Classification Assignment

Problem Statement or Requirement:

A requirement from the Hospital, Management asked us to create a predictive

model which will predict the Chronic Kidney Disease (CKD) based on the

several parameters. The Client has provided the dataset of the same.

1.) Identify your problem statement. To identify the CKD based on the parameters like age bp, sugar level etc.,

2.) Tell basic info about the dataset (Total number of rows, columns) . The dataset has 399, rows 25 columns

3.) Mention the pre-processing method if you're doing any (like converting

string to number – nominal data)

1) The output column is classification which is yes or no type of answer, so converted it into 1 or 0 using dataset['classification'] = dataset['classification'].map({'yes':1,'no':0})
2) converted the data to ordinal using pd.get dummies dataset = pd.get_dummies(dataset,drop_first=True)
3) converted the output classification column to 1D array dependent = dataset[['classification']].values.ravel()
4) Standardised the input colums using ('sc',StandardScaler())

4.) Develop a good model with good evaluation metric. You can use any

machine learning algorithm; you can create many models. Finally, you

have to come up with final model.

5.) All the research values of each algorithm should be documented. (You

can make tabulation or screenshot of the results.)

6.) Mention your final model, justify why u have chosen the same.

Note: Mentioned points are necessary, kindly mail your document as

well as .ipynb (code file) with respective name.

 Sub file name also should be properly named for Example

(SVM_Ramisha_Assi-5.ipynb)

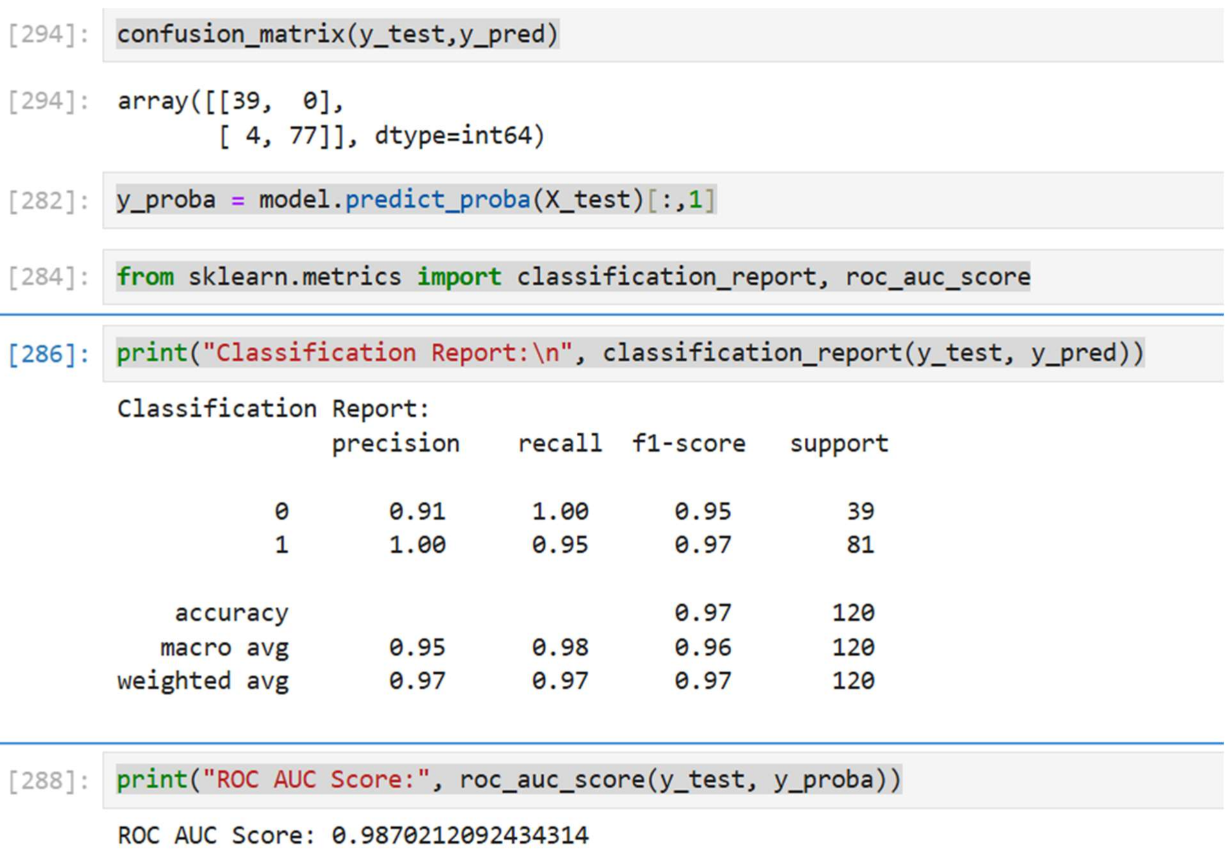
Communication is important (How you are representing the

document.)

