Assignment on Classification Algorithms – Chronic Kidney Disease (CKD)

Question :

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
2. Tell basic info about the dataset (Total number of rows, columns)
3. Mention the pre-processing method if you're doing any (like converting string to number – nominal data)
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.

Solution:

1. **Problem Statement Identification** : Predicting Chronic Kidney Disease (CKD).

   3 Stages of Problem Identification :

   Stage 1 : ML
   Stage 2 : Supervised Learning
   Stage 3 : Classification

2. **Dataset Basic Info** : 399 rows × 25 columns

3. **Pre-Processing Method** : One Hot Encoding & Ordinal Encoding

4. **Good model with r2_score :** Refer the algorithm files under the folder named CKD

5. **Research on best model based on Accuracy, F1-Macro Value & ROC AUC Score :**

   Refer the document below.

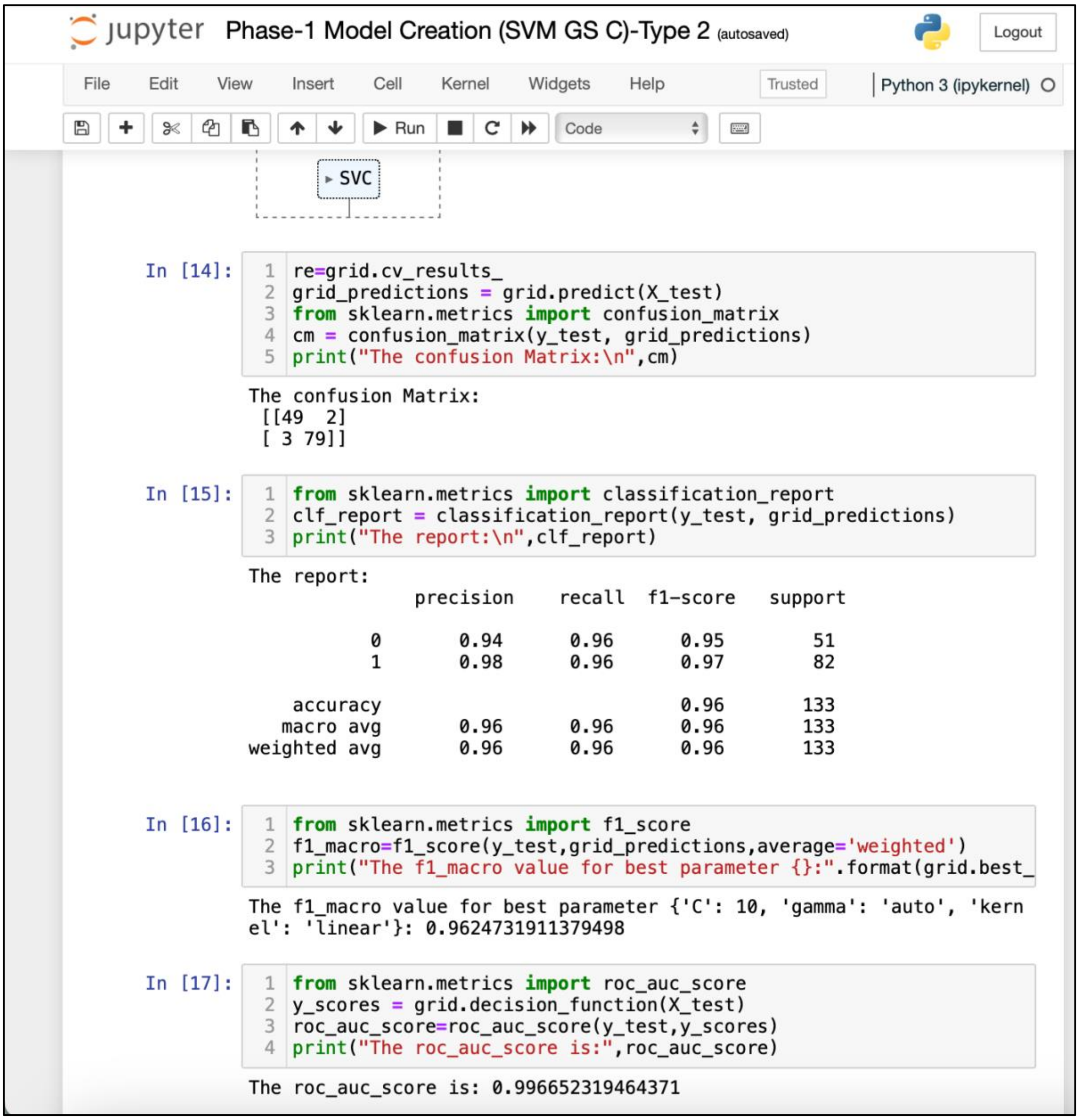
Research on Best Model Via Accuracy, F1-Macro Value & ROC AUC Score

1.SVM

Inference :

Accuracy = 0.96
F1 Macro = 0.9625 for Best Parameters = 'C': 10, 'gamma': 'auto', 'kernel': 'linear'
ROC – AUC Score = 0.9966

Proof:



In [14]:

```
re=grid.cv_results_
grid_predictions = grid.predict(X_test)
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, grid_predictions)
print("The confusion Matrix:\n",cm)
```

```
The confusion Matrix:
 [[49  2]
 [ 3 79]]
```

In [15]:

```
from sklearn.metrics import classification_report
clf_report = classification_report(y_test, grid_predictions)
print("The report:\n",clf_report)
```

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

           0       0.94      0.96      0.95        51
           1       0.98      0.96      0.97        82

    accuracy                           0.96       133
   macro avg       0.96      0.96      0.96       133
weighted avg       0.96      0.96      0.96       133
```

In [16]:

```
from sklearn.metrics import f1_score
f1_macro=f1_score(y_test,grid_predictions,average='weighted')
print("The f1_macro value for best parameter {}:".format(grid.best_
```

