

SPRINT – 2 PROJECT DOCUMENT

Date	13 November 2022
Team ID	PNT2022TMID15779
Project Name	Flight Delay Prediction Using Machine Learning

DEVELOPMENT PHASE:

SPRINT-2:

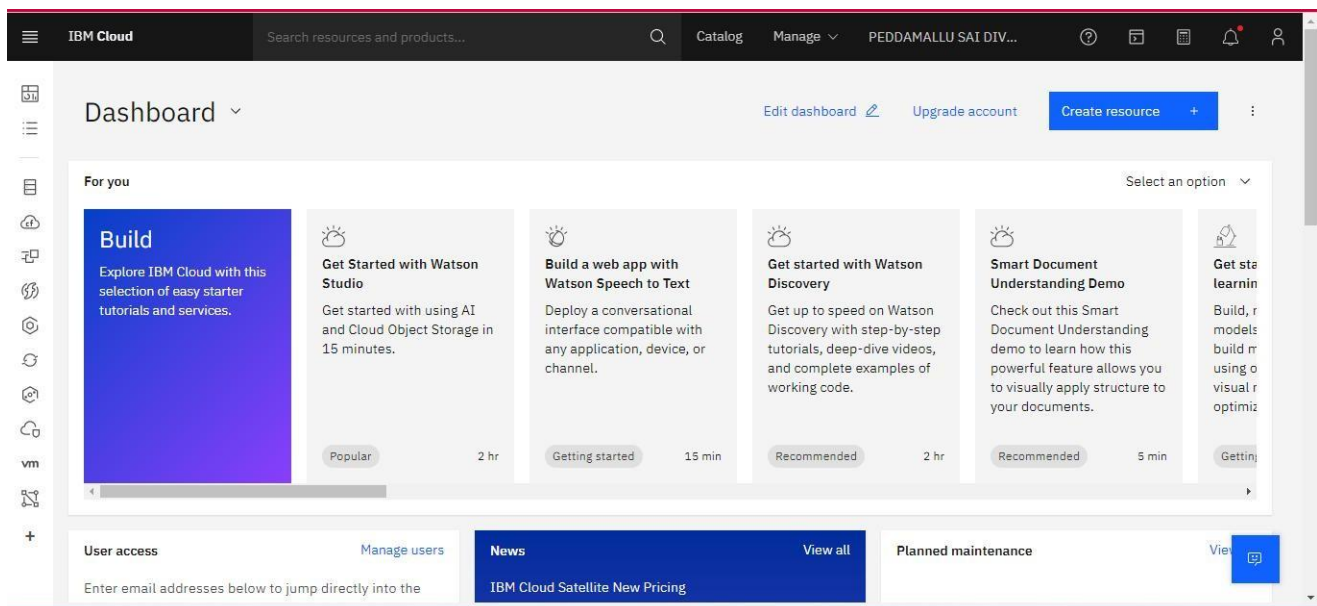
- Creating IBM cloud account & Required Resources
- Importing jupyter notebook file in ibm watson
- Deploy our model in IBM Watson
- Predict the result

Creating IBM cloud account & Required Resources:

Creating IBM cloud account:

Frist, need to create IBM Cloud account by using SI Mail Id and SI Password which is provided by IBM in profile.

Below dashboard of an account after created,



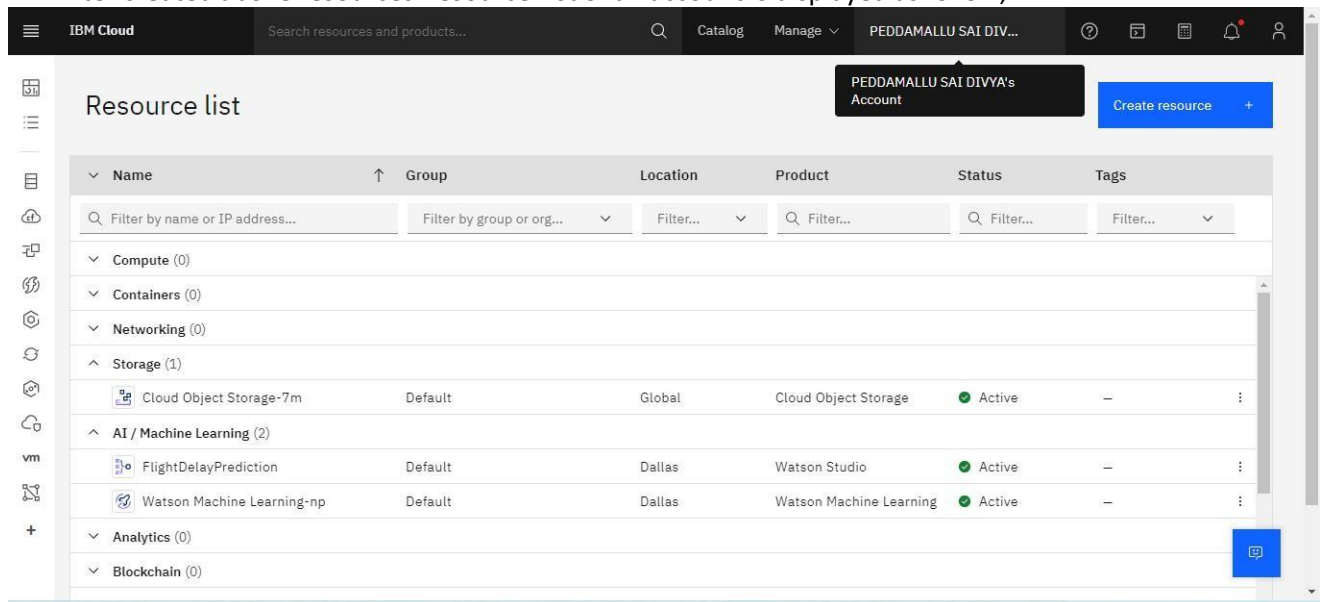
Creating IBM Cloud Required Resources:

