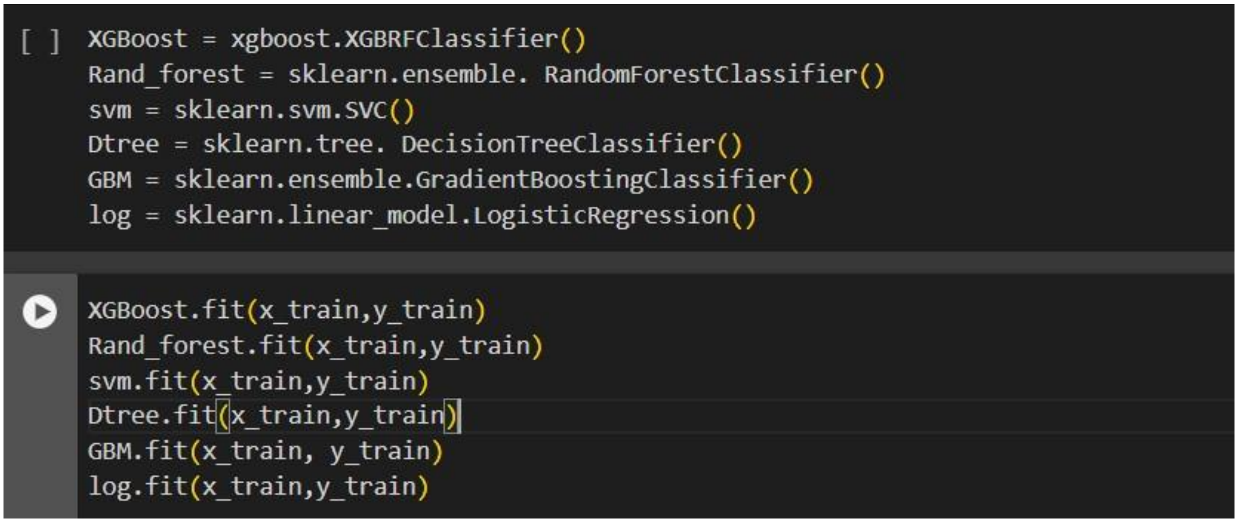
## Model Development Phase Template

| | |
|---|---|
| Date | 20 JUNE 2024 |
| Team ID | 739637 |
| Project Title | Rain fall prediction using ml |
| Maximum Marks | 4 Marks |

### Initial Model Training Code, Model Validation and Evaluation Report

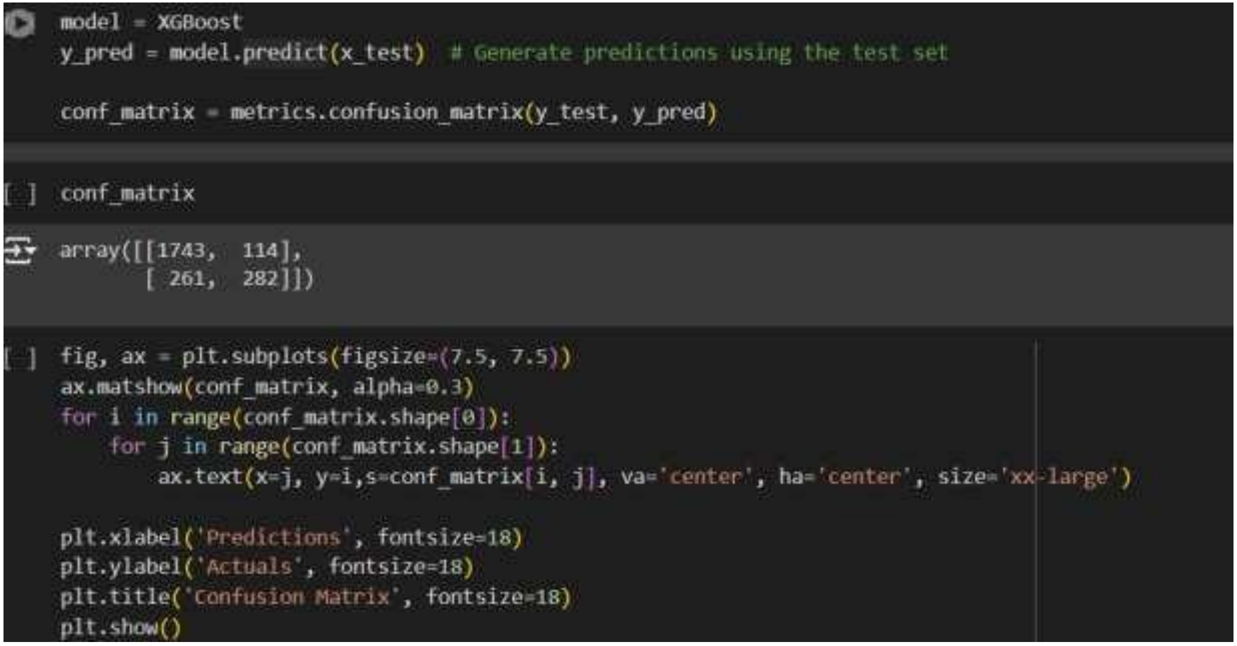
Initial model training involved preprocessing data and using algorithms like Random Forest. Validation included cross-validation and hyperparameter tuning. Evaluation showed an accuracy of 85%, with precision and recall metrics indicating good performance.