```
The f1_macro value for best parameter {'C': 10, 'gamma': 'auto', 'kern
el': 'linear'}: 0.9624731911379498
```

In [17]:

```
from sklearn.metrics import roc_auc_score
y_scores = grid.decision_function(X_test)
roc_auc_score=roc_auc_score(y_test,y_scores)
print("The roc_auc_score is:",roc_auc_score)
```

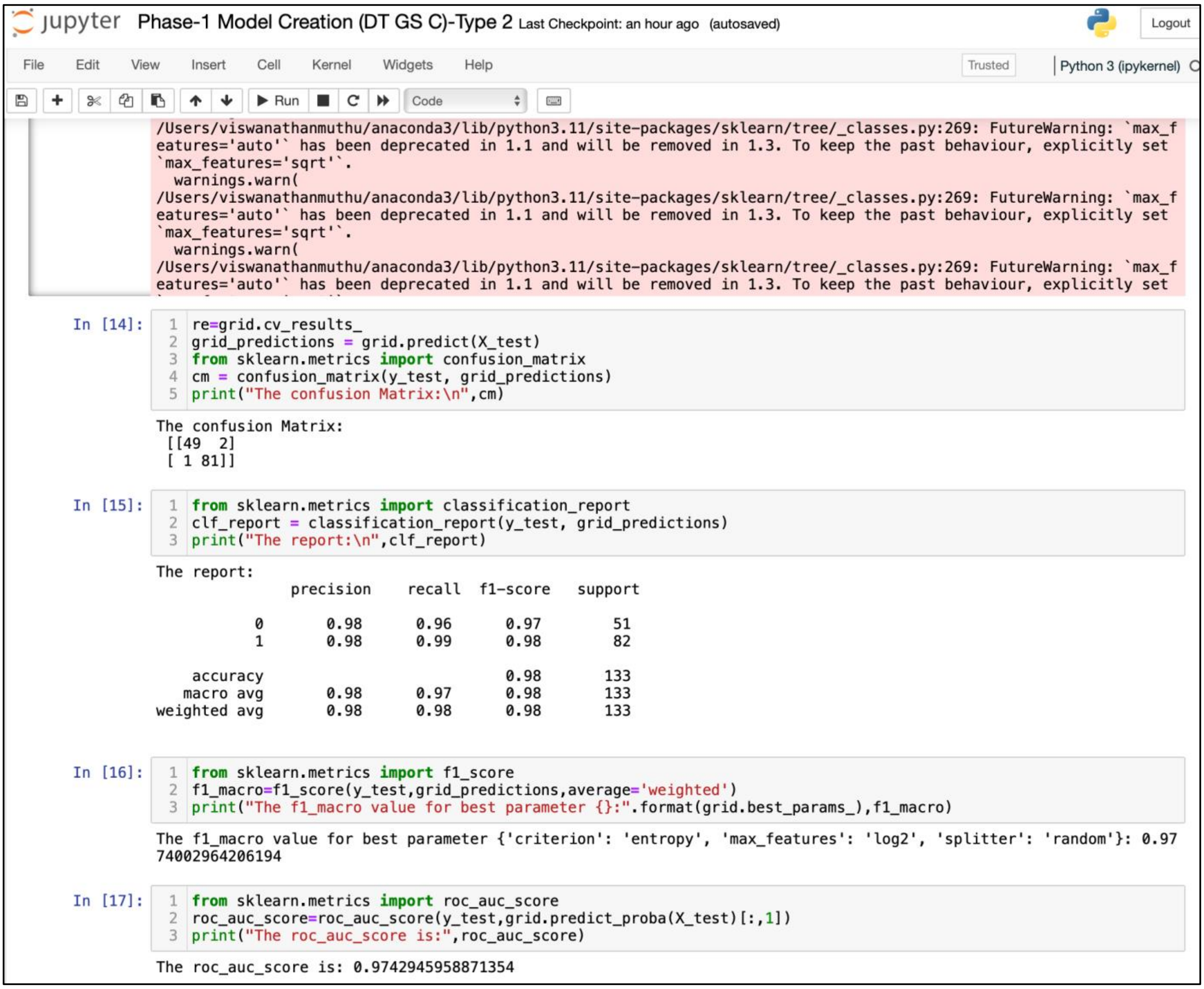
```
The roc_auc_score is: 0.996652319464371
```

2. Decision Tree

Inference :

Accuracy = 0.98
F1 Macro = 0.9774 for Best Parameters = 'criterion': 'entropy', 'max_features': 'log2', 'splitter': 'random'
ROC – AUC Score = 0.9743

Proof:



```
/Users/viswanathanmuthu/anaconda3/lib/python3.11/site-packages/sklearn/tree/_classes.py:269: FutureWarning: `max_f
eatures='auto'` has been deprecated in 1.1 and will be removed in 1.3. To keep the past behaviour, explicitly set
`max_features='sqrt'`.
  warnings.warn(
/Users/viswanathanmuthu/anaconda3/lib/python3.11/site-packages/sklearn/tree/_classes.py:269: FutureWarning: `max_f
eatures='auto'` has been deprecated in 1.1 and will be removed in 1.3. To keep the past behaviour, explicitly set
`max_features='sqrt'`.
  warnings.warn(
/Users/viswanathanmuthu/anaconda3/lib/python3.11/site-packages/sklearn/tree/_classes.py:269: FutureWarning: `max_f
eatures='auto'` has been deprecated in 1.1 and will be removed in 1.3. To keep the past behaviour, explicitly set
```

In [14]:

```
re=grid.cv_results_
grid_predictions = grid.predict(X_test)
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, grid_predictions)
print("The confusion Matrix:\n",cm)
```

```
The confusion Matrix:
 [[49  2]
 [ 1 81]]
```

In [15]:

```
from sklearn.metrics import classification_report
clf_report = classification_report(y_test, grid_predictions)
print("The report:\n",clf_report)
```

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

           0       0.98      0.96      0.97        51
           1       0.98      0.99      0.98        82

    accuracy                           0.98       133
   macro avg       0.98      0.97      0.98       133
weighted avg       0.98      0.98      0.98       133
```

In [16]:

```
from sklearn.metrics import f1_score
f1_macro=f1_score(y_test,grid_predictions,average='weighted')
print("The f1_macro value for best parameter {}:".format(grid.best_params_),f1_macro)
```

```
The f1_macro value for best parameter {'criterion': 'entropy', 'max_features': 'log2', 'splitter': 'random'}: 0.97
74002964206194
```

In [17]:

```
from sklearn.metrics import roc_auc_score
roc_auc_score=roc_auc_score(y_test,grid.predict_proba(X_test)[:,1])
print("The roc_auc_score is:",roc_auc_score)
```

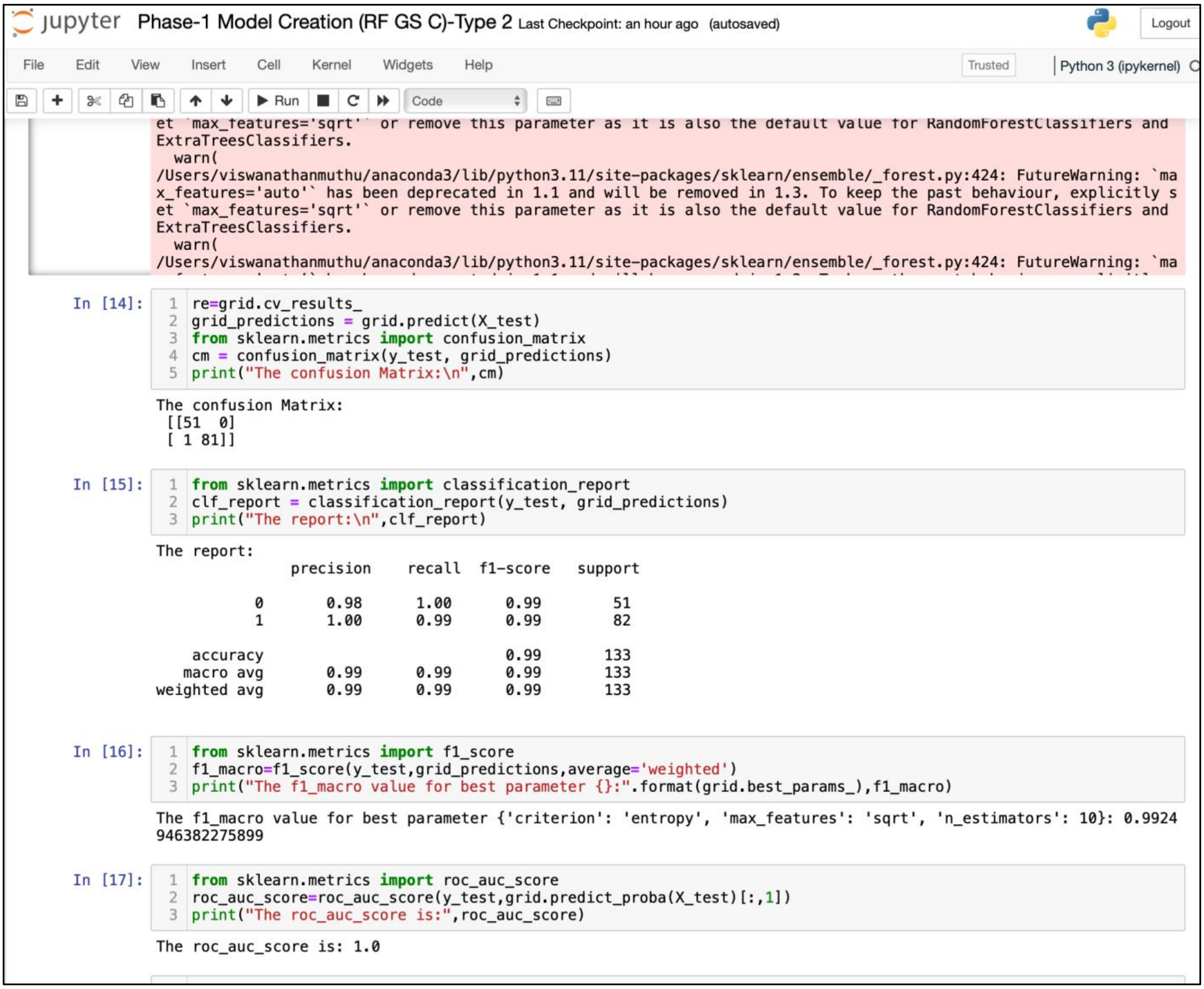
```
The roc_auc_score is: 0.9742945958871354
```

3. Random Forest

Inference :

Accuracy = 0.99
F1 Macro = 0.9925 for Best Parameters = 'criterion': 'entropy', 'max_features': 'sqrt', 'n_estimators': 10
ROC – AUC Score = 1

Proof:



```
et `max_features='sqrt'` or remove this parameter as it is also the default value for RandomForestClassifiers and
ExtraTreesClassifiers.
  warn(
/Users/viswanathanmuthu/anaconda3/lib/python3.11/site-packages/sklearn/ensemble/_forest.py:424: FutureWarning: `ma
x_features='auto'` has been deprecated in 1.1 and will be removed in 1.3. To keep the past behaviour, explicitly s
et `max_features='sqrt'` or remove this parameter as it is also the default value for RandomForestClassifiers and
ExtraTreesClassifiers.
  warn(
/Users/viswanathanmuthu/anaconda3/lib/python3.11/site-packages/sklearn/ensemble/_forest.py:424: FutureWarning: `ma
```