Kindly uploaded in the Github and Share it with us

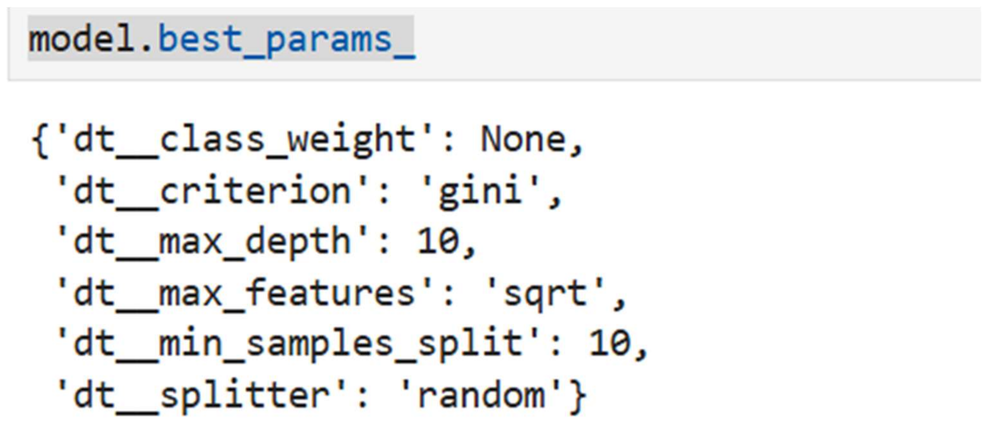
Decision Tree

has score of ROC AUC 0.987



```
[294]: confusion_matrix(y_test,y_pred)

[294]: array([[39,  0],
              [ 4, 77]], dtype=int64)

[282]: y_proba = model.predict_proba(X_test)[:,1]

[284]: from sklearn.metrics import classification_report, roc_auc_score

[286]: print("Classification Report:\n", classification_report(y_test, y_pred))

       Classification Report:
                      precision    recall  f1-score   support

                  0       0.91      1.00      0.95        39
                  1       1.00      0.95      0.97        81

           accuracy                           0.97       120
          macro avg       0.95      0.98      0.96       120
       weighted avg       0.97      0.97      0.97       120

[288]: print("ROC AUC Score:", roc_auc_score(y_test, y_proba))

       ROC AUC Score: 0.9870212092434314
```

Best params are

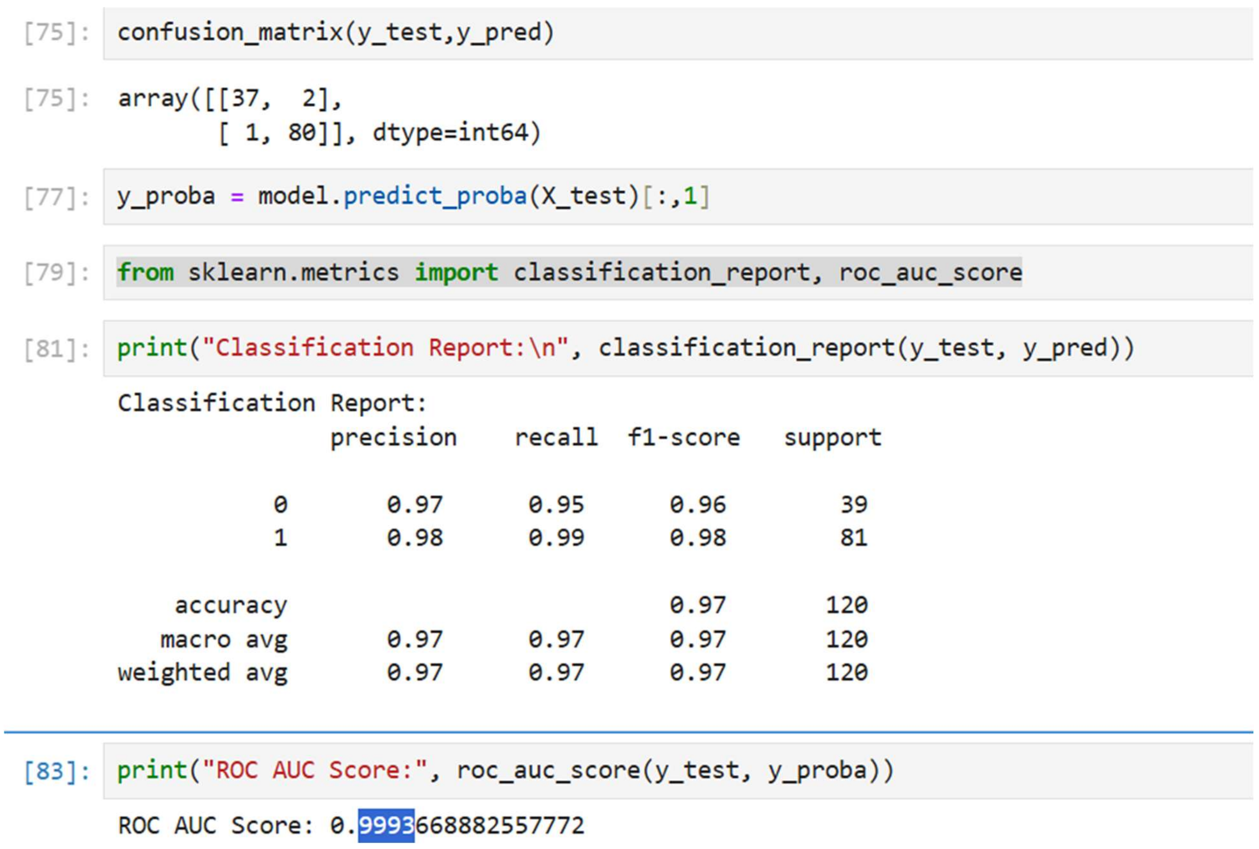


```
model.best_params_
```

```
{'dt__class_weight': None,
 'dt__criterion': 'gini',
 'dt__max_depth': 10,
 'dt__max_features': 'sqrt',
 'dt__min_samples_split': 10,
 'dt__splitter': 'random'}
```

