Assignment for classification:

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:

3 stages of problem identification method is

Machine Learning - (so far It's a Number data)
Supervised Learning -(Both input and Output is very clear)
Classification – (Output is Categorical data)

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

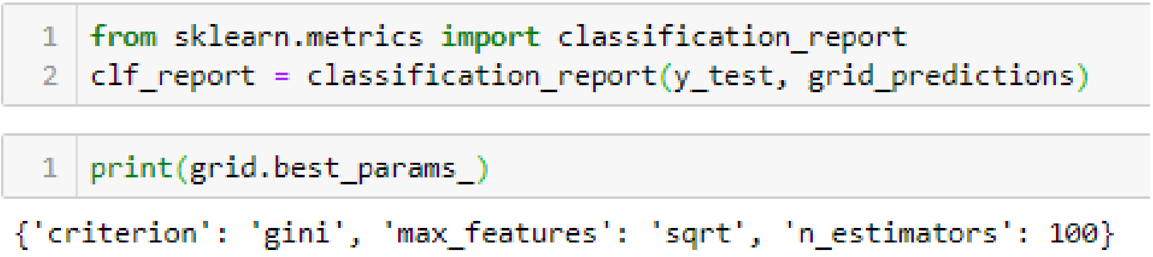
total number of rows -399
total number of columns -25

3)Mention the pre-processing method if you're doing any (like converting string to number – nominal data)

one hot encoding

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

Random Forest Classification:



```
from sklearn.metrics import classification_report
clf_report = classification_report(y_test, grid_predictions)
```

```
print(grid.best_params_)
```

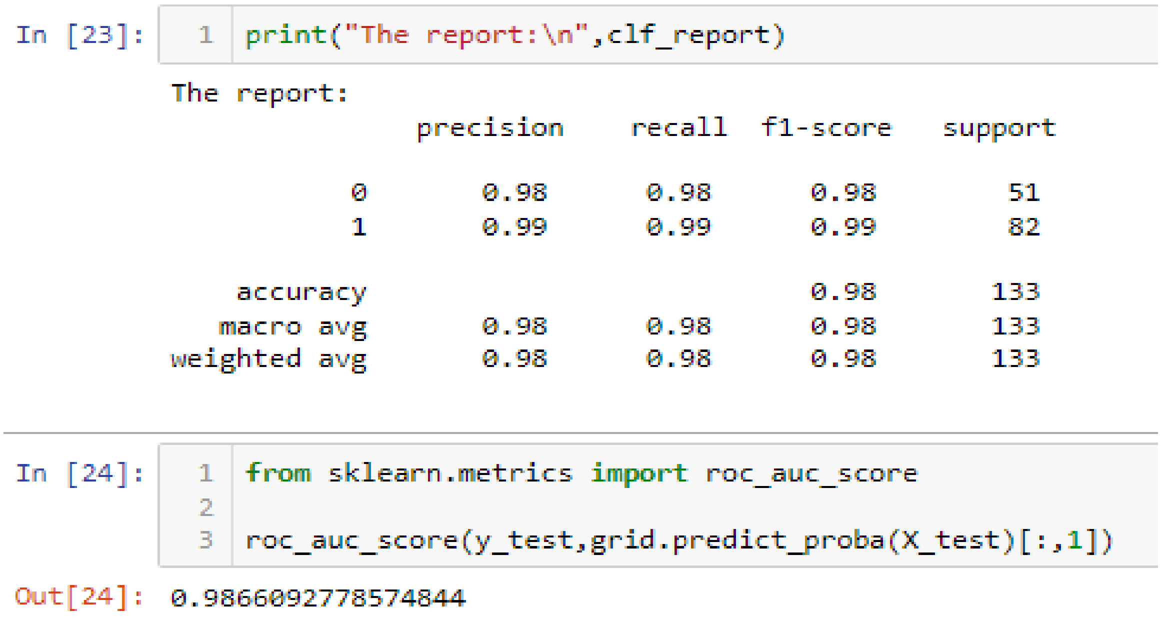
```
{'criterion': 'gini', 'max_features': 'sqrt', 'n_estimators': 100}
```