In [14]:

```
re=grid.cv_results_
grid_predictions = grid.predict(X_test)
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, grid_predictions)
print("The confusion Matrix:\n",cm)
```

```
The confusion Matrix:
 [[51  0]
 [ 1 81]]
```

In [15]:

```
from sklearn.metrics import classification_report
clf_report = classification_report(y_test, grid_predictions)
print("The report:\n",clf_report)
```

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

           0       0.98      1.00      0.99        51
           1       1.00      0.99      0.99        82

    accuracy                           0.99       133
   macro avg       0.99      0.99      0.99       133
weighted avg       0.99      0.99      0.99       133
```

In [16]:

```
from sklearn.metrics import f1_score
f1_macro=f1_score(y_test,grid_predictions,average='weighted')
print("The f1_macro value for best parameter {}:".format(grid.best_params_),f1_macro)
```

```
The f1_macro value for best parameter {'criterion': 'entropy', 'max_features': 'sqrt', 'n_estimators': 10}: 0.9924
946382275899
```

In [17]:

```
from sklearn.metrics import roc_auc_score
roc_auc_score=roc_auc_score(y_test,grid.predict_proba(X_test)[:,1])
print("The roc_auc_score is:",roc_auc_score)
```

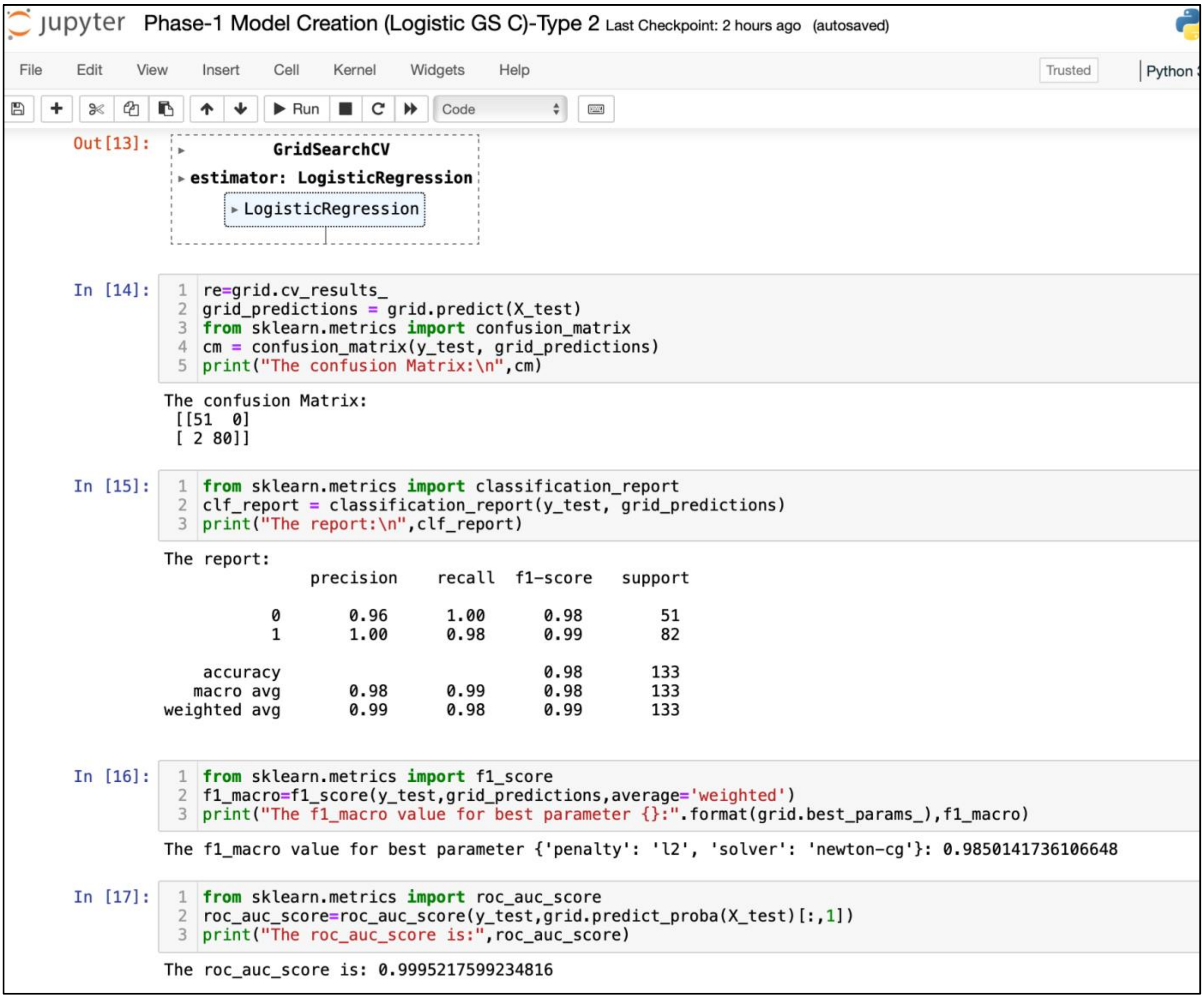
```
The roc_auc_score is: 1.0
```

4.Logistic Regression

Inference :

Accuracy = 0.98
F1 Macro = 0.9850 for Best Parameters = 'penalty': 'l2', 'solver': 'newton-cg'
ROC – AUC Score = 0.9995

Proof:


jupyter
Phase-1 Model Creation (Logistic GS C)-Type 2 Last Checkpoint: 2 hours ago (autosaved)
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3
Run
Code

```
Out[13]:  GridSearchCV
          ▸ estimator: LogisticRegression
              ▸ LogisticRegression
```

```
In [14]:
re=grid.cv_results_
grid_predictions = grid.predict(X_test)
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, grid_predictions)
print("The confusion Matrix:\n",cm)
```

