0
128037    0
Name: Class, Length: 377567, dtype: int64
```

```
ytest
```

```
21403     0
121631    0
535152    1
487001    0
69899     0
         ..
```

```
     186290    0
     91478     1
     304596    1
     483940    1
     328383    0
     Name: Class, Length: 161815, dtype: int64
```

```
from sklearn.linear_model import LogisticRegression
```

```
lr = LogisticRegression()
```

```
lr.fit(xtrain,ytrain)
```

```
     ▾ LogisticRegression
     LogisticRegression()
```

```
ypred = lr.predict(xtest)
```

```
from sklearn.metrics import classification_report
```

```
print(classification_report(ytest,ypred))
```

```
                   precision    recall  f1-score   support

                0       0.60      0.77      0.67     89585
                1       0.55      0.36      0.43     72230

         accuracy                           0.58    161815
        macro avg       0.57      0.56      0.55    161815
     weighted avg       0.58      0.58      0.56    161815
```

```
from sklearn.ensemble import RandomForestClassifier
```

```
rfc = RandomForestClassifier(n_estimators=45,min_samples_leaf=4)
rfc.fit(xtrain,ytrain)
ypred1 = rfc.predict(xtest)
```

```
print(classification_report(ytest,ypred1))
```

```
                   precision    recall  f1-score   support

                0       0.67      0.74      0.71     89585
                1       0.63      0.56      0.59     72230

         accuracy                           0.66    161815
        macro avg       0.65      0.65      0.65    161815
     weighted avg       0.66      0.66      0.66    161815
```

```
import tensorflow as tf
from tensorflow.keras import Sequential
from tensorflow.keras.layers import Dense
```

```
from tensorflow.keras.callbacks import EarlyStopping
```

```
es = EarlyStopping(monitor = "val_loss", mode = "min", verbose = 1, patience=20)
```

```
d = Sequential()
```

```
d.add(Dense(units=25,activation="relu"))
```

```
d.add(Dense(units=25,activation="relu"))
```

```
d.add(Dense(units=22,activation="relu"))
```

```
d.add(Dense(units=1,activation="sigmoid"))
```

```
d.compile(optimizer = 'adam',loss = "binary_crossentropy", metrics = ["accuracy"])
```