After creating IBM cloud account, to deploy ML model, need to create following resources such as,
Cloud Object Storage

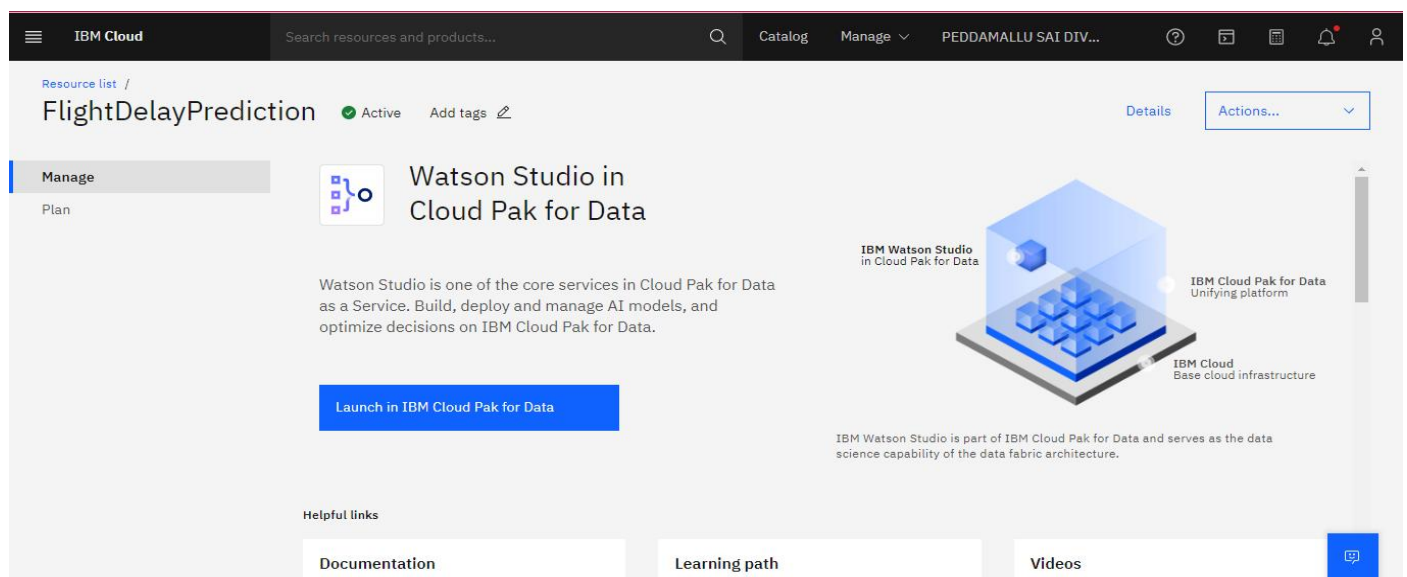
Watson Machine Learning

Watson Studio

After created above resources Resource List of an account is displayed as follow,



Name	Group	Location	Product	Status	Tags
Filter by name or IP address... Filter by group or org... Filter... Filter... Filter... Filter...					
Compute (0)					
Containers (0)					
Networking (0)					
Storage (1)					
Cloud Object Storage-7m	Default	Global	Cloud Object Storage	Active	-
AI / Machine Learning (2)					
FlightDelayPrediction	Default	Dallas	Watson Studio	Active	-
Watson Machine Learning-np	Default	Dallas	Watson Machine Learning	Active	-
Analytics (0)					
Blockchain (0)					



FlightDelayPrediction Active [Add tags](#) [Details](#) [Actions...](#)

Manage

Plan

Watson Studio in Cloud Pak for Data

Watson Studio is one of the core services in Cloud Pak for Data as a Service. Build, deploy and manage AI models, and optimize decisions on IBM Cloud Pak for Data.