```
The confusion Matrix:
 [[51  0]
 [ 2 80]]
```

```
In [15]:
from sklearn.metrics import classification_report
clf_report = classification_report(y_test, grid_predictions)
print("The report:\n",clf_report)
```

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

           0       0.96      1.00      0.98        51
           1       1.00      0.98      0.99        82

    accuracy                           0.98       133
   macro avg       0.98      0.99      0.98       133
weighted avg       0.99      0.98      0.99       133
```

```
In [16]:
from sklearn.metrics import f1_score
f1_macro=f1_score(y_test,grid_predictions,average='weighted')
print("The f1_macro value for best parameter {}:".format(grid.best_params_),f1_macro)
```

```
The f1_macro value for best parameter {'penalty': 'l2', 'solver': 'newton-cg'}: 0.9850141736106648
```

```
In [17]:
from sklearn.metrics import roc_auc_score
roc_auc_score=roc_auc_score(y_test,grid.predict_proba(X_test)[:,1])
print("The roc_auc_score is:",roc_auc_score)
```

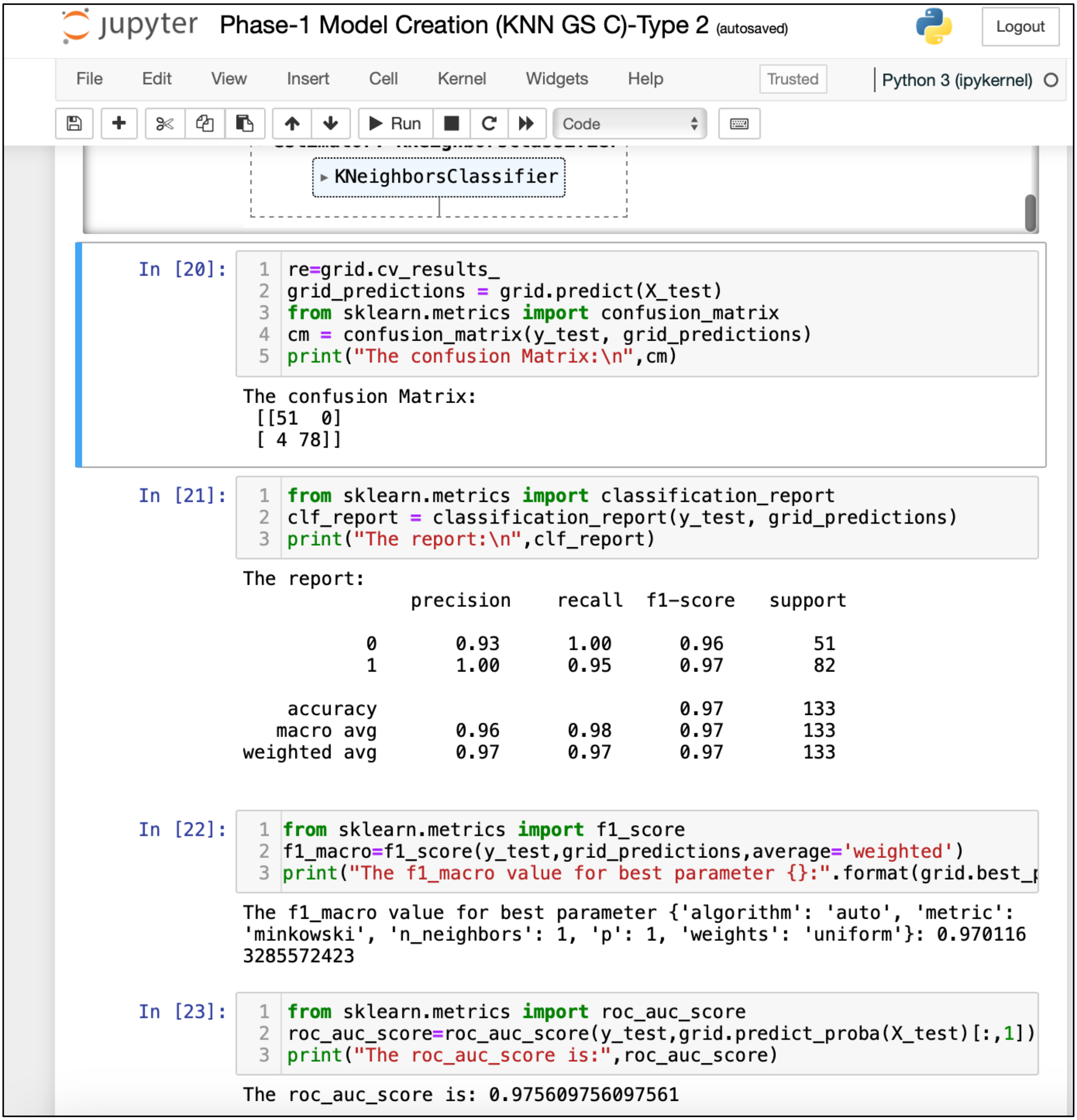
```
The roc_auc_score is: 0.9995217599234816
```

5.KNN

Inference:

Accuracy = 0.97
F1 Macro = 0.9701 for Best Parameters = 'algorithm': 'auto', 'metric': 'minkowski', 'n_neighbors': 1, 'p': 1, 'weights': 'uniform'
ROC – AUC Score = 0.9756

Proof



In [20]:
```
re=grid.cv_results_
grid_predictions = grid.predict(X_test)
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, grid_predictions)
print("The confusion Matrix:\n",cm)
```

```
The confusion Matrix:
 [[51  0]
 [ 4 78]]
```

In [21]:
```
from sklearn.metrics import classification_report
clf_report = classification_report(y_test, grid_predictions)
print("The report:\n",clf_report)
```