Best parameter using Random Forest:

Criterion – gini

Max features – sqrt

n- estimators – 100



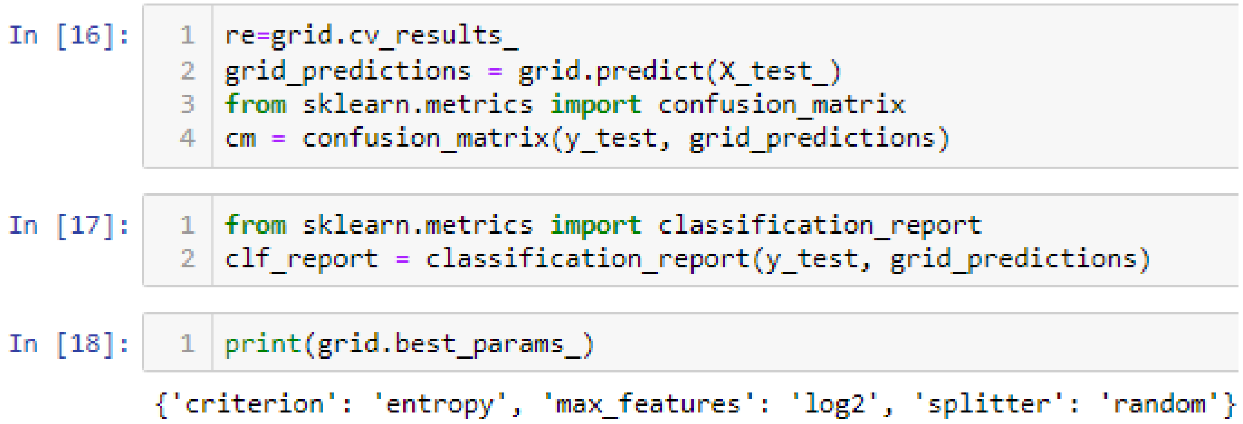
```
In [23]: print("The report:\n",clf_report)
```

```
The report:
               precision    recall  f1-score   support

           0       0.98      0.98      0.98        51
           1       0.99      0.99      0.99        82

    accuracy                           0.98       133
   macro avg       0.98      0.98      0.98       133
weighted avg       0.98      0.98      0.98       133
```

```
In [24]: from sklearn.metrics import roc_auc_score

         roc_auc_score(y_test,grid.predict_proba(X_test)[:,1])

Out[24]: 0.9866092778574844
```

The Random Forest Classification best The confusion Matrix ROC value is 0.98

Decision Tree Classification Method:



```
In [16]: re=grid.cv_results_
         grid_predictions = grid.predict(X_test_)
         from sklearn.metrics import confusion_matrix
         cm = confusion_matrix(y_test, grid_predictions)
```

```
In [17]: from sklearn.metrics import classification_report
         clf_report = classification_report(y_test, grid_predictions)
```

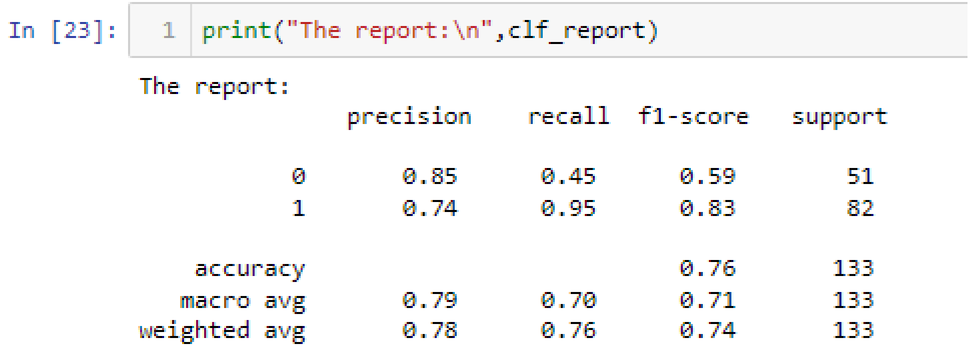
```
In [18]: print(grid.best_params_)

         {'criterion': 'entropy', 'max_features': 'log2', 'splitter': 'random'}
```

