

## Project Development Phase

### Model Performance Test

Date	10 November 2022
Team ID	PNT2022TMID06255
Project Name	Project – Early Detection of Chronic Kidney Disease using Machine Learning
Maximum Marks	10 Marks

### Model Performance Testing:

Project team shall fill the following information in model performance testing template.

S.No.	Parameter	Values	Screenshot
1.	Metrics	<p><b>Regression Model:</b> MAE - , MSE - , RMSE - , R2 score -</p> <p><b>Classification Model:</b> Confusion Matrix - , Accuracy Score- &amp; Classification Report -</p>	See Below
2.	Tune the Model	Hyperparameter Tuning - Validation Method -	See Below

## 1. Metrics

### Model: Random Forest Classifier

The screenshot shows a Jupyter Notebook interface with the following code:

```

# Importing the model upon training data
model.fit(x_train, y_train)

# Randomized test (Classifier)

# Predicting the test data
y_pred = model.predict(x_test)

# Displaying predicted data
y_pred

# Printing the predicted data
y_pred

# Importing metrics to check the performance of the model
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix

# Displaying accuracy score of testing data
accuracy_score(y_test, y_pred)

```

The output of the notebook shows the predicted values for the test data, which are mostly 0s and 1s, indicating a binary classification task. The accuracy score is also displayed at the bottom of the notebook.

## 2. Tune the Model

### Hyperparameter Tuning:

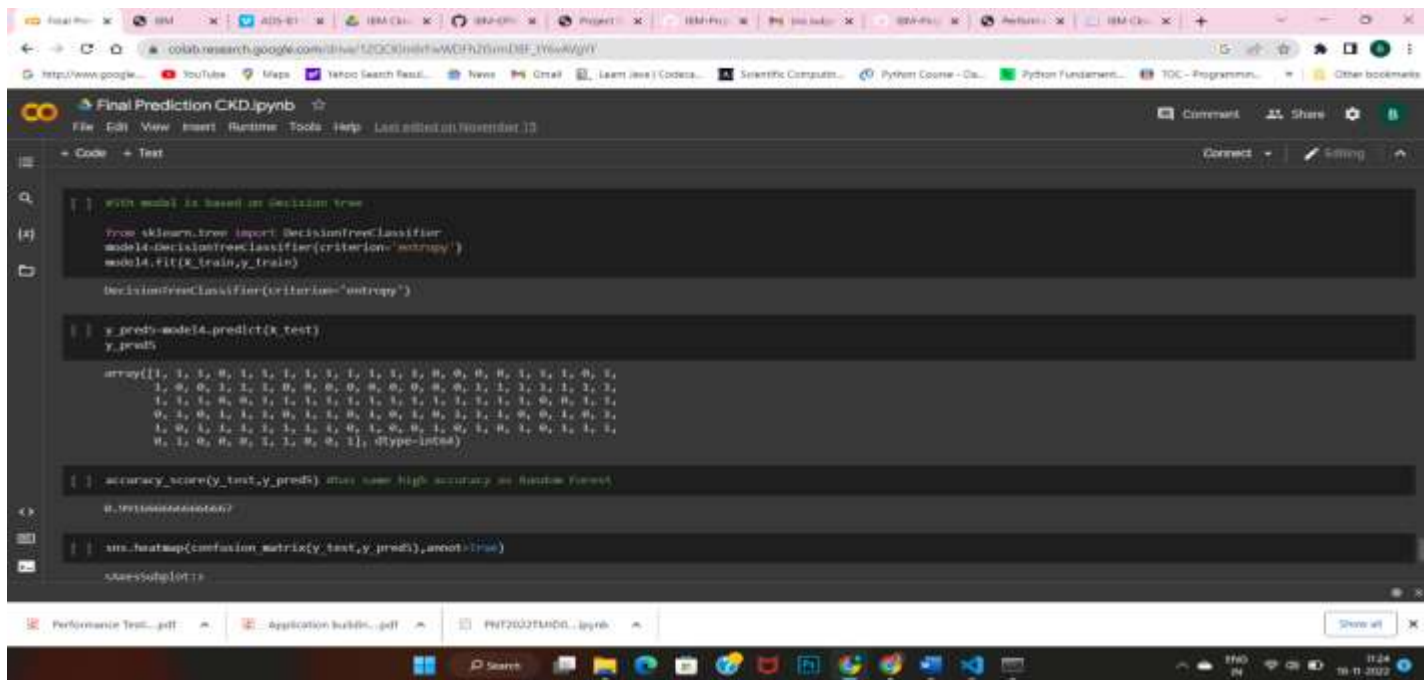
- The number of features is important and should be tuned in random forest classification.
- Initially all parameters in the dataset are taken as independent values to arrive at the dependent decision of Chronic Kidney Disease or No Chronic Kidney Disease.
- But the result was not accurate so used only 8 more correlated values as independent values to arrive at the dependent decision of Chronic Kidney Disease or not.