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

           0       0.93      1.00      0.96        51
           1       1.00      0.95      0.97        82

    accuracy                           0.97       133
   macro avg       0.96      0.98      0.97       133
weighted avg       0.97      0.97      0.97       133
```

In [22]:
```
from sklearn.metrics import f1_score
f1_macro=f1_score(y_test,grid_predictions,average='weighted')
print("The f1_macro value for best parameter {}:".format(grid.best_
```

```
The f1_macro value for best parameter {'algorithm': 'auto', 'metric':
'minkowski', 'n_neighbors': 1, 'p': 1, 'weights': 'uniform'}: 0.970116
3285572423
```

In [23]:
```
from sklearn.metrics import roc_auc_score
roc_auc_score=roc_auc_score(y_test,grid.predict_proba(X_test)[:,1])
print("The roc_auc_score is:",roc_auc_score)
```

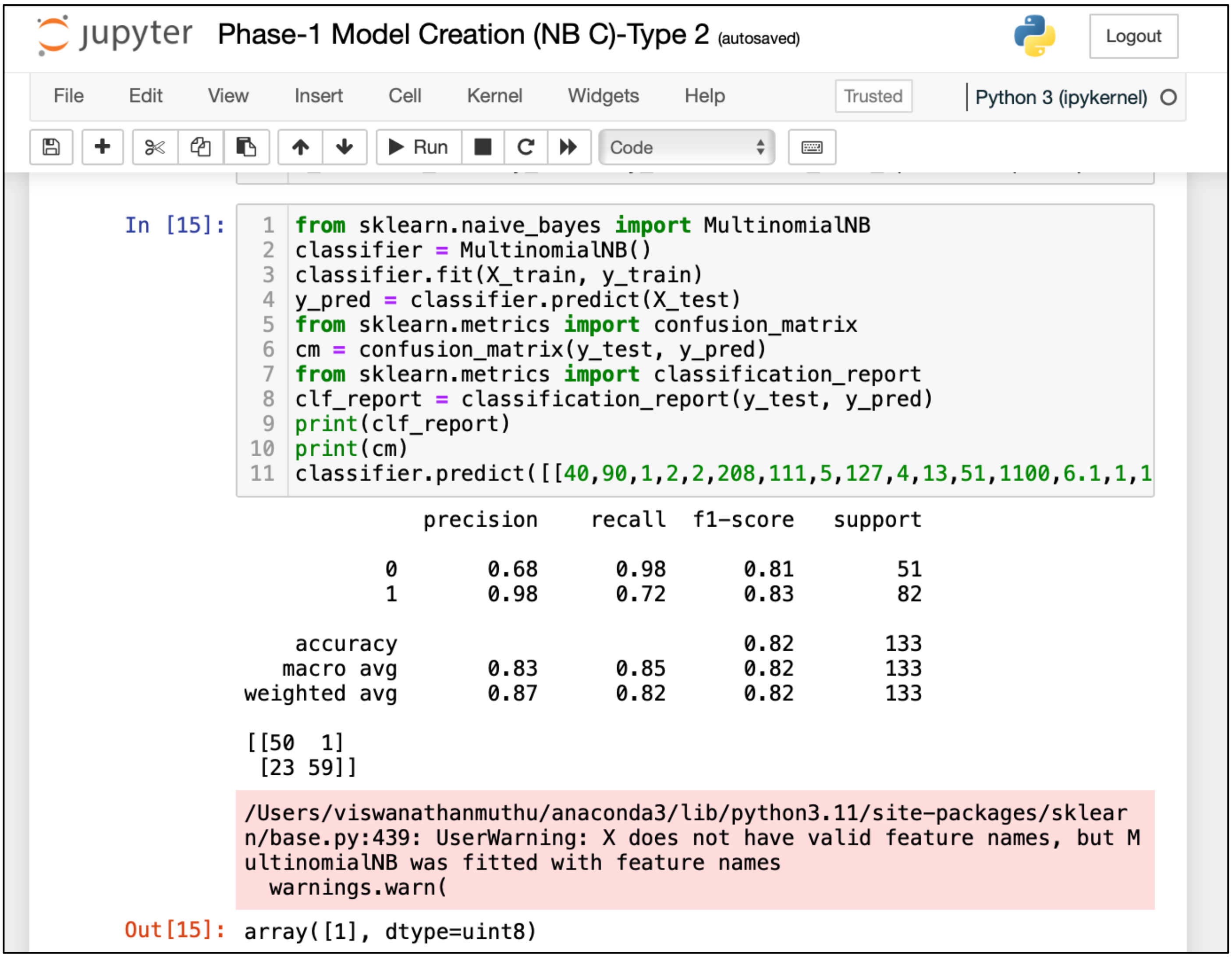
```
The roc_auc_score is: 0.975609756097561
```

6.NB (Multinomial, Bernoulli, Complement)

Multinomial NB Inference:

Accuracy = 0.82

Proof