Best parameter using Decision Tree :

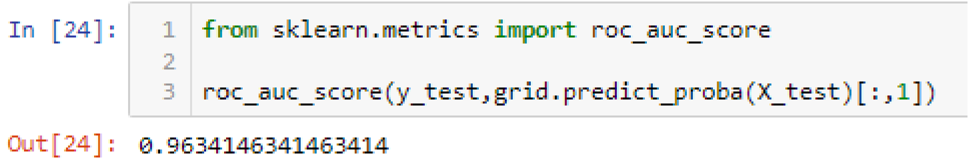
Criterion – entropy

Max features – log2

Splitter – random



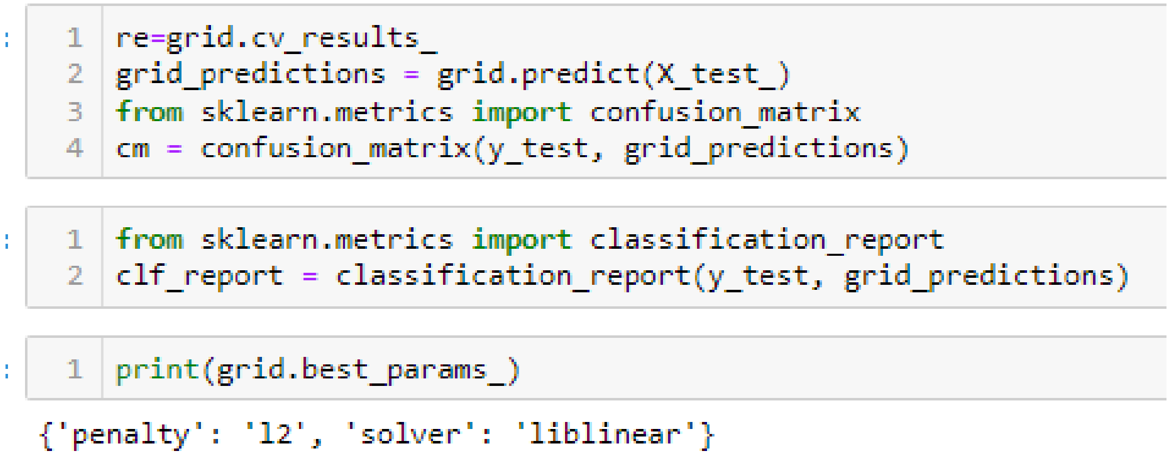
```
In [23]: print("The report:\n",clf_report)

         The report:
                       precision    recall  f1-score   support

                    0       0.85      0.45      0.59        51
                    1       0.74      0.95      0.83        82

             accuracy                           0.76       133
            macro avg       0.79      0.70      0.71       133
         weighted avg       0.78      0.76      0.74       133
```



```
In [24]: from sklearn.metrics import roc_auc_score

         roc_auc_score(y_test,grid.predict_proba(X_test)[:,1])

Out[24]: 0.9634146341463414
```

The Decision tree Classification best The confusion Matrix ROC value is 0.96

Logistic Regression Classification Method:



```
re=grid.cv_results_
grid_predictions = grid.predict(X_test_)
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, grid_predictions)
```

```
from sklearn.metrics import classification_report
clf_report = classification_report(y_test, grid_predictions)
```

