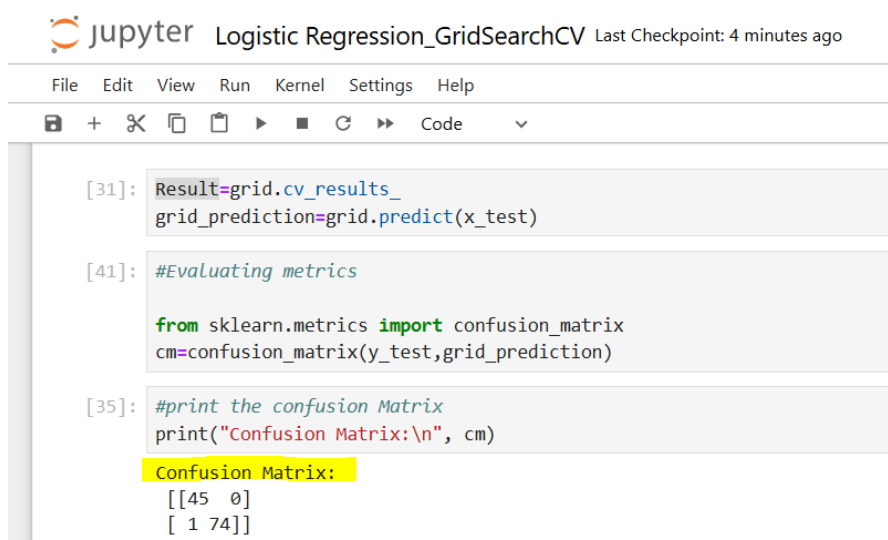


Evaluation Metrics Report for Different Algorithms

1. Logistic Regression

	precision	Recall	F1-score	Support
False	0.98	1.00	0.99	45
True	1.00	0.99	0.99	75
accuracy			0.99	120
Macro Avg	0.99	0.99	0.99	120
Weighted Avg	0.99	0.99	0.99	120

Confusion Matrix Screenshot



```
jupyter Logistic Regression_GridSearchCV Last Checkpoint: 4 minutes ago
File Edit View Run Kernel Settings Help
+ ✂ 📄 📌 ▶ ■ ↺ ▶▶ Code ▼

[31]: Result=grid.cv_results_
      grid_prediction=grid.predict(x_test)

[41]: #Evaluating metrics

      from sklearn.metrics import confusion_matrix
      cm=confusion_matrix(y_test,grid_prediction)

[35]: #print the confusion Matrix
      print("Confusion Matrix:\n", cm)

Confusion Matrix:
[[45  0]
 [ 1 74]]
```

Classification Report Screenshot

Jupyter Logistic Regression_GridSearchCV Last Checkpoint: 5 minutes ago

File Edit View Run Kernel Settings Help

+ ✂ 📄 📄 ▶ ■ ↺ ▶▶ Code ▾

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

[39]: *#print the classification report*
print("classification report:\n",clf_report)

classification report:

	precision	recall	f1-score	support
False	0.98	1.00	0.99	45
True	1.00	0.99	0.99	75
accuracy			0.99	120
macro avg	0.99	0.99	0.99	120
weighted avg	0.99	0.99	0.99	120

Accuracy = 0.99

ROC AUC Score Screenshot

Jupyter Logistic Regression_GridSearchCV Last Checkpoint: 6 minutes ago

File Edit View Run Kernel Settings Help

+ ✂ 📄 📄 ▶ ■ ↺ ▶▶ Code ▾

```
classification_report:
      precision    recall  f1-score   support

 False       0.98      1.00      0.99        45
  True       1.00      0.99      0.99        75

 accuracy          0.99          120
 macro avg       0.99      0.99      0.99        120
 weighted avg    0.99      0.99      0.99        120
```

[43]: *# Evaluation Metrics*

```
from sklearn.metrics import roc_auc_score
roc_auc_score(y_test,grid.predict_proba(x_test)[:,:1])
```