```
from sklearn.naive_bayes import MultinomialNB
classifier = MultinomialNB()
classifier.fit(X_train, y_train)
y_pred = classifier.predict(X_test)
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
from sklearn.metrics import classification_report
clf_report = classification_report(y_test, y_pred)
print(clf_report)
print(cm)
classifier.predict([[40,90,1,2,2,208,111,5,127,4,13,51,1100,6.1,1,1
```

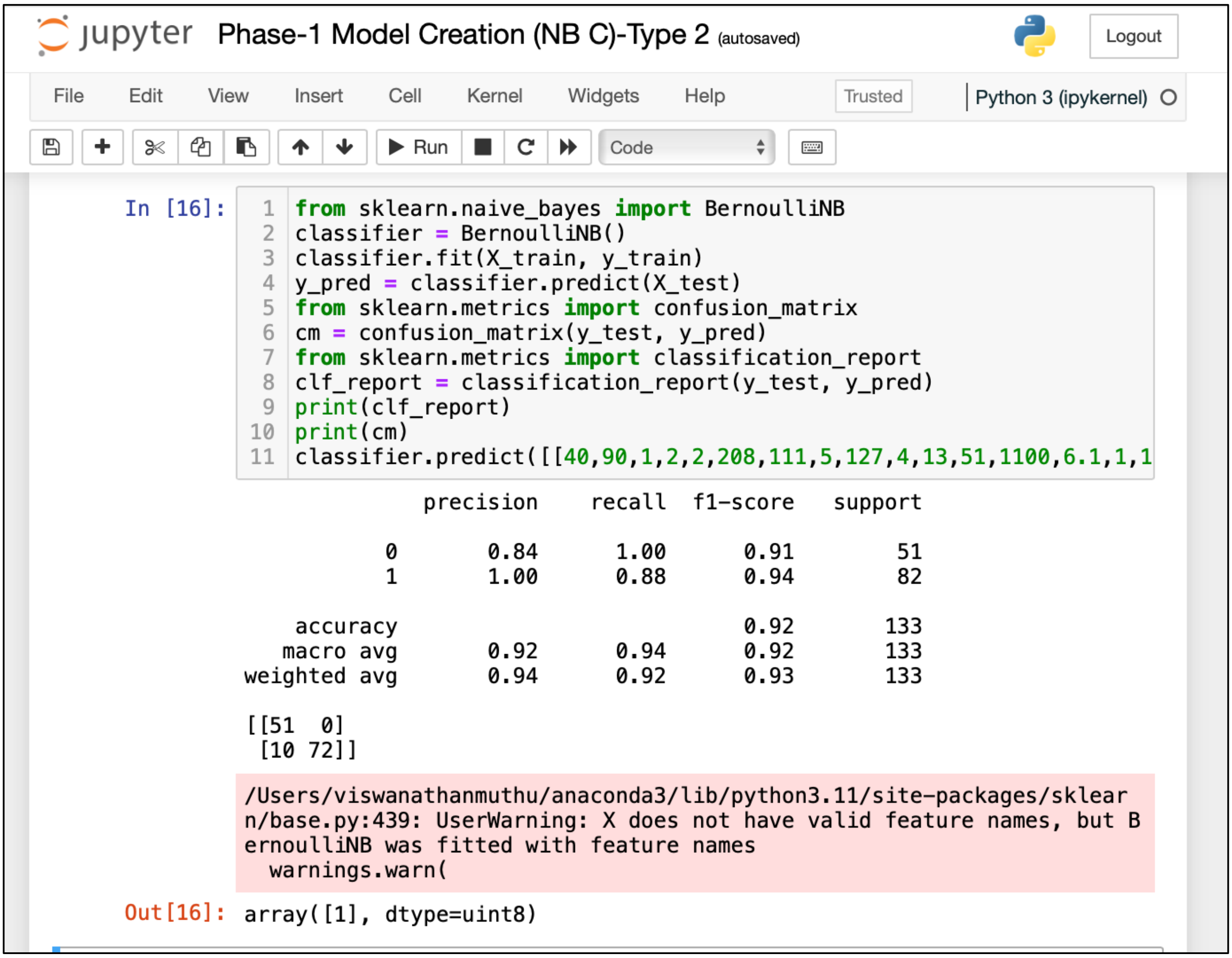
```
              precision    recall  f1-score   support

           0       0.68      0.98      0.81        51
           1       0.98      0.72      0.83        82

    accuracy                           0.82       133
   macro avg       0.83      0.85      0.82       133
weighted avg       0.87      0.82      0.82       133

[[50  1]
 [23 59]]
/Users/viswanathanmuthu/anaconda3/lib/python3.11/site-packages/sklear
n/base.py:439: UserWarning: X does not have valid feature names, but M
ultinomialNB was fitted with feature names
  warnings.warn(
```

```
array([1], dtype=uint8)
```

Bernoulli NB Inference:

Accuracy = **0.92**

Proof



In [16]:

```python
from sklearn.naive_bayes import BernoulliNB
classifier = BernoulliNB()
classifier.fit(X_train, y_train)
y_pred = classifier.predict(X_test)
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
from sklearn.metrics import classification_report
clf_report = classification_report(y_test, y_pred)
print(clf_report)
print(cm)
classifier.predict([[40,90,1,2,2,208,111,5,127,4,13,51,1100,6.1,1,1
```

```
              precision    recall  f1-score   support

           0       0.84      1.00      0.91        51
           1       1.00      0.88      0.94        82

    accuracy                           0.92       133
   macro avg       0.92      0.94      0.92       133
weighted avg       0.94      0.92      0.93       133

