**PROJECT DEVELOPMENT PHASE**

**Sprint - III**

|  |  |
| --- | --- |
| **Date** | 07-Nov-2022 |
| **Team ID** | PNT2022TMID12521 |
| **Project Name** | Developing a Flight Delay Model Using Machine Learning |
| **Maximum Marks** | 8 Marks |

[Training the model on IBM](https://dataplatform.cloud.ibm.com/analytics/notebooks/v2/03f09323-7f6f-44c6-bd07-29fc3e422d2a/view?access_token=6d1c6a19cce2474972b50202b0f2ce18ebd84c2d5300058ddeba48488efc8473)

# Import libraries

**import** numpy **as** np

**import** pandas **as** pd

# Import label encoder

**from** sklearn.preprocessing **import** LabelEncoder **from** sklearn.ensemble **import** RandomForestClassifier **from** sklearn.metrics **import** classification\_report **from** sklearn.metrics **import** jaccard\_score

**from** sklearn.model\_selection **import** train\_test\_split

# Import dataset

import os, types import pandas as pd

from botocore.client import Config import ibm\_boto3

def iter (self): return 0

@hidden\_cell

The following code accesses a file in your IBM Cloud Object Storage

You might want to remove those credentials before you share the notebook

cos\_client **=** ibm\_boto3**.**client(service\_name**=**'s3', ibm\_api\_key\_id**=**'BmleA4MV5fW02WAmF6zCBnBmBBkh7otufBwtC7V84yVO', ibm\_auth\_endpoint**=**"https://iam.cloud.ibm.com/oidc/token", config**=**Config(signature\_version**=**'oauth'), endpoint\_url**=**'https://s3.private.us.cloud-object-storage.appdomain.cloud')

bucket **=** 'randommodel-donotdelete-pr-jpkful51t7p3nj' object\_key **=** 'Processed\_data15.csv'

body **=** cos\_client**.**get\_object(Bucket**=**bucket,Key**=**object\_key)['Body']

Add missing iter method, so pandas accepts body as file-like object

if not hasattr(body, " iter "): body. iter = types.MethodType( iter , body )

df **=** pd**.**read\_csv(body) df**.**head()

df**.**head(90)

columns**=** ['carrier','dest', 'origin'] le**=**LabelEncoder()

for i in columns: df[i]=le.fit\_transform(df[i])

df['carrier']**.**unique()

df['origin']**.**unique()

df['dest']**.**unique() df**.**head(90)

# From column(years) to column(distance)

X **=** df**.**iloc[:, 0:6]**.**values X[0:5]

y **=** df['delayed'] y**.**head()**.**to\_frame()

**for** i **in** range(0, 20):

X\_train, X\_test, y\_train, y\_test **=** train\_test\_split(X, y, test\_size**=**0.25, random\_state**=**i)

# Creating random forest classifier

clf **=** RandomForestClassifier(random\_state**=**i) clf**.**fit(X\_train, y\_train)

# Determining the score

train\_score **=** clf**.**score(X\_train, y\_train) test\_score **=** clf**.**score(X\_test, y\_test)

print("Test: {}, Train: {} and Random State: {}"**.**format(test\_score, train\_score, i))

X\_train, X\_test, y\_train, y\_test **=** train\_test\_split(X, y, test\_size**=**0.25, random\_state**=**18) clf **=** RandomForestClassifier(random\_state**=**18)

clf**.**fit(X\_train, y\_train)

print("Train set: ", clf**.**score(X\_train, y\_train)) print("Test set: ", clf**.**score(X\_test, y\_test))

# Predicting the trained Classifier to the test

yhat **=** clf**.**predict(X\_test)

# Viewing the predicted probabilities of first 10 observations

yhat\_prob **=** clf**.**predict\_proba(X\_test)[:10] print(classification\_report(y\_test, yhat)) **import** joblib

joblib**.**dump(clf, 'classifier.pkl')

**!**pip install -U ibm-watson-machine-learning

**from** ibm\_watson\_machine\_learning **import** APIClient

**import** json

**import** numpy **as** np

wml\_credentials **=** { "apikey":"gyOvc0l0Hde4zdTmNc47N4Vh1zmMTFh7FlK8BEcKPADB", "url": "https://us-south.ml.cloud.ibm.com" }

wml\_client **=** APIClient(wml\_credentials) wml\_client**.**spaces**.**list()

SPACE\_ID **=** "7c5663ee-671c-49d2-a415-a27bac157d6d"

wml\_client**.**set**.**default\_space(SPACE\_ID) wml\_client**.**software\_specifications**.**list(500)

# Save and Deploy the model

**import** sklearn sklearn**.** version

MODEL\_NAME **=** 'Flight' DEPLOYMENT\_NAME **=** 'model\_deploy' DEMO\_MODEL **=** clf

# Set Python Version

software\_spec\_uid **=** wml\_client**.**software\_specifications**.**get\_id\_by\_name('runtime-22.1-py3.9')

# Setup model meta

model\_props **=** {

wml\_client**.**repository**.**ModelMetaNames**.**NAME: MODEL\_NAME, wml\_client**.**repository**.**ModelMetaNames**.**TYPE: 'scikit-learn\_1.0', wml\_client**.**repository**.**ModelMetaNames**.**SOFTWARE\_SPEC\_UID: software\_spec\_uid

}

# Save model

model\_details **=** wml\_client**.**repository**.**store\_model( model**=**DEMO\_MODEL, meta\_props**=**model\_props, training\_data**=**X\_train,

training\_target**=**y\_train

)

model\_details

model\_id **=** wml\_client**.**repository**.**get\_model\_id(model\_details)

model\_id

# Set meta

deployment\_props **=** { wml\_client**.**deployments**.**ConfigurationMetaNames**.**NAME:DEPLOYMENT\_NAME, wml\_client**.**deployments**.**ConfigurationMetaNames**.**ONLINE: {}

}

# Deploy

deployment **=** wml\_client**.**deployments**.**create( artifact\_uid**=**model\_id, meta\_props**=**deployment\_props

)