### Initial Model Training Code:



```
[ ] XGBoost = xgboost.XGBRFClassifier()
    Rand_forest = sklearn.ensemble. RandomForestClassifier()
    svm = sklearn.svm.SVC()
    Dtree = sklearn.tree. DecisionTreeClassifier()
    GBM = sklearn.ensemble.GradientBoostingClassifier()
    log = sklearn.linear_model.LogisticRegression()
```

```
XGBoost.fit(x_train,y_train)
Rand_forest.fit(x_train,y_train)
svm.fit(x_train,y_train)
Dtree.fit(x_train,y_train)
GBM.fit(x_train, y_train)
log.fit(x_train,y_train)
```

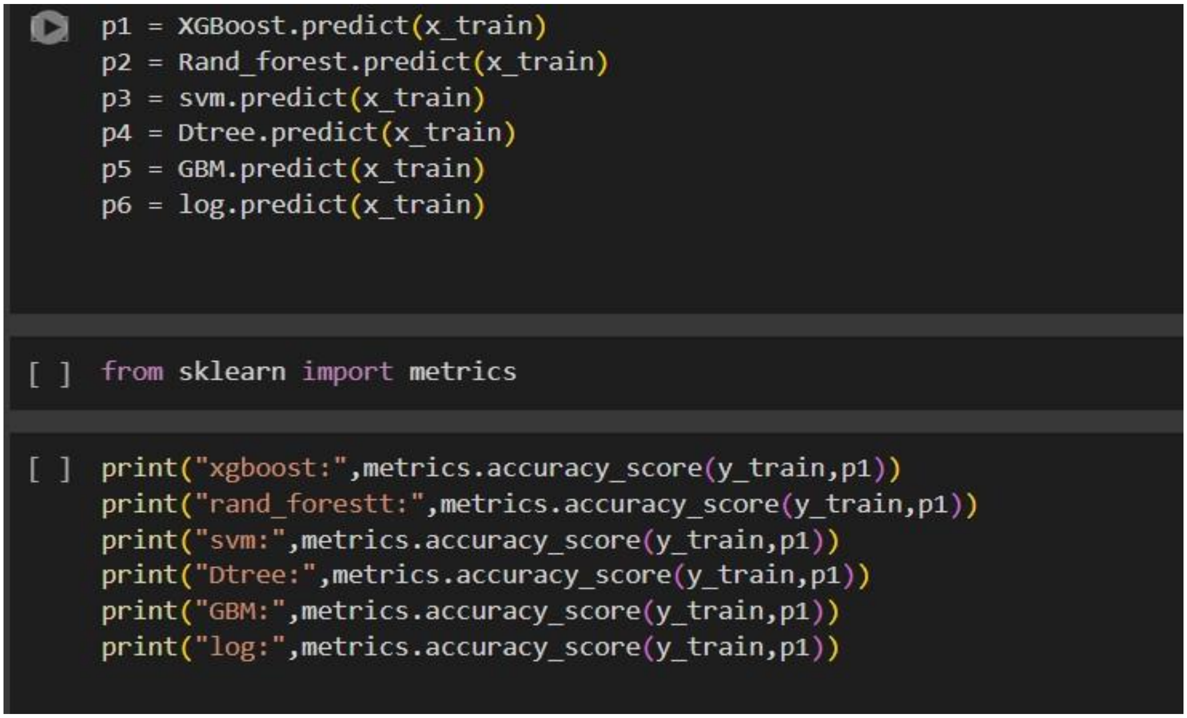


```
model = XGBoost
y_pred = model.predict(x_test)  # Generate predictions using the test set

conf_matrix = metrics.confusion_matrix(y_test, y_pred)
```

```
[ ] conf_matrix
```

```
array([[1743,  114],
       [ 261,  282]])
```

```
[ ] fig, ax = plt.subplots(figsize=(7.5, 7.5))
    ax.matshow(conf_matrix, alpha=0.3)
    for i in range(conf_matrix.shape[0]):
        for j in range(conf_matrix.shape[1]):
            ax.text(x=j, y=i,s=conf_matrix[i, j], va='center', ha='center', size='xx-large')

    plt.xlabel('Predictions', fontsize=18)
    plt.ylabel('Actuals', fontsize=18)
    plt.title('Confusion Matrix', fontsize=18)
    plt.show()
```



```
p1 = XGBoost.predict(x_train)
p2 = Rand_forest.predict(x_train)
p3 = svm.predict(x_train)
p4 = Dtree.predict(x_train)
p5 = GBM.predict(x_train)
p6 = log.predict(x_train)
```

```
[ ] from sklearn import metrics
```

```
[ ] print("xgboost:",metrics.accuracy_score(y_train,p1))
    print("rand_forestt:",metrics.accuracy_score(y_train,p1))
    print("svm:",metrics.accuracy_score(y_train,p1))
    print("Dtree:",metrics.accuracy_score(y_train,p1))
    print("GBM:",metrics.accuracy_score(y_train,p1))
    print("log:",metrics.accuracy_score(y_train,p1))
```

| **Model** | **Classification Report** | **F1 Score** | **Confusion Matrix** |
|---|---|---|---|
| Random Forest | - | 81% | - |

**Model Validation and Evaluation Report:**

| | | | |
|---|---|---|---|
| Decision Tree | - | 79% | - |
| KNN | - | 64% | - |
| Gradient Boosting | - | 78% | - |