

# Training & Testing Model on IBM cloud

**Team ID - PNT2022TMID24853**

Team Leader – YARRADODDI CHATURYA

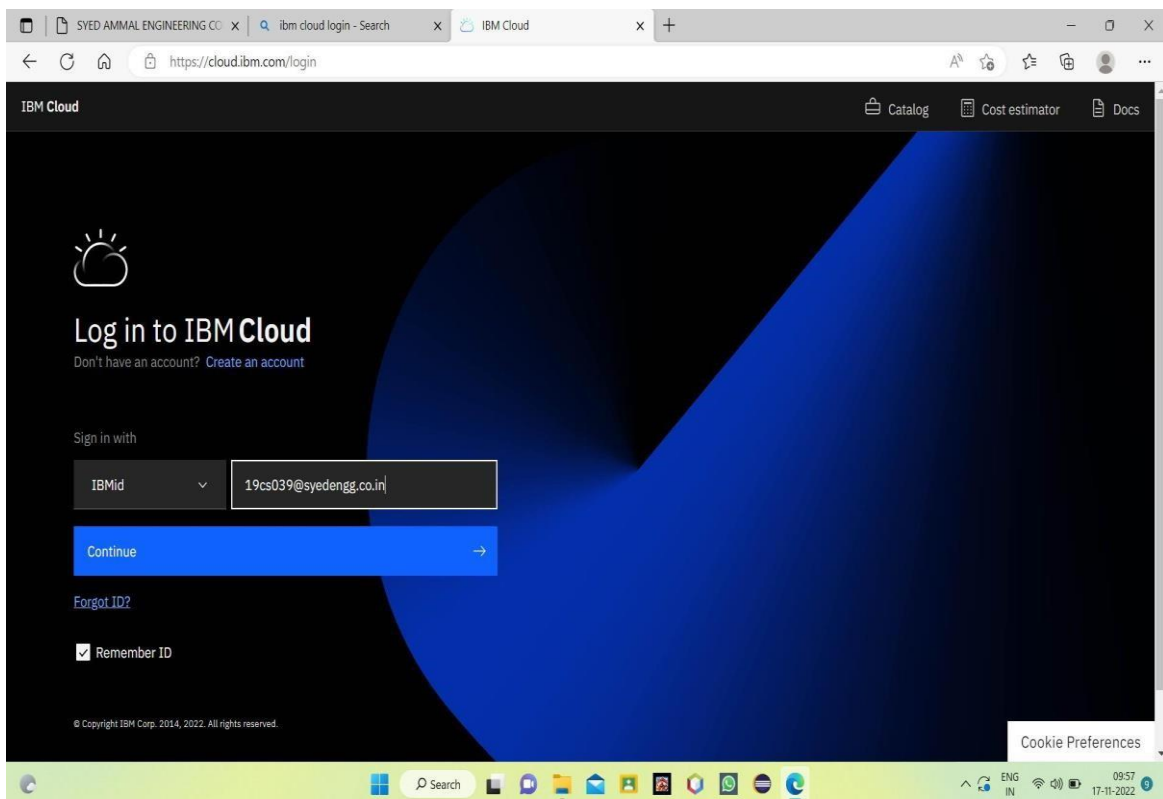
Team Member – ALLA MOUNIKA

Team Member – KOMERLA VENKATA SHANMUKHA SARVAR

Team Member – MURALASETTY JAYASREE