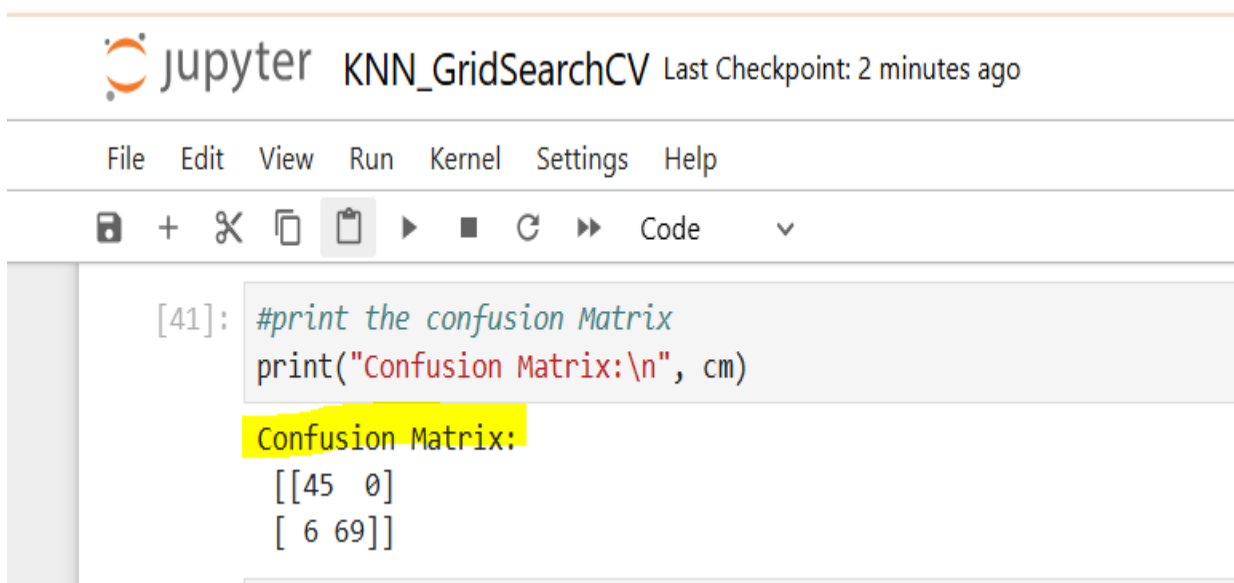
[43]: 1.0

roc_auc_score = 1.0

2. KNN

	precision	Recall	F1-score	Support
False	0.88	1.00	0.94	45
True	1.00	0.92	0.96	75
Accuracy			0.95	120
Macro Avg	0.94	0.96	0.95	120
Weighted Avg	0.96	0.95	0.95	120

Confusion Matrix Screenshot



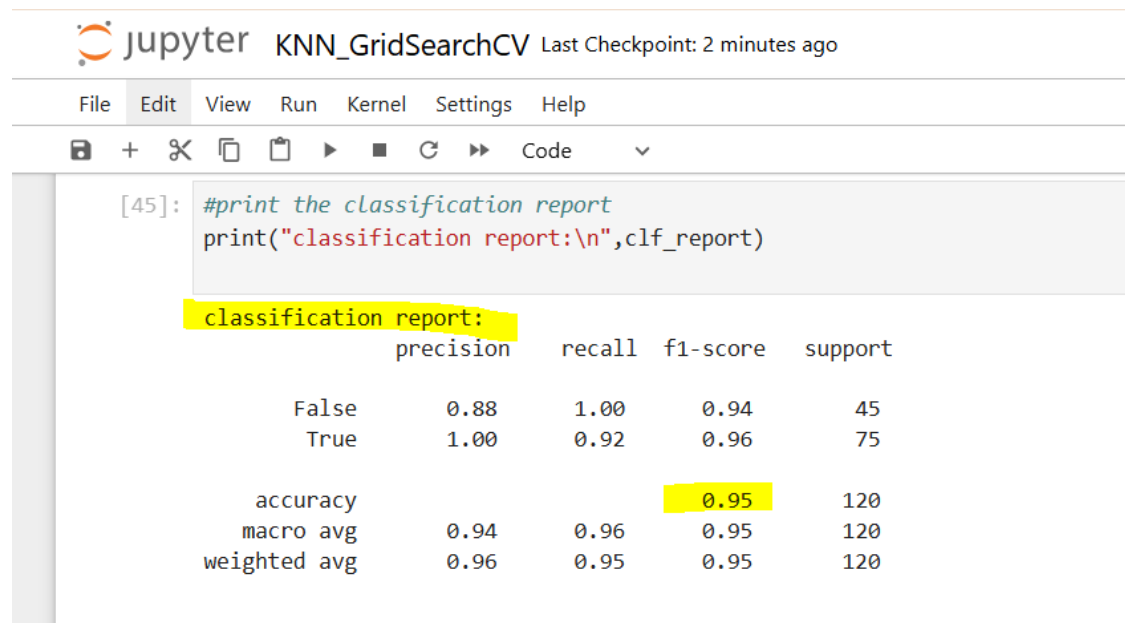
The screenshot shows a Jupyter Notebook interface for a file named 'KNN_GridSearchCV'. The last checkpoint was 2 minutes ago. The menu bar includes File, Edit, View, Run, Kernel, Settings, and Help. The toolbar contains icons for saving, adding, deleting, copying, pasting, running, and other actions. The code cell [41] contains the following Python code:

```
[41]: #print the confusion Matrix
print("Confusion Matrix:\n", cm)
```

