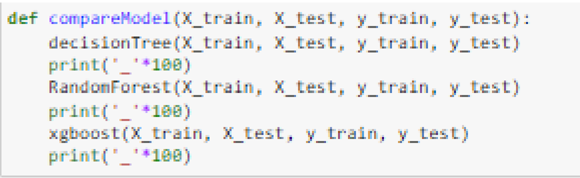
Testing Model With Multiple Evaluation Metrics:

Multiple evaluation metrics means evaluating the model's performance on a test set using different performance measures. This can provide a more comprehensive understanding of the model's strengths and weaknesses. We are using evaluation metrics for classification tasks including accuracy, precision, recall, support and F1-score.

Compare The Model:

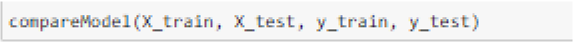
For comparing the above four models, the compareModel function is defined.



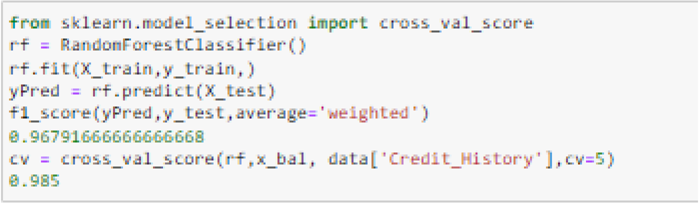
```
def compareModel(X_train, X_test, y_train, y_test):
    decisionTree(X_train, X_test, y_train, y_test)
    print('_'*100)
    RandomForest(X_train, X_test, y_train, y_test)
    print('_'*100)
    xgboost(X_train, X_test, y_train, y_test)
    print('_'*100)
```

Comparing Model Accuracy Before & After Applying Hyperparameter Tuning

Evaluating performance of the model From sklearn, cross_val_score is used to evaluate the score of the model. On the parameters, we have given rf (model name), x, y, cv (as 5 folds). Our model is performing well. So, we are saving the model by pickle.dump().



```
compareModel(X_train, X_test, y_train, y_test)
```



```
from sklearn.model_selection import cross_val_score
rf = RandomForestClassifier()
rf.fit(X_train,y_train,)
yPred = rf.predict(X_test)
f1_score(yPred,y_test,average='weighted')
0.9679166666666668
cv = cross_val_score(rf,x_bal, data['Credit_History'],cv=5)
0.985
```