[Launch in IBM Cloud Pak for Data](#)

Helpful links

[Documentation](#) [Learning path](#) [Videos](#)

IBM Watson Studio in Cloud Pak for Data

IBM Cloud Pak for Data Unifying platform

IBM Cloud Base cloud infrastructure

IBM Watson Studio is part of IBM Cloud Pak for Data and serves as the data science capability of the data fabric architecture.

All the resource are in active state.

All the required cloud resources are created successfully.

Import .ipynb file of sprint-1 which ML models are build in Jupyter notebook.

IBM Watson Studio interface showing the 'Assets' tab for a project named 'Flight Delay'. The interface includes a sidebar with 'All assets' and 'Asset types' (Data, Notebooks). The main area lists three assets: 'Use AutoAI and Lale to predict credit...', 'FlightDelay', and 'flightdata.csv'. A 'Data in this project' panel on the right shows a drop zone for data files.

Import Required Libraries

The first step is usually importing the libraries that will be needed in the program.

```
In [84]: !pip install imbalanced-learn
!pip install imblearn

Requirement already satisfied: imblearn in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (0.0)
Requirement already satisfied: imbalanced-learn in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from imblearn) (0.9.1)
Requirement already satisfied: threadpoolctl>=2.0.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from imbalanced-learn->imblearn) (2.2.0)
Requirement already satisfied: joblib>=1.0.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from imbalanced-learn->imblearn) (1.2.0)
Requirement already satisfied: scipy>=1.3.2 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from imbalanced-learn->imblearn) (1.7.3)
Requirement already satisfied: numpy>=1.17.3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from imbalanced-learn->imblearn) (1.20.3)
Requirement already satisfied: scikit-learn>=1.1.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from imbalanced-learn->imblearn) (1.1.3)

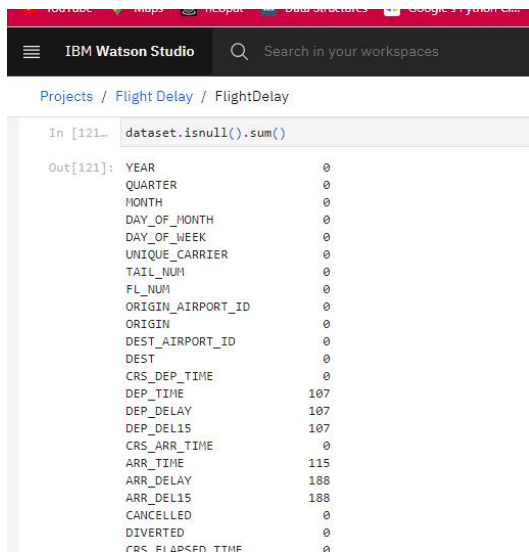
In [116]: import sys
import numpy as np
import pandas as pd
import seaborn as sns
import pickle
import sklearn
%matplotlib inline
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import OneHotEncoder
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
import sklearn.metrics as metrics
from matplotlib import pyplot as plt
import imblearn
```