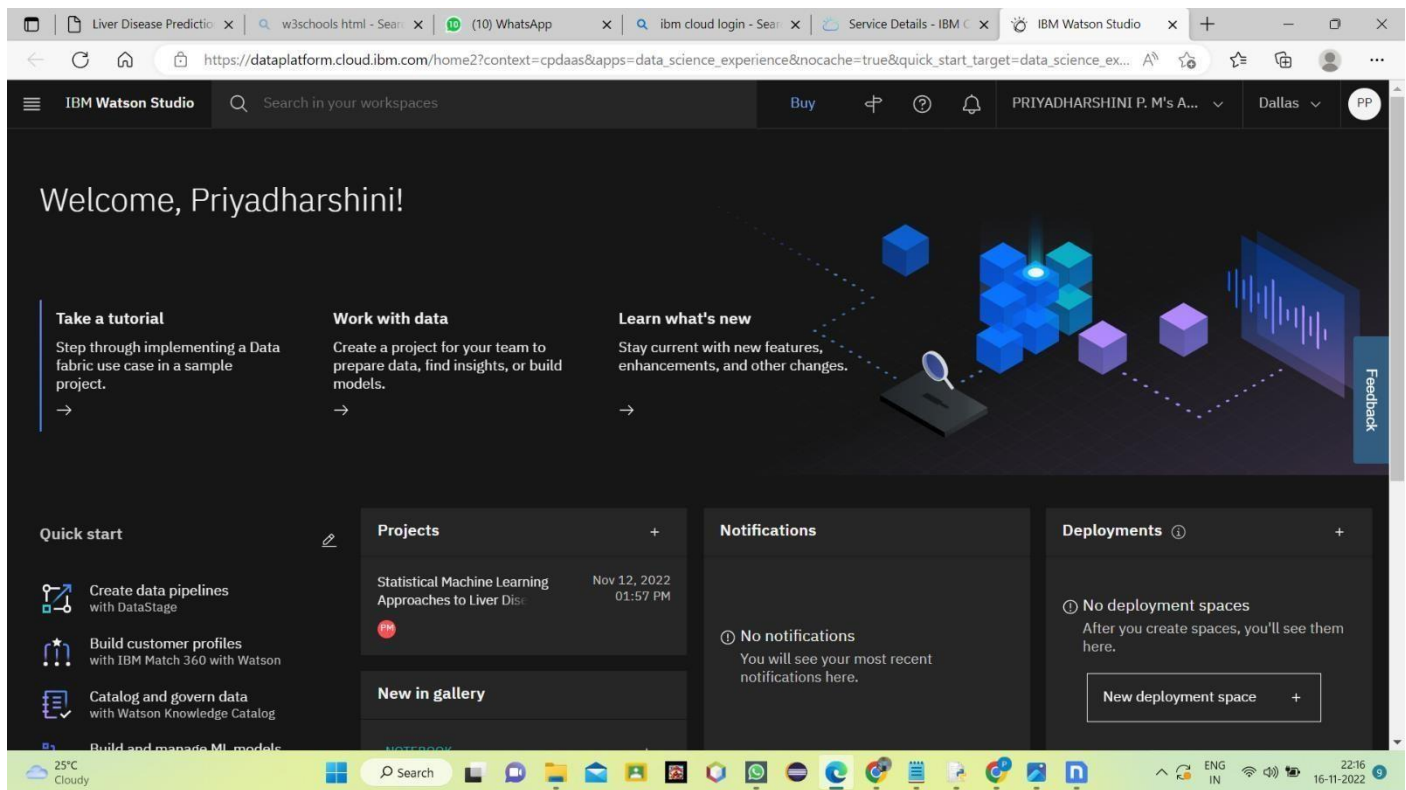
## 1.Register & Login IBM cloud

From given link we can Register and afterwards login the IBM cloud using credentials.



## 2.Opening IBM Watson Studio

To Run our model we use IBM watson studio inside we will create our own new project.