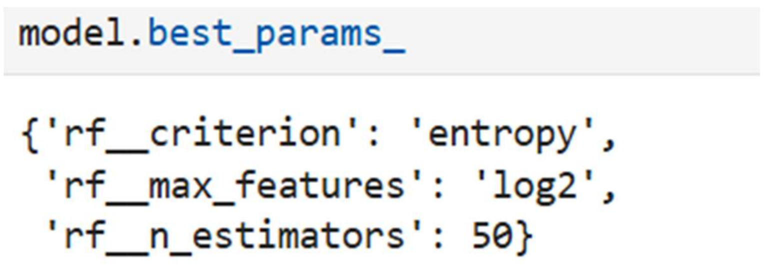
Random Forest

has score of ROC AUC 0.9993



```
[75]: confusion_matrix(y_test,y_pred)

[75]: array([[37,  2],
             [ 1, 80]], dtype=int64)

[77]: y_proba = model.predict_proba(X_test)[:,1]

[79]: from sklearn.metrics import classification_report, roc_auc_score

[81]: print("Classification Report:\n", classification_report(y_test, y_pred))

      Classification Report:
                     precision    recall  f1-score   support

                 0       0.97      0.95      0.96        39
                 1       0.98      0.99      0.98        81

          accuracy                           0.97       120
         macro avg       0.97      0.97      0.97       120
      weighted avg       0.97      0.97      0.97       120

[83]: print("ROC AUC Score:", roc_auc_score(y_test, y_proba))

      ROC AUC Score: 0.9993668882557772
```