Importing The Dataset and Analyze The Data set

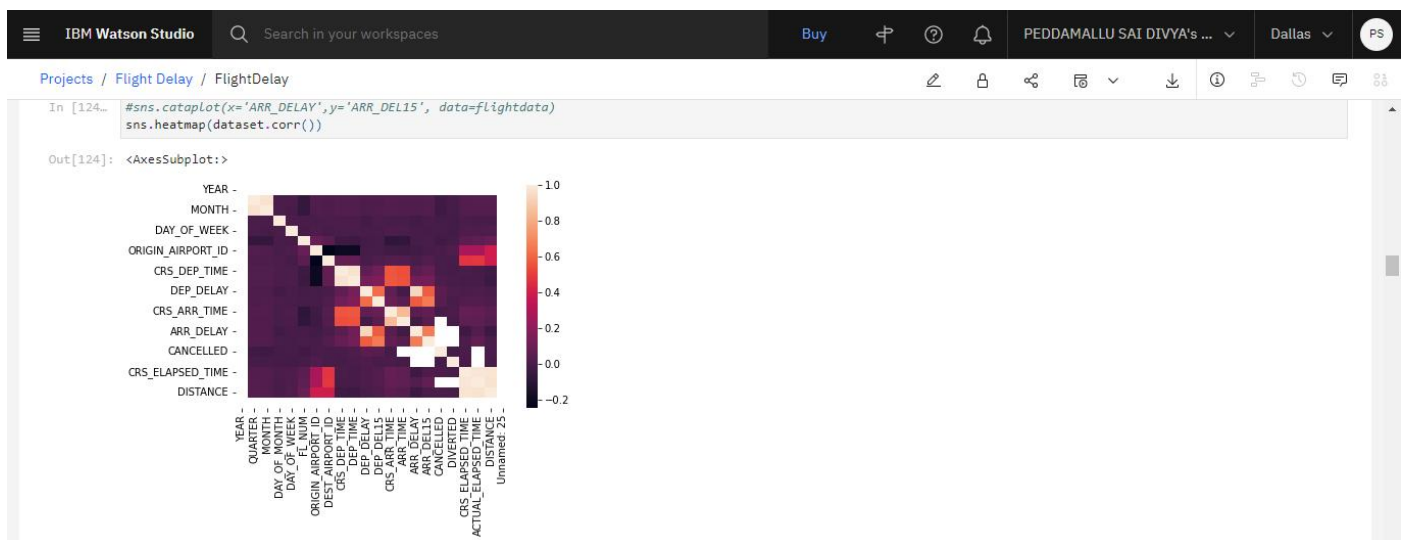
```
In [119]: dataset.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11231 entries, 0 to 11230
Data columns (total 26 columns):
#   Column                Non-Null Count  Dtype
---  ---                ---
0   YEAR                  11231 non-null  int64
1   QUARTER                11231 non-null  int64
2   MONTH                 11231 non-null  int64
3   DAY_OF_MONTH           11231 non-null  int64
4   DAY_OF_WEEK            11231 non-null  int64
5   UNIQUE_CARRIER        11231 non-null  object
6   TAIL_NUM               11231 non-null  object
7   FL_NUM                 11231 non-null  int64
8   ORIGIN_AIRPORT_ID      11231 non-null  int64
9   ORIGIN                 11231 non-null  object
10  DEST_AIRPORT_ID        11231 non-null  int64
11  DEST                   11231 non-null  object
12  CRS_DEP_TIME            11231 non-null  int64
13  DEP_TIME                11124 non-null  float64
14  NFP_DELAY               11124 non-null  float64
```