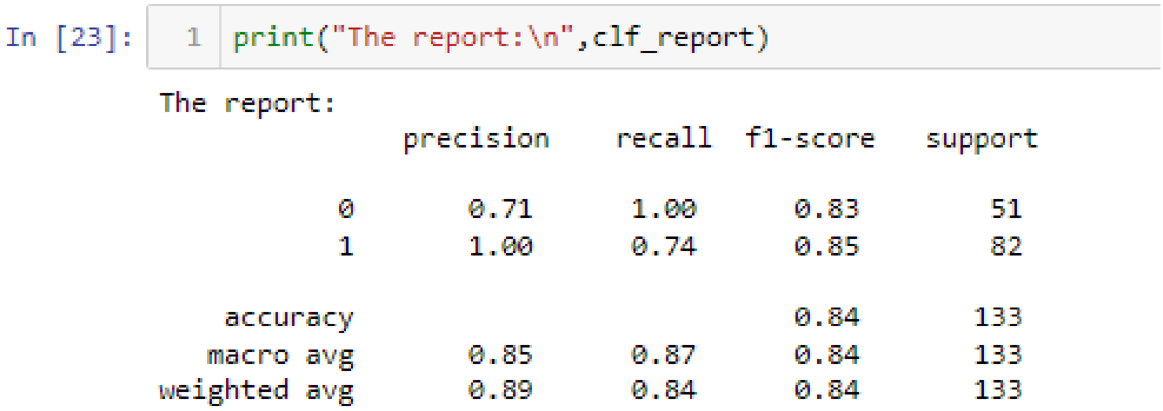
```
print(grid.best_params_)
```

```
{'penalty': 'l2', 'solver': 'liblinear'}
```

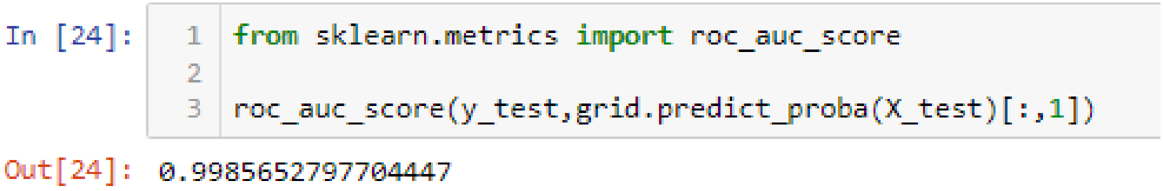
Best parameter using Logistic Regression:

Penalty – 12

Solver – liblinear



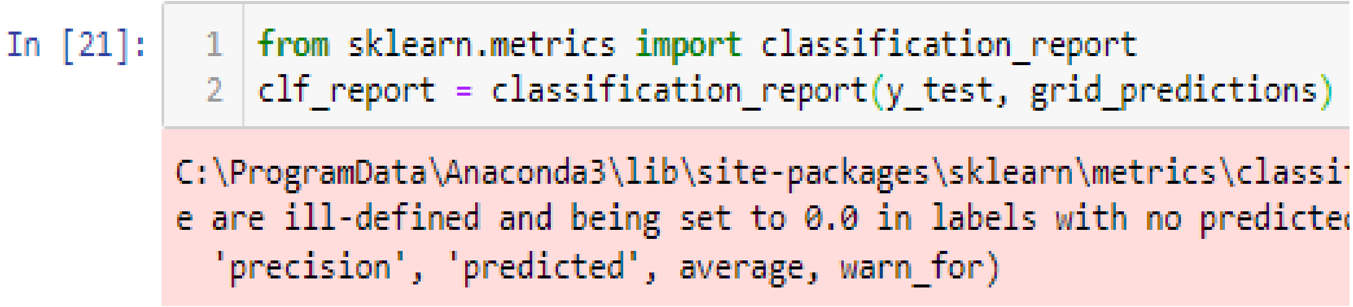
```
In [23]:  print("The report:\n",clf_report)

The report:
               precision    recall  f1-score   support

           0       0.71      1.00      0.83        51
           1       1.00      0.74      0.85        82

    accuracy                           0.84       133
   macro avg       0.85      0.87      0.84       133
weighted avg       0.89      0.84      0.84       133
```