Best params are

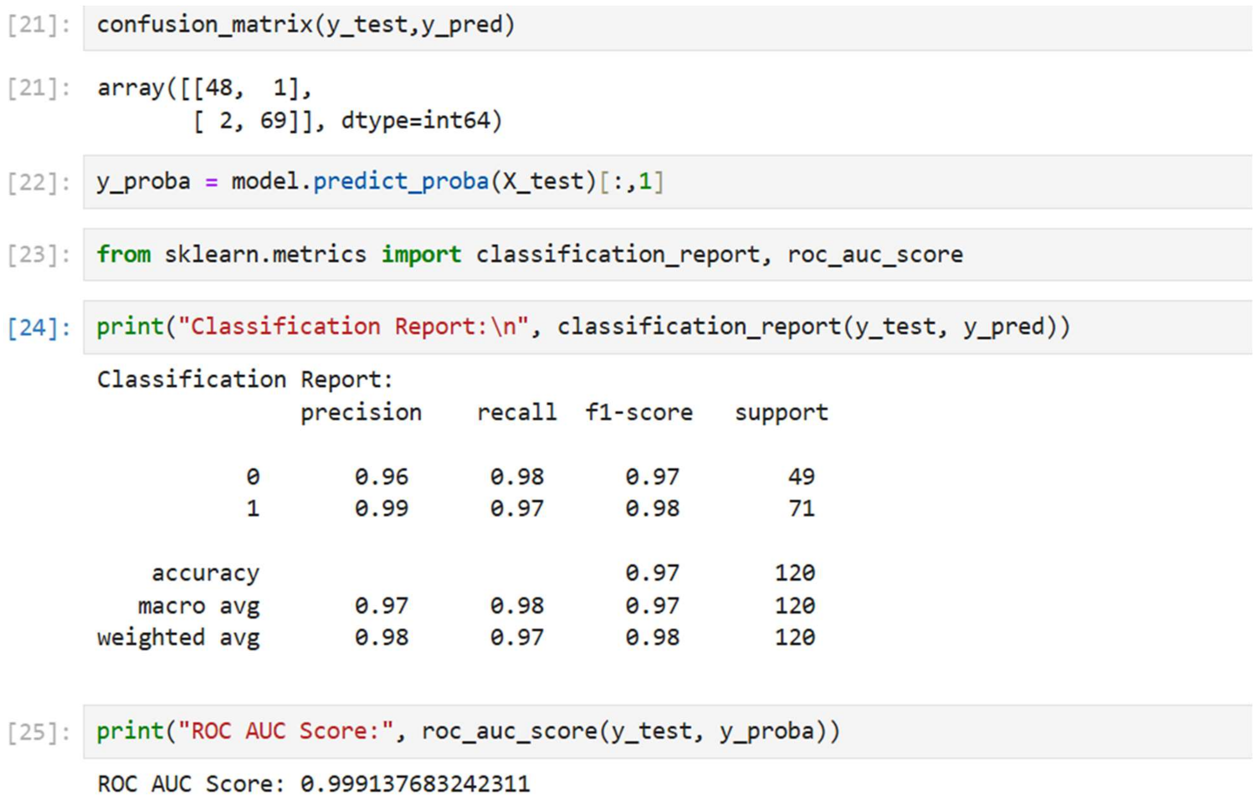


```
model.best_params_
```

```
{'rf__criterion': 'entropy',
 'rf__max_features': 'log2',
 'rf__n_estimators': 50}
```

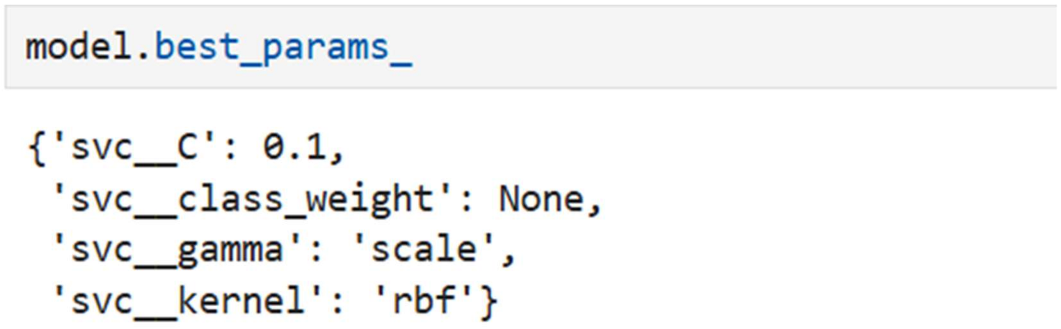
SVM classifiation

has score of ROC AUC 0.9991



```
[21]: confusion_matrix(y_test,y_pred)

[21]: array([[48,  1],
             [ 2, 69]], dtype=int64)

[22]: y_proba = model.predict_proba(X_test)[:,1]

[23]: from sklearn.metrics import classification_report, roc_auc_score

[24]: print("Classification Report:\n", classification_report(y_test, y_pred))

      Classification Report:
                    precision    recall  f1-score   support

                 0       0.96      0.98      0.97        49
                 1       0.99      0.97      0.98        71

          accuracy                           0.97       120
         macro avg       0.97      0.98      0.97       120
      weighted avg       0.98      0.97      0.98       120


[25]: print("ROC AUC Score:", roc_auc_score(y_test, y_proba))

      ROC AUC Score: 0.999137683242311
```

Best params are



```
model.best_params_
```

```
{'svc__C': 0.1,
 'svc__class_weight': None,
 'svc__gamma': 'scale',
 'svc__kernel': 'rbf'}
```

As a final conclusion, Support vector machine classification has the highest rate of AUC score that can be considered as best model for CKD use case