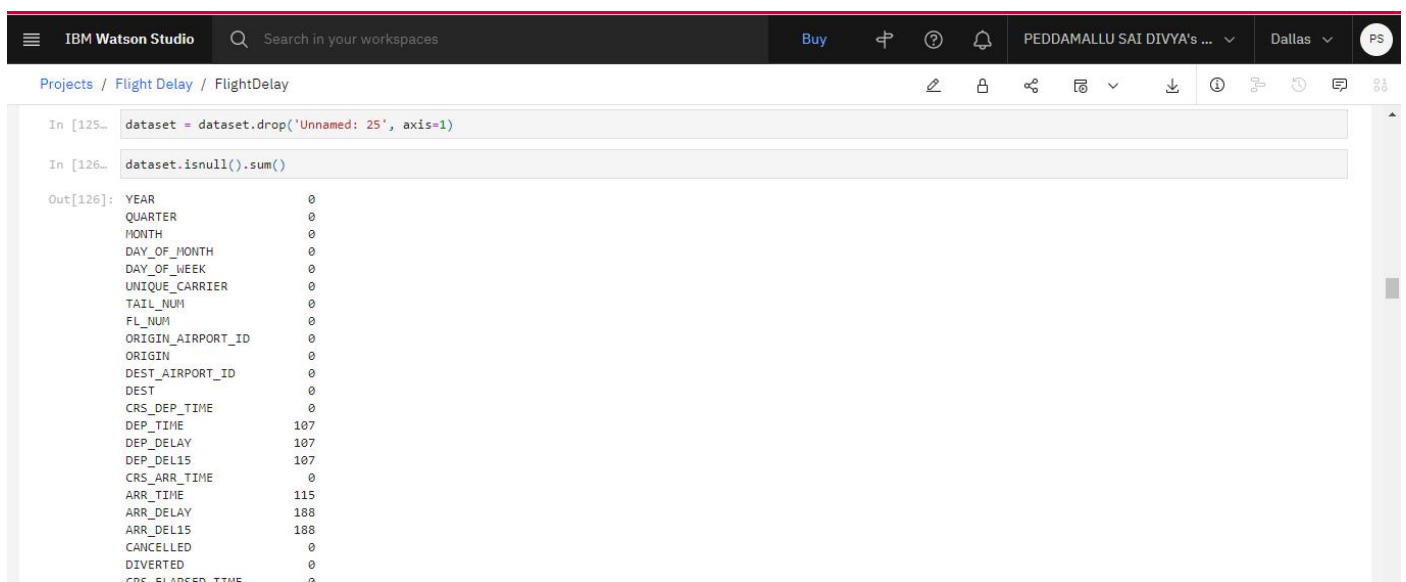
Handling Missing Values



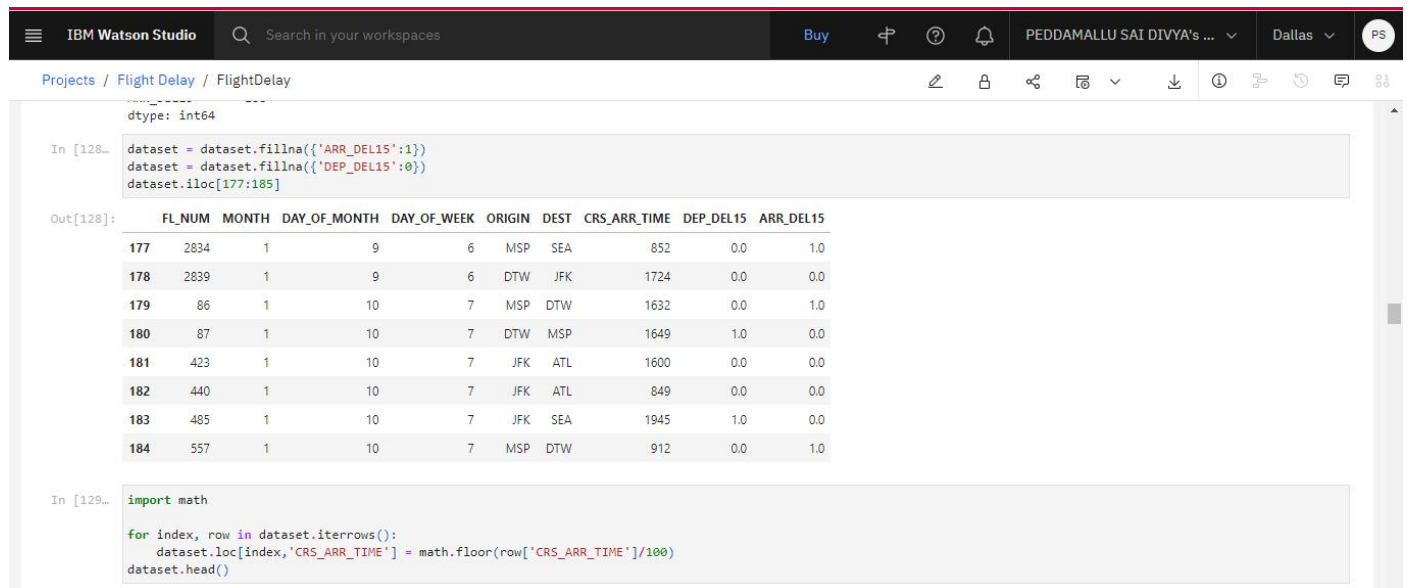
Data Visualization



Dropping Un-Necessary Columns



Label Encoding & One Hot Encoding



IBM Watson Studio interface showing a Jupyter notebook for Flight Delay data. The notebook includes code for filling missing values and a preview of the dataset.

```
In [128]: dataset = dataset.fillna({'ARR_DEL15':1})
dataset = dataset.fillna({'DEP_DEL15':0})
dataset.iloc[177:185]
```

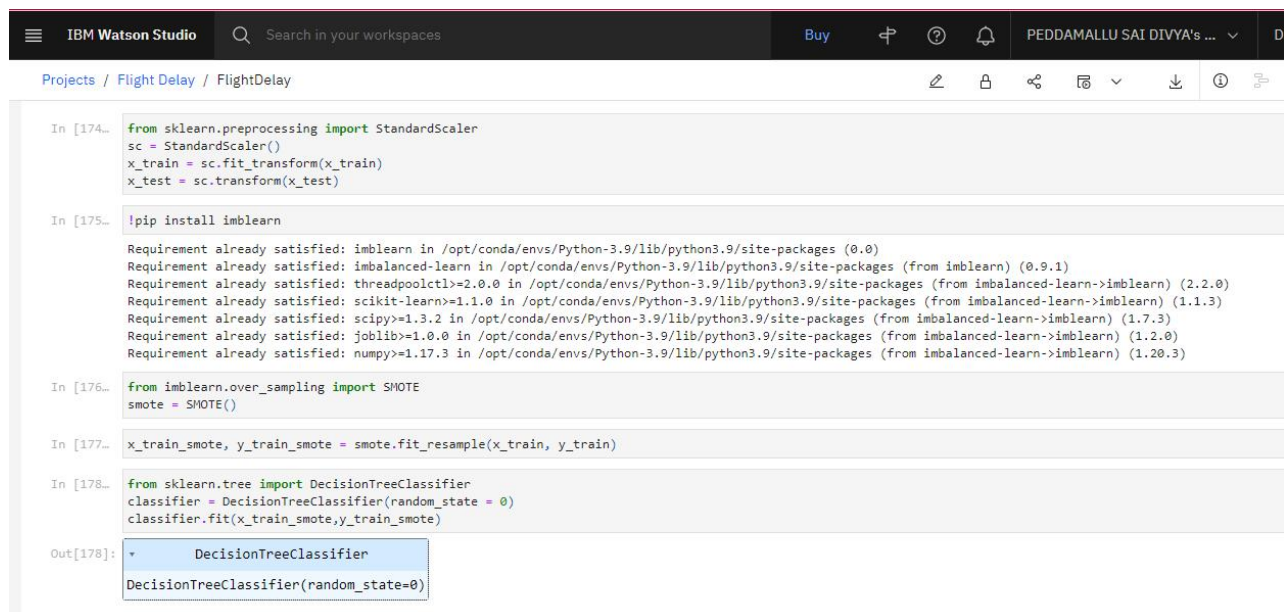
Out[128]:

	FL_NUM	MONTH	DAY_OF_MONTH	DAY_OF_WEEK	ORIGIN	DEST	CRS_ARR_TIME	DEP_DEL15	ARR_DEL15
177	2834	1	9	6	MSP	SEA	852	0.0	1.0
178	2839	1	9	6	DTW	JFK	1724	0.0	0.0
179	86	1	10	7	MSP	DTW	1632	0.0	1.0
180	87	1	10	7	DTW	MSP	1649	1.0	0.0
181	423	1	10	7	JFK	ATL	1600	0.0	0.0
182	440	1	10	7	JFK	ATL	849	0.0	0.0
183	485	1	10	7	JFK	SEA	1945	1.0	0.0
184	557	1	10	7	MSP	DTW	912	0.0	1.0

```
In [129]: import math

for index, row in dataset.iterrows():
    dataset.loc[index, 'CRS_ARR_TIME'] = math.floor(row['CRS_ARR_TIME']/100)
dataset.head()
```

Splitting The Dataset Into Dependent And Independent Variables



IBM Watson Studio interface showing a Jupyter notebook for splitting the dataset. The notebook includes code for scaling, installing imblearn, and using SMOTE for oversampling.

```
In [174]: from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
x_train = sc.fit_transform(x_train)
x_test = sc.transform(x_test)
```

```
In [175]: !pip install imblearn

Requirement already satisfied: imblearn in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (0.0)
Requirement already satisfied: imbalanced-learn in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from imblearn) (0.9.1)
Requirement already satisfied: threadpoolctl>=2.0.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from imbalanced-learn->imblearn) (2.2.0)
Requirement already satisfied: scikit-learn>=1.1.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from imbalanced-learn->imblearn) (1.1.3)
Requirement already satisfied: scipy>=1.3.2 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from imbalanced-learn->imblearn) (1.7.3)
Requirement already satisfied: joblib>=1.0.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from imbalanced-learn->imblearn) (1.2.0)
Requirement already satisfied: numpy>=1.17.3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from imbalanced-learn->imblearn) (1.20.3)
```

```
In [176]: from imblearn.over_sampling import SMOTE
smote = SMOTE()
```

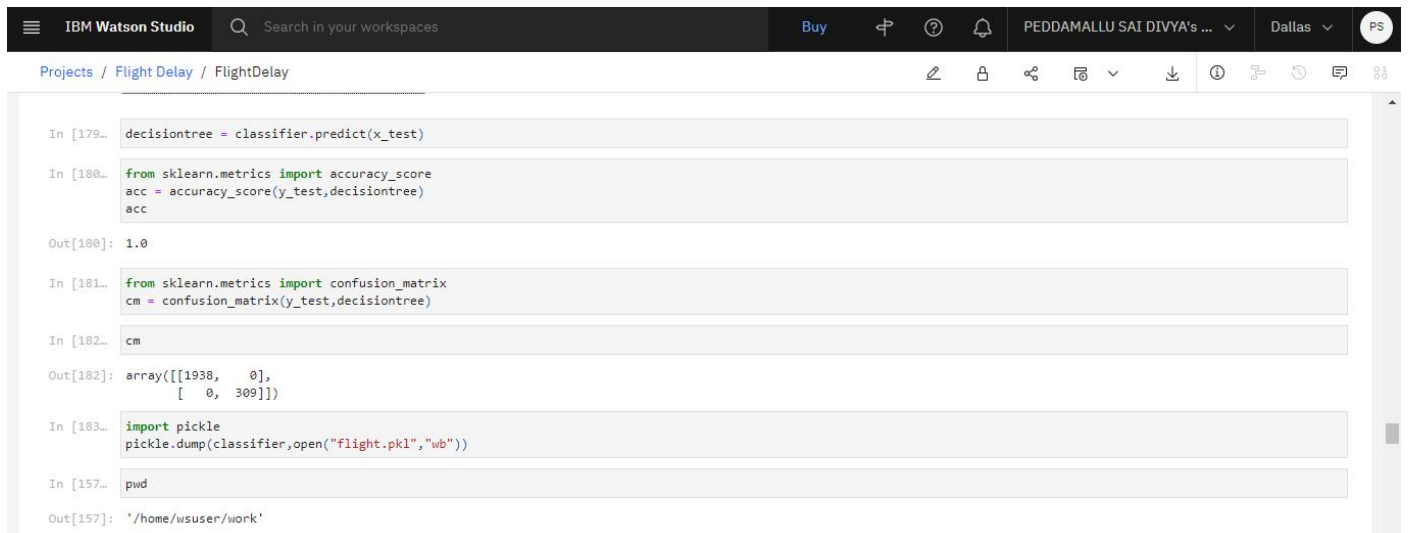
```
In [177]: x_train_smote, y_train_smote = smote.fit_resample(x_train, y_train)
```

```
In [178]: from sklearn.tree import DecisionTreeClassifier
classifier = DecisionTreeClassifier(random_state = 0)
classifier.fit(x_train_smote, y_train_smote)
```