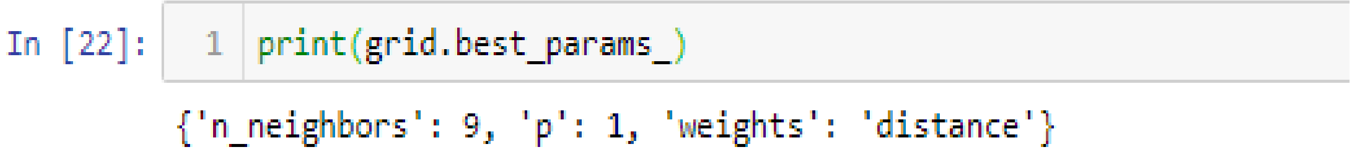
```
In [24]:  from sklearn.metrics import roc_auc_score

          roc_auc_score(y_test,grid.predict_proba(X_test)[:,1])

Out[24]:  0.9985652797704447
```

The Logistic Regression Classification best The confusion Matrix ROC value is 0.99

KNN Classification :



```
In [21]:  from sklearn.metrics import classification_report
          clf_report = classification_report(y_test, grid_predictions)

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\metrics\classi
e are ill-defined and being set to 0.0 in labels with no predicte
  'precision', 'predicted', average, warn_for)
```

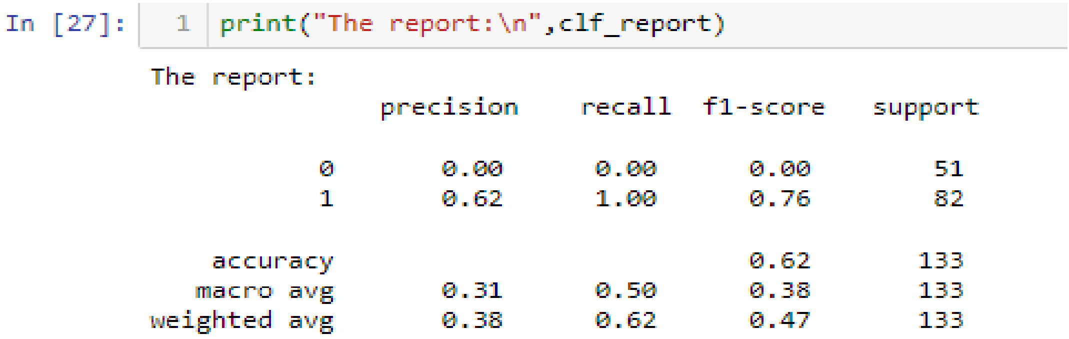


```
In [22]:  print(grid.best_params_)

{'n_neighbors': 9, 'p': 1, 'weights': 'distance'}
```

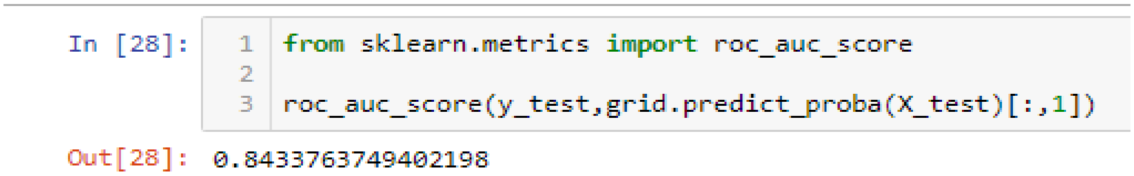
Best parameter using KNN:

n_neighbors – 9

p= 1
Weights = distance



```
In [27]:  print("The report:\n",clf_report)

The report:
               precision    recall  f1-score   support

           0       0.00      0.00      0.00        51
           1       0.62      1.00      0.76        82

    accuracy                           0.62       133
   macro avg       0.31      0.50      0.38       133
weighted avg       0.38      0.62      0.47       133
```