```
d.fit(xtrain,ytrain,epochs=200,validation_data=(xtest,ytest),verbose=1,batch_size=100,callbacks=[es])
```

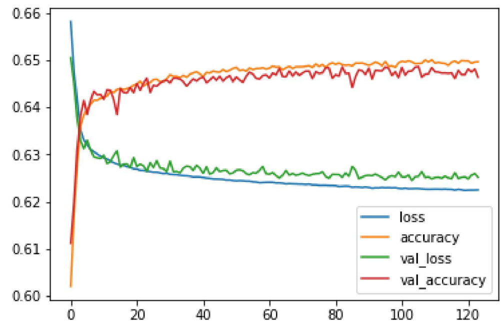
```
3776/3776 [==============================] - 13s 3ms/step - loss: 0.6230 - accuracy: 0.6487 - val_loss: 0.6252 - val_accuracy: 0.6483
Epoch 98/200
3776/3776 [==============================] - 13s 3ms/step - loss: 0.6229 - accuracy: 0.6488 - val_loss: 0.6255 - val_accuracy: 0.6468
Epoch 99/200
3776/3776 [==============================] - 13s 3ms/step - loss: 0.6229 - accuracy: 0.6485 - val_loss: 0.6253 - val_accuracy: 0.6476
Epoch 100/200
3776/3776 [==============================] - 13s 3ms/step - loss: 0.6228 - accuracy: 0.6494 - val_loss: 0.6263 - val_accuracy: 0.6463
Epoch 101/200
3776/3776 [==============================] - 12s 3ms/step - loss: 0.6227 - accuracy: 0.6498 - val_loss: 0.6252 - val_accuracy: 0.6485
Epoch 102/200
3776/3776 [==============================] - 13s 3ms/step - loss: 0.6228 - accuracy: 0.6496 - val_loss: 0.6252 - val_accuracy: 0.6469
Epoch 103/200
3776/3776 [==============================] - 13s 3ms/step - loss: 0.6227 - accuracy: 0.6499 - val_loss: 0.6256 - val_accuracy: 0.6477
Epoch 104/200
3776/3776 [==============================] - 13s 3ms/step - loss: 0.6227 - accuracy: 0.6497 - val_loss: 0.6245 - val_accuracy: 0.6477
Epoch 105/200
3776/3776 [==============================] - 13s 3ms/step - loss: 0.6228 - accuracy: 0.6493 - val_loss: 0.6252 - val_accuracy: 0.6485
Epoch 106/200
3776/3776 [==============================] - 12s 3ms/step - loss: 0.6227 - accuracy: 0.6493 - val_loss: 0.6254 - val_accuracy: 0.6487
Epoch 107/200
3776/3776 [==============================] - 13s 3ms/step - loss: 0.6227 - accuracy: 0.6494 - val_loss: 0.6264 - val_accuracy: 0.6464
Epoch 108/200
3776/3776 [==============================] - 13s 3ms/step - loss: 0.6226 - accuracy: 0.6501 - val_loss: 0.6251 - val_accuracy: 0.6471
Epoch 109/200
3776/3776 [==============================] - 13s 3ms/step - loss: 0.6226 - accuracy: 0.6496 - val_loss: 0.6252 - val_accuracy: 0.6471
Epoch 110/200
3776/3776 [==============================] - 12s 3ms/step - loss: 0.6226 - accuracy: 0.6501 - val_loss: 0.6248 - val_accuracy: 0.6484
Epoch 111/200
3776/3776 [==============================] - 13s 3ms/step - loss: 0.6226 - accuracy: 0.6494 - val_loss: 0.6251 - val_accuracy: 0.6475
Epoch 112/200
3776/3776 [==============================] - 13s 3ms/step - loss: 0.6226 - accuracy: 0.6496 - val_loss: 0.6252 - val_accuracy: 0.6475
Epoch 113/200
3776/3776 [==============================] - 13s 3ms/step - loss: 0.6226 - accuracy: 0.6494 - val_loss: 0.6249 - val_accuracy: 0.6473
Epoch 114/200
3776/3776 [==============================] - 13s 3ms/step - loss: 0.6226 - accuracy: 0.6493 - val_loss: 0.6254 - val_accuracy: 0.6473
Epoch 115/200
3776/3776 [==============================] - 13s 3ms/step - loss: 0.6226 - accuracy: 0.6489 - val_loss: 0.6250 - val_accuracy: 0.6472
Epoch 116/200
3776/3776 [==============================] - 13s 3ms/step - loss: 0.6226 - accuracy: 0.6496 - val_loss: 0.6249 - val_accuracy: 0.6473
Epoch 117/200
3776/3776 [==============================] - 13s 3ms/step - loss: 0.6224 - accuracy: 0.6499 - val_loss: 0.6261 - val_accuracy: 0.6461
Epoch 118/200
3776/3776 [==============================] - 13s 3ms/step - loss: 0.6226 - accuracy: 0.6494 - val_loss: 0.6249 - val_accuracy: 0.6479
Epoch 119/200
3776/3776 [==============================] - 12s 3ms/step - loss: 0.6225 - accuracy: 0.6500 - val_loss: 0.6251 - val_accuracy: 0.6470
Epoch 120/200
3776/3776 [==============================] - 12s 3ms/step - loss: 0.6224 - accuracy: 0.6499 - val_loss: 0.6252 - val_accuracy: 0.6469
Epoch 121/200
3776/3776 [==============================] - 12s 3ms/step - loss: 0.6224 - accuracy: 0.6499 - val_loss: 0.6248 - val_accuracy: 0.6482
Epoch 122/200
3776/3776 [==============================] - 12s 3ms/step - loss: 0.6224 - accuracy: 0.6494 - val_loss: 0.6256 - val_accuracy: 0.6475
Epoch 123/200
3776/3776 [==============================] - 12s 3ms/step - loss: 0.6224 - accuracy: 0.6496 - val_loss: 0.6259 - val_accuracy: 0.6482
Epoch 124/200
3776/3776 [==============================] - 13s 3ms/step - loss: 0.6225 - accuracy: 0.6497 - val_loss: 0.6252 - val_accuracy: 0.6464
Epoch 124: early stopping
<keras.callbacks.History at 0x7f85369740d0>
```

```
d.history.history
```

```
 0.6466582417488098,
 0.6486234068870544,
 0.6471835374832153,
 0.6472638249397278,
 0.6442418694496155,
 0.6465593576431274,
 0.6479806900024414,
 0.6477644443511963,
 0.646979570388794,
 0.648308277130127,
 0.6460093259811401,
 0.6478818655014038,
 0.6478015184402466,
 0.6477644443511963,
 0.6485739946365356,
 0.6482958793640137,
 0.6468003392219543,
 0.6475728750228882,
 0.6463121175765991,
 0.6484936475753784,
 0.6468559503555298,
 0.6476778984069824,
 0.6477397084236145,
 0.6485307216644287,
 0.6486666798591614,
 0.6463924646377563,
 0.6470599174499512,
 0.647084653377533,
 0.6484256982803345,
 0.6475481390953064,
 0.6475048661231995,
 0.6472885608673096,
 0.6473132967948914,
 0.6472391486167908,
 0.6473256349563599,
 0.6461390852928162,
 0.6478633284568787,
 0.6469733715057373,
 0.6469486951828003,
 0.6481537818908691,
 0.6475234031677246,
 0.6481661200523376,
 0.6464172005653381]}
```

```
l=pd.DataFrame(d.history.history)
l.plot()
```

```
<Axes: >
```



```
ypred2 = d.predict(xtest)
```

```
5057/5057 [==============================] - 8s 2ms/step
```

```
ypred2 = ypred2>0.5
```

```
print(classification_report(ytest,ypred2))
```

```
              precision    recall  f1-score   support

           0       0.65      0.78      0.71     89585
           1       0.64      0.48      0.55     72230

    accuracy                           0.65    161815
   macro avg       0.64      0.63      0.63    161815
weighted avg       0.65      0.65      0.64    161815
```

✓ 0s completed at 2:13 PM ● ×