Out[178]:

```
DecisionTreeClassifier
DecisionTreeClassifier(random_state=0)
```

Splitting The Dataset Into Dependent And Independent Variables



```
In [179]: decisiontree = classifier.predict(x_test)

In [180]: from sklearn.metrics import accuracy_score
acc = accuracy_score(y_test,decisiontree)
acc

Out[180]: 1.0

In [181]: from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test,decisiontree)

In [182]: cm

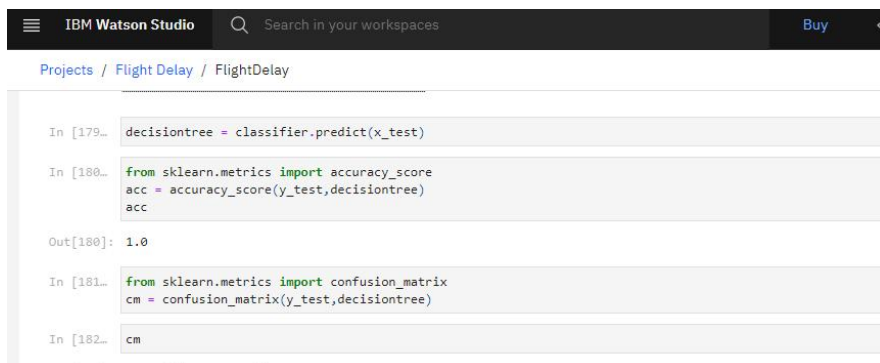
Out[182]: array([[1938,  0],
               [  0, 309]])

In [183]: import pickle
pickle.dump(classifier,open("flight.pkl","wb"))

In [157]: pwd

Out[157]: '/home/wsuser/work'
```

Train And Test The Model Using Decision Tree Classifier



```
In [179]: decisiontree = classifier.predict(x_test)

In [180]: from sklearn.metrics import accuracy_score
acc = accuracy_score(y_test,decisiontree)
acc

Out[180]: 1.0

In [181]: from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test,decisiontree)

In [182]: cm
```

Model Evaluation



```
In [180]: from sklearn.metrics import accuracy_score
acc = accuracy_score(y_test,decisiontree)
acc

Out[180]: 1.0

In [181]: from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test,decisiontree)

In [182]: cm
```

Deploy our model in IBM Watson:

To deploy ML model in IBM cloud, need to create project in IBM Watson. After successful creation of project import .ipynb file of sprint-1 which ML models are build in Jupyter notebook.

Upload required datasets and import it.

Deploy model using following code,

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

Authenticate and set Space

```
from ibm_watson_machine_learning import APIClient
import json
import numpy as np
wml_cred={
    "apikey":"okbr7ARn0QjyplTOyyNFC2QVkfCF6q7afpci065Hucby8",
    "url":"https://us-south.ml.cloud.ibm.com"
}
wml_clients=APIClient(wml_cred)
wml_clients.spaces.list()
space_id="6d7c1218-3aca-4256-be3d-d610732530b1"
```

```

'name': 'flight',
'owner': 'IBMid-6630042GII',
'resource_key': '1185cdbf-9975-49a4-ac41-7f372c902d5f',
'space_id': 'b6783a9b-4513-4465-b052-a15d738cf69d',
'system': {'warnings': []}}

```

In [171]:

```

model_uid = wml_client.repository.get_model_id(model_details)
model_uid

```

Out[171]:

```

'b124f04c-391d-4bea-8bb5-6c9d9e875765'

```

In [172]:

```

deployment_props={
    wml_client.deployments.ConfigurationMetaNames.NAME:DEPLOYMENT_NAME,
    wml_client.deployments.ConfigurationMetaNames.ONLINE: {}
}