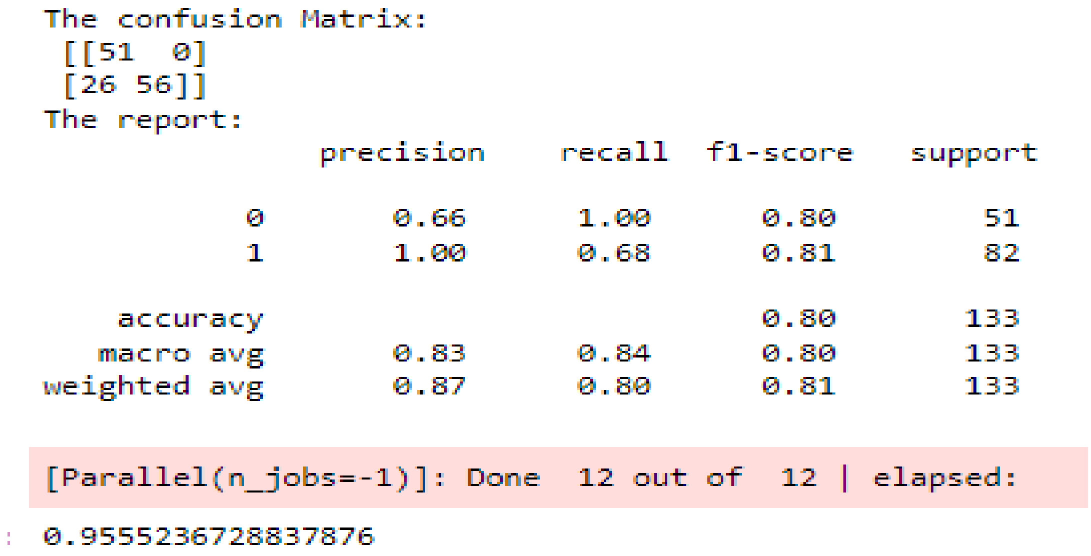


```
In [28]:  from sklearn.metrics import roc_auc_score

          roc_auc_score(y_test,grid.predict_proba(X_test)[:,1])

Out[28]:  0.8433763749402198
```

The Logistic Regression Classification best The confusion Matrix ROC value is 0.84

Naïve Bayes :

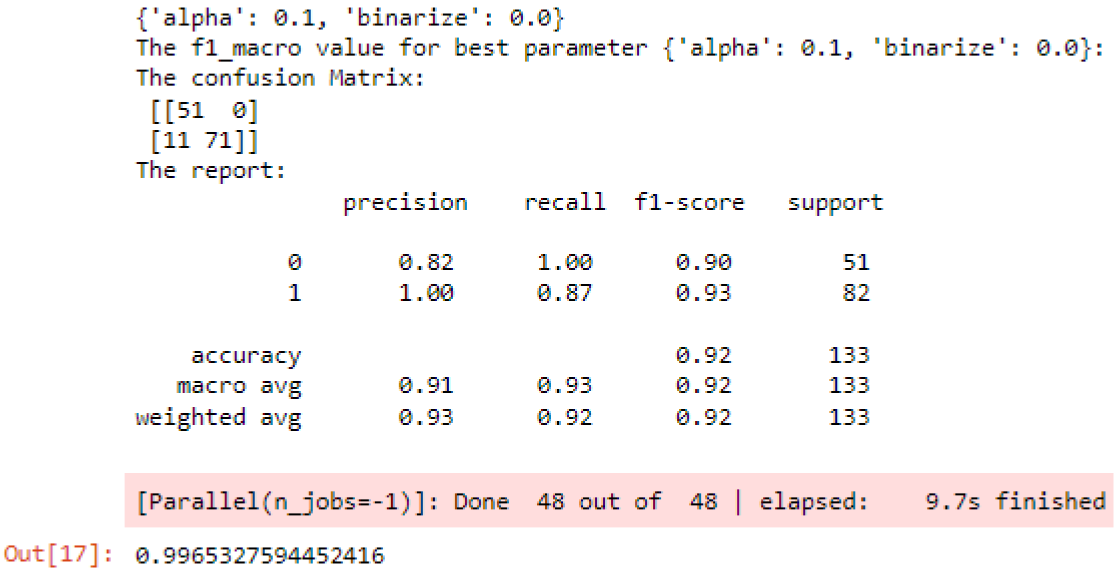
1) MultinomialNB:



```
The confusion Matrix:
 [[51  0]
 [26 56]]
The report:
              precision    recall  f1-score   support

           0       0.66      1.00      0.80        51
           1       1.00      0.68      0.81        82

    accuracy                           0.80       133
   macro avg       0.83      0.84      0.80       133
weighted avg       0.87      0.80      0.81       133

[Parallel(n_jobs=-1)]: Done  12 out of  12 | elapsed:

: 0.9555236728837876
```