[[51  0]
 [10 72]]
```

```
/Users/viswanathanmuthu/anaconda3/lib/python3.11/site-packages/sklear
n/base.py:439: UserWarning: X does not have valid feature names, but B
ernoulliNB was fitted with feature names
  warnings.warn(
```

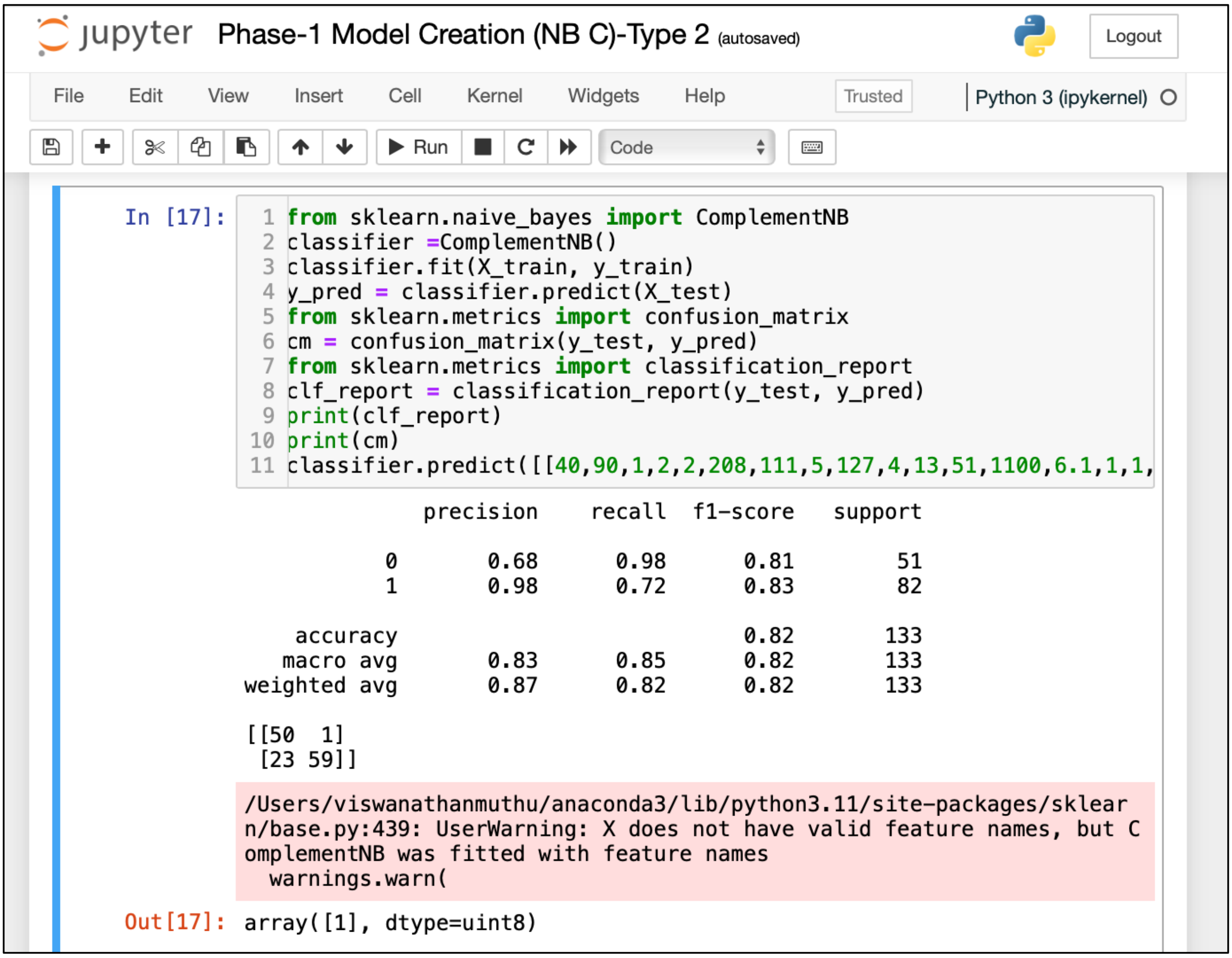
Out[16]:

```
array([1], dtype=uint8)
```

Complement NB Inference:

Accuracy = **0.82**

Proof



In [17]:

```python
from sklearn.naive_bayes import ComplementNB
classifier =ComplementNB()
classifier.fit(X_train, y_train)
y_pred = classifier.predict(X_test)
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
from sklearn.metrics import classification_report
clf_report = classification_report(y_test, y_pred)
print(clf_report)
print(cm)
classifier.predict([[40,90,1,2,2,208,111,5,127,4,13,51,1100,6.1,1,1,
```

```
              precision    recall  f1-score   support

           0       0.68      0.98      0.81        51
           1       0.98      0.72      0.83        82

    accuracy                           0.82       133
   macro avg       0.83      0.85      0.82       133
weighted avg       0.87      0.82      0.82       133

[[50  1]
 [23 59]]
/Users/viswanathanmuthu/anaconda3/lib/python3.11/site-packages/sklear
n/base.py:439: UserWarning: X does not have valid feature names, but C
omplementNB was fitted with feature names
  warnings.warn(
```

Out[17]: `array([1], dtype=uint8)`

Summary

Algorithm		HTP	Best Accuracy	Best F1- Macro Value	Best ROC-AUC Score
SVM		'C': 10, 'gamma': 'auto', ' kernel': 'linear'	0.96	0.9625	0.9966
DT		'criterion': 'entropy', 'max_features': 'log2', 'splitter': 'random'	0.98	0.9774	0.9743
RF		'criterion': 'entropy', 'max_features': 'sqrt', 'n_estimators': 10	0.99	0.9925	1
Logistic Regression		'penalty': 'l2', 'solver': 'newton-cg'	0.98	0.9850	0.9995
KNN		'algorithm': 'auto', 'metric': 'minkowski', 'n_neighbors': 1, 'p': 1, 'weights': 'uniform'	0.97	0.9701	0.9756
NB	Multinomial NB		0.82		
	Bernoulli NB		0.92		
	Complement NB		0.82		
	Categorical NB		-		

Result Analysis:

For the given dataset **RF Classification** algorithm for **HTP criterion** = '**criterion': 'entropy', 'max_features': 'sqrt', 'n_estimators': 10** suits the best with a maximum **accuracy value** = **0.99 , F1- Macro Value = 0.9925 & ROC AUC Score = 1** when compared to the models created by other classification algorithms.

Appendix:

Abbreviations	Expansion
SVM	Support Vector Machine
DT	Decision Tree
RF	Random Forest
KNN	K Nearest Neighbour
NB	Naïve Bayes
HTP	Hyper Tuning Parameters

6. Final Model :

For the given dataset **RF Classification** algorithm for **HTP criterion** = '**criterion': 'entropy', 'max_features': 'sqrt', 'n_estimators': 10** suits the best with a maximum **accuracy value** = **0.99 , F1- Macro Value = 0.9925 & ROC AUC Score = 1** when compared to the models created by other classification algorithms.