```

In [173]:

```

deployment = wml_client.deployments.create(
    artifact_uid=model_uid,
    meta_props = deployment_props)

```

#####

Synchronous deployment creation for uid: 'b124f04c-391d-4bea-8bb5-6c9d9e875765' started

#####

```
wml_clients.set.default_space(space_id)
wml_clients.software_specifications.list(500)
MODEL_NAME="randomforest"
DEPLOYMENT_NAME="rf_deployment"
DEMO_MODEL=rf
soft_sepc_id=wml_clients.software_specifications.get_id_by_name("runtime-22.1-py3.9")
```

In [115]:

```
model_props={ wml_clients.repository.ModelMetaNames.NAME:MODEL_NAME,
               wml_clients.repository.ModelMetaNames.TYPE:"scikit-learn_1.0",
               wml_clients.repository.ModelMetaNames.SOFTWARE_SPEC_UID: soft_sepc_id
}


```

In [116]:

```
model_details=wml_clients.repository.store_model(model=DEMO_MODEL,meta_props=model_props,training_data=x_train,
                                                  training_target=y_train.values.ravel())
```

In [117]:

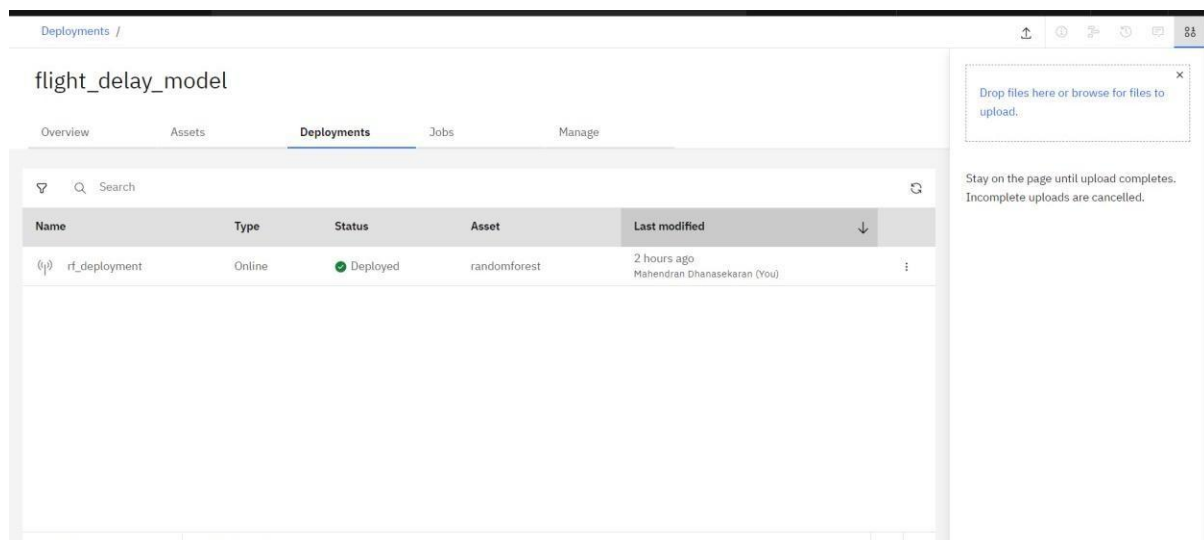
```
model_details
model_id=wml_clients.repository.get_model_id(model_details)
dep_props={
    wml_clients.deployments.ConfigurationMetaNames.NAME:DEPLOYMENT_NAME,
    wml_clients.deployments.ConfigurationMetaNames.ONLINE:{}
}
```

In [125]:

```
deployment=wml_clients.deployments.create(artifact_uid=model_id,meta_props=dep_props)
```

NOTE: APIKey must need to create to deploy and connect API

After successful of deployment, deployed is appeared in Deployment section as follow,



Testing of deployed model as follow, by giving values of all the features and it gives prediction.

Deployments / flight_delay_model / randomforest /

rf_deployment Deployed Online

API reference **Test**

Enter input data

Text input

JSON input

Enter data manually or use a CSV file to populate the spreadsheet. Max file size is 50 MB.

Download CSV template

Browse local files

Search in space

Clear all

	QUARTER (int64)	MONTH (int64)	DAY_OF_MONTH (int64)	DAY_OF_WEEK (int64)	FL_NUM (int64)	ORIGIN (int64)	DEST (int64)	CRS_DEP_TIME.1 (int64)	CRS_ARR_TIME.1 (int64)
1	Start typing or drag and drop a CSV file...								
2									
3									
4									
5									
6									

0 rows, 12 columns

Predict

Output is predicted by ML model successfully.