Roc value is – 0.99

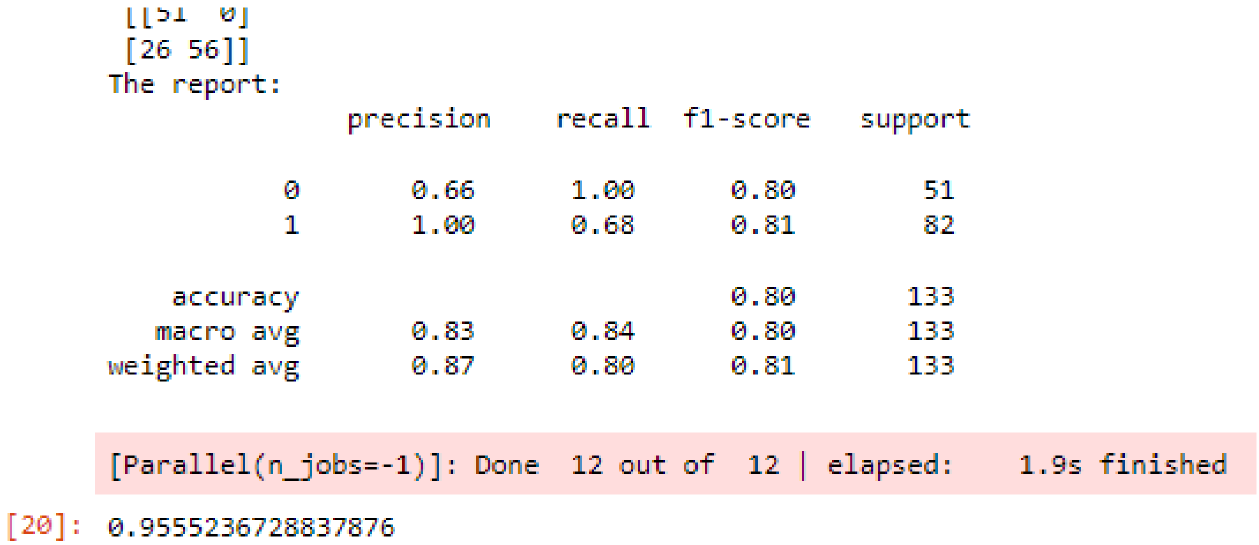
2) BernoulliNB:



```
{'alpha': 0.1, 'binarize': 0.0}
The f1_macro value for best parameter {'alpha': 0.1, 'binarize': 0.0}:
The confusion Matrix:
 [[51  0]
 [11 71]]
The report:
              precision    recall  f1-score   support

           0       0.82      1.00      0.90        51
           1       1.00      0.87      0.93        82

    accuracy                           0.92       133
   macro avg       0.91      0.93      0.92       133
weighted avg       0.93      0.92      0.92       133

[Parallel(n_jobs=-1)]: Done  48 out of  48 | elapsed:    9.7s finished

Out[17]: 0.9965327594452416
```

Roc value is – 0.99

3) ComplementNB:



```
[[51  0]
 [26 56]]
The report:
               precision    recall  f1-score   support

           0       0.66      1.00      0.80        51
           1       1.00      0.68      0.81        82

    accuracy                           0.80       133
   macro avg       0.83      0.84      0.80       133
weighted avg       0.87      0.80      0.81       133

[Parallel(n_jobs=-1)]: Done  12 out of  12 | elapsed:    1.9s finished
[20]: 0.9555236728837876
```

Roc value is – 0.95

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

So far Analys the all classification algorithm we got a best The confusion Matrix value (Accuracy & Roc) is 0.98 & 0.98 using Random Forest classification.. So we choose the final model is Random Forest Method.