### Validation Method:

It involves **partitioning the training data set into subsets, where one subset is held out to test the performance of the model.** This data set is called the validation data set.

Cross validation is to use different models and identify the best:

### Decision Tree Model performance values:



```
[ ] #CKD model is based on Decision Tree

from sklearn.tree import DecisionTreeClassifier
model4=DecisionTreeClassifier(criterion='entropy')
model4.fit(X_train,y_train)

DecisionTreeClassifier(criterion='entropy')

[ ] y_preds=model4.predict(X_test)
y_preds

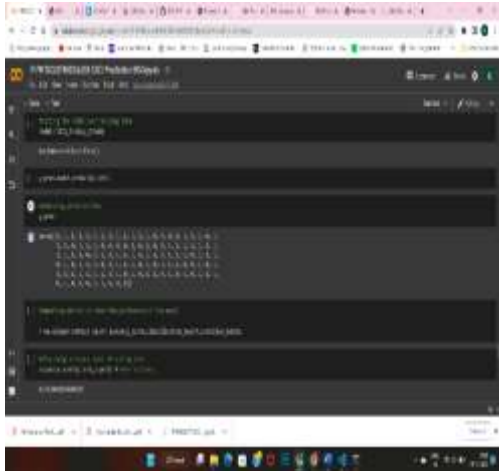
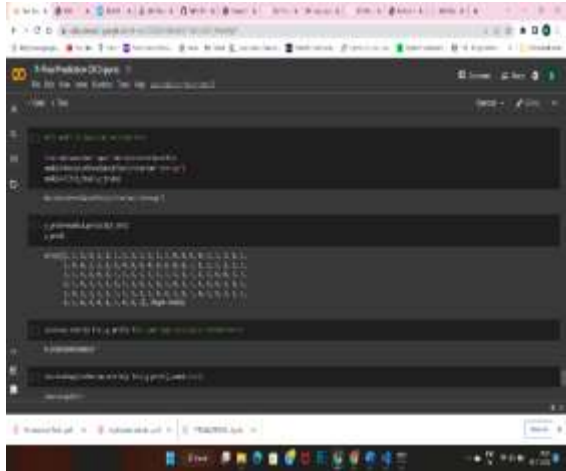
array([1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 1,
       1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1,
       0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1,
       1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1,
       0, 1, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1])

[ ] accuracy_score(y_test,y_preds) #this gave high accuracy as Random Forest
0.7714285714285714

[ ] sm.heatmap(confusion_matrix(y_test,y_preds),annot=True)

%matplotlib inline
```

Hence we tested with Logistic regression and Random Forest Classification wherein the accuracy of Random Forest classification is 95% compared with Logistic Regression.

Metric	Random Forest Classification	Decision Tree Classification
Accuracy	0.9916	0.989
Other metrics		

The above table shows that Random Forest Classification gives better results over Decision TreeRegression.