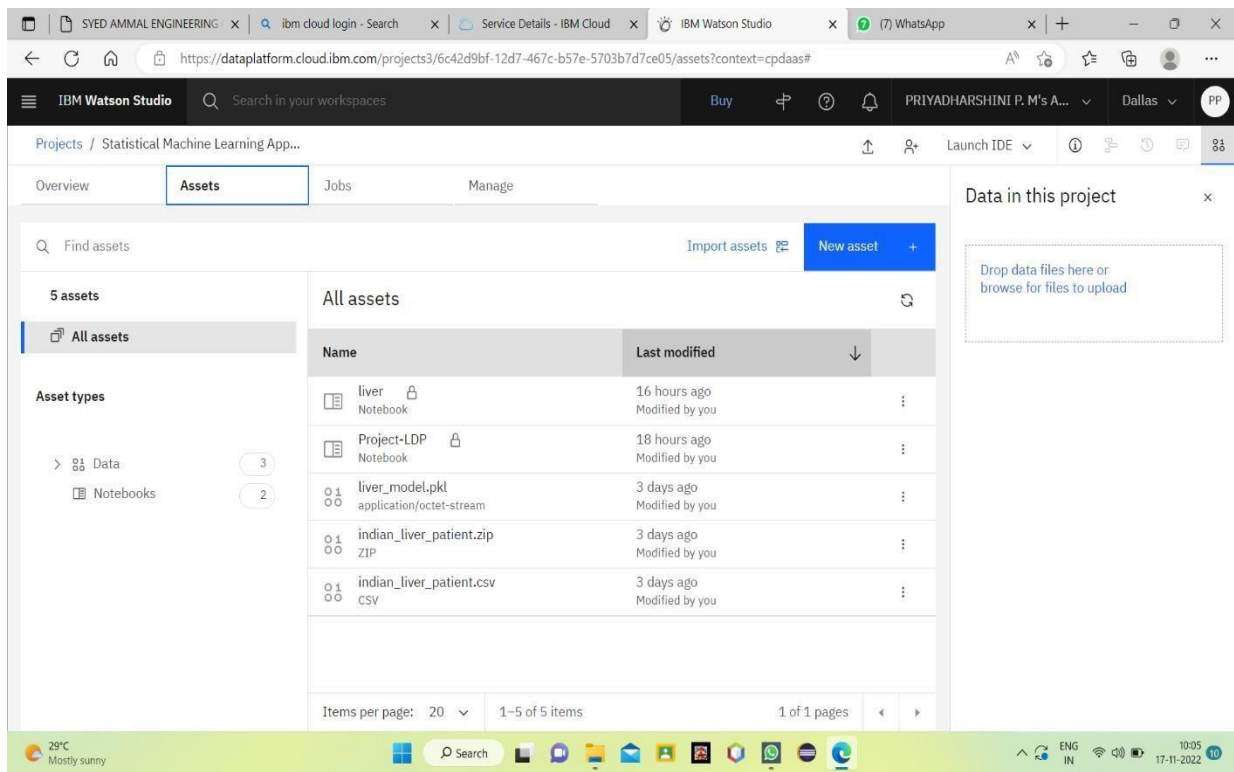


### 3.Adding Assests

Once we create a new project named Liver



Then we will Add some Assets like Python Notebook with extension of .ipynb

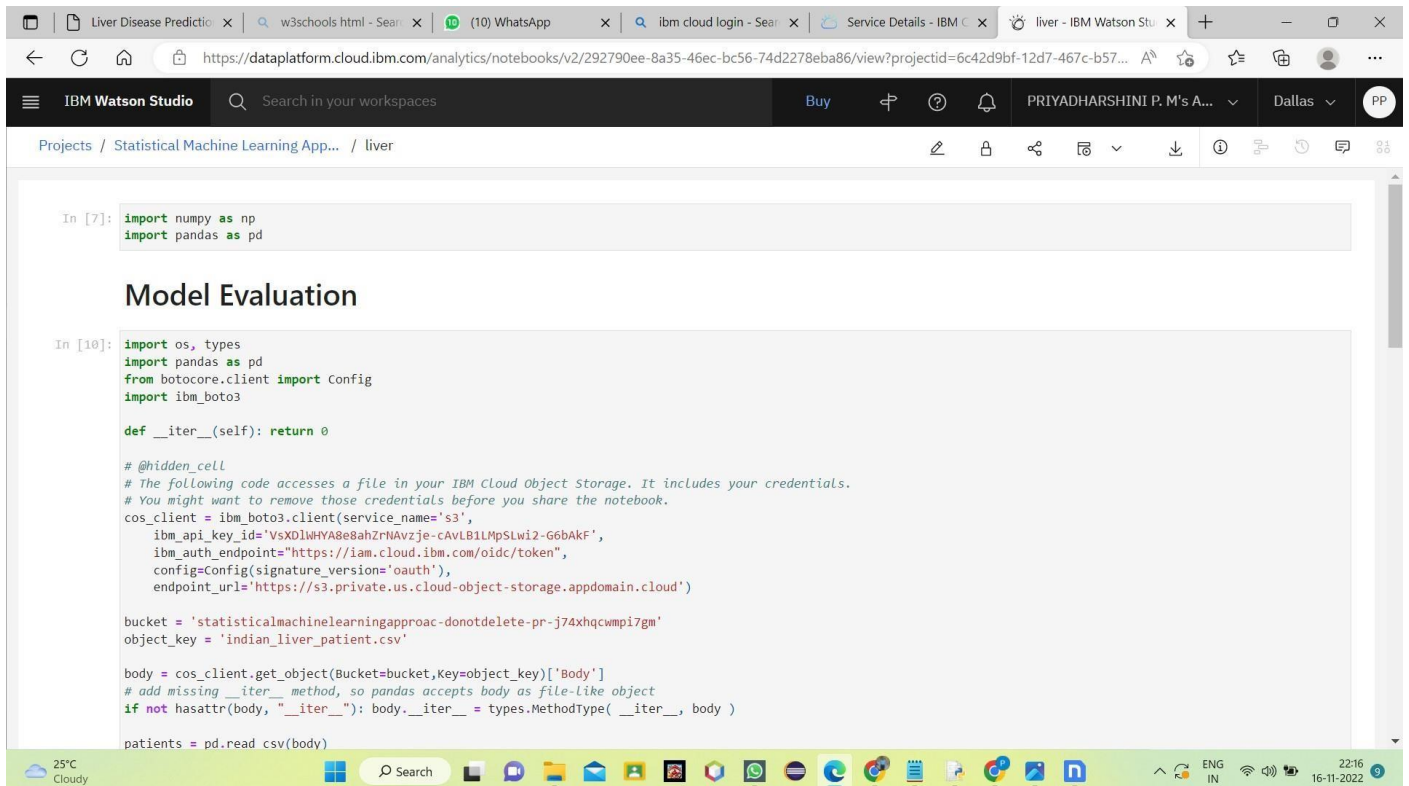


## 4 .Opening notebook file

At the end we will open the notebook file and after we download the dataset once we download it into zip file or we will add the file into data assets.

## 5.Insert File code

Then we will create a new cell and insert the code into it.



IBM Watson Studio

Projects / Statistical Machine Learning App... / liver

```

patients = pd.read_csv(body)
patients.head()

```

Out[10]:

	Age	Gender	Total_Bilirubin	Direct_Bilirubin	Alkaline_Phosphatase	Alamine_Aminotransferase	Aspartate_Aminotransferase	Total_Protiens	Albumin	Albumin_and_Globulin_Ratio	Dataset
0	65	Female	0.7	0.1	187	16	18	6.8	3.3	0.90	1
1	62	Male	10.9	5.5	699	64	100	7.5	3.2	0.74	1
2	62	Male	7.3	4.1	490	60	68	7.0	3.3	0.89	1
3	58	Male	1.0	0.4	182	14	20	6.8	3.4	1.00	1
4	72	Male	3.9	2.0	195	27	59	7.3	2.4	0.40	1

```

In [11]: from sklearn import ensemble
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
import joblib

patients['Gender']=patients['Gender'].apply(lambda x:1 if x=='Male' else 0)
patients=patients.fillna(0.94)

X=patients[['Total_Bilirubin', 'Direct_Bilirubin',
            'Alkaline_Phosphatase', 'Alamine_Aminotransferase',
            'Total_Protiens', 'Albumin', 'Albumin_and_Globulin_Ratio']]
y=patients['Dataset']

X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3,random_state=123)

print('Shape training set: X:{}, y:{}'.format(X_train.shape, y_train.shape))
print('Shape test set: X:{}, y:{}'.format(X_test.shape, y_test.shape))

```

Know About the directory by using `pwd` command.

```

In [1]: pwd
Out[1]: '/home/wsuser/work'

```

## 6. Train the Model in IBM Watson Studio

After Completion of these stuffs we will move forward to start Train the model.

IBM Watson Studio

Projects / Statistical Machine Learning App... / liver

```

In [11]: from sklearn import ensemble
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
import joblib

patients['Gender']=patients['Gender'].apply(lambda x:1 if x=='Male' else 0)
patients=patients.fillna(0.94)

X=patients[['Total_Bilirubin', 'Direct_Bilirubin',
            'Alkaline_Phosphatase', 'Alamine_Aminotransferase',
            'Total_Protiens', 'Albumin', 'Albumin_and_Globulin_Ratio']]
y=patients['Dataset']

X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3,random_state=123)

print('Shape training set: X:{}, y:{}'.format(X_train.shape, y_train.shape))
print('Shape test set: X:{}, y:{}'.format(X_test.shape, y_test.shape))

model = ensemble.RandomForestClassifier()
model.fit(X_train, y_train) #Put X_Train.values while running The app.py---
y_pred = model.predict(X_test)
print('Accuracy : {}'.format(accuracy_score(y_test, y_pred)))

clf_report = classification_report(y_test, y_pred)
print('Classification report')
print("-----")
print(clf_report)
print("-----")

joblib.dump(model,r"home\wsuser\work\liver_model.pkl")

```

Outcome of the Trained Model will be shown after compiling and summarizing it.

The screenshot shows the IBM Watson Studio interface. The top navigation bar includes the IBM Watson Studio logo, a search bar, and user information (PRIYADHARSHINI P. M's A...). The main workspace displays a Jupyter Notebook with the following code and output:

```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3,random_state=123)

print('Shape training set: X:{}, y:{}'.format(X_train.shape, y_train.shape))
print('Shape test set: X:{}, y:{}'.format(X_test.shape, y_test.shape))

model = ensemble.RandomForestClassifier()
model.fit(X_train, y_train) #Put X_train.values while running the app.py--
y_pred = model.predict(X_test)
print('Accuracy : {}'.format(accuracy_score(y_test, y_pred)))

clf_report = classification_report(y_test, y_pred)
print('Classification report')
print("-----")
print(clf_report)
print("-----")

joblib.dump(model,r"\home\wsuser\work\liver_model.pkl")
```

The output shows the shape of the training and test sets, the accuracy score, and the classification report:

```
Shape training set: X:(6703, 7), y:(6703,)
Shape test set: X:(2874, 7), y:(2874,)
Accuracy : 1.0
Classification report
-----
              precision    recall  f1-score   support

         1         1.00        1.00         1.00        2038
         2         1.00        1.00         1.00         836

 accuracy          1.00          1.00          1.00        2874
 macro avg          1.00          1.00          1.00        2874
weighted avg          1.00          1.00          1.00        2874
-----
```

This how the Model was Trained in IBM watson Studio.

## 7. Testing the Model

At the end of the day we will try to test the model which can give the desired and precise prediction based on the trained model.

The screenshot shows the IBM Watson Studio interface. The top navigation bar includes the IBM Watson Studio logo, a search bar, and user information (PRIYADHARSHINI P. M's A...). The main workspace displays a Jupyter Notebook with the following code:

```
In [2]: pwd
Out[2]: '/home/wsuser/work'

In [1]: from flask import Flask, render_template, url_for, flash, redirect
import joblib
from flask import request
import numpy as np

app = Flask(__name__, template_folder='templates')

@app.route('/')
def home():
    return render_template("home")

@app.route('/info.html')
def info():
    return render_template("info.html")

@app.route('/contact.html')
def contact():
    return render_template("contact.html")

@app.route('/liver')
def cancer():
    return render_template("liver.html")

def ValuePredictor(to_predict_list, size):
    to_predict = np.array(to_predict_list).reshape(1,size)
    if(size==7):
        loaded_model = joblib.load(r'\liver_model.pkl')
        result = loaded_model.predict(to_predict)
    return result[0]
```

This How the Model was tested in IBM Watson Studio.