The output of the code is displayed below the code cell:

```
Confusion Matrix:
[[45  0]
 [ 6 69]]
```

Classification Report Screenshot



A screenshot of a Jupyter Notebook interface. The top bar shows the Jupyter logo, the notebook name 'KNN_GridSearchCV', and 'Last Checkpoint: 2 minutes ago'. The menu bar includes 'File', 'Edit', 'View', 'Run', 'Kernel', 'Settings', and 'Help'. Below the menu is a toolbar with icons for saving, adding, deleting, copying, pasting, running, and code completion. The code cell [45] contains the following Python code:

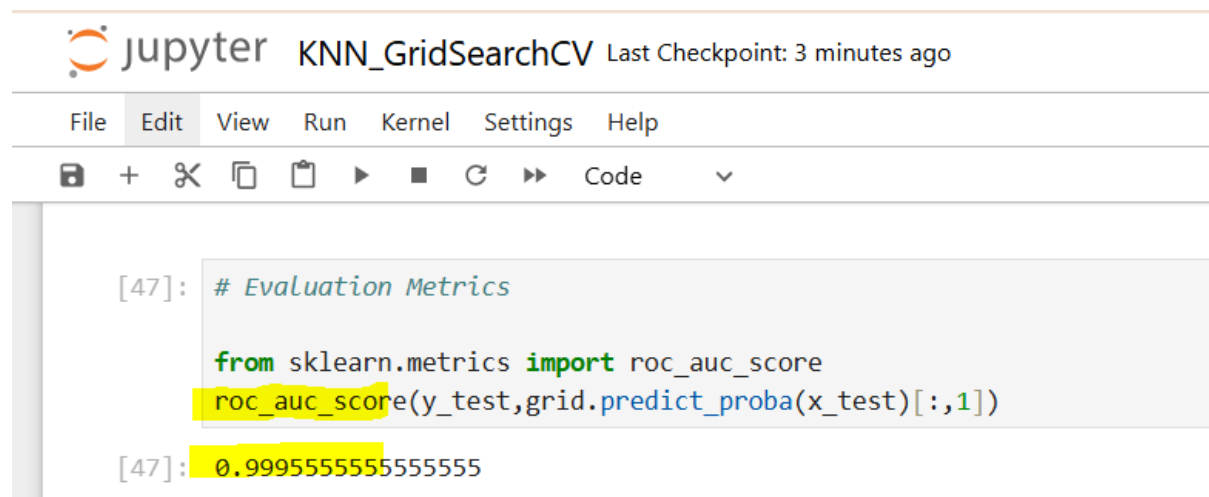
```
[45]: #print the classification report
print("classification report:\n",clf_report)
```

The output of the code is a classification report displayed as a table:

	precision	recall	f1-score	support
False	0.88	1.00	0.94	45
True	1.00	0.92	0.96	75
accuracy			0.95	120
macro avg	0.94	0.96	0.95	120
weighted avg	0.96	0.95	0.95	120

Accuracy = 0.95

ROC AUC Score Screenshot



A screenshot of a Jupyter Notebook interface. The top bar shows the Jupyter logo, the notebook name 'KNN_GridSearchCV', and 'Last Checkpoint: 3 minutes ago'. The menu bar includes 'File', 'Edit', 'View', 'Run', 'Kernel', 'Settings', and 'Help'. Below the menu is a toolbar with icons for saving, adding, deleting, copying, pasting, running, and code completion. The code cell [47] contains the following Python code:

```
[47]: # Evaluation Metrics

from sklearn.metrics import roc_auc_score
roc_auc_score(y_test,grid.predict_proba(x_test)[:,:1])
```

The output of the code is the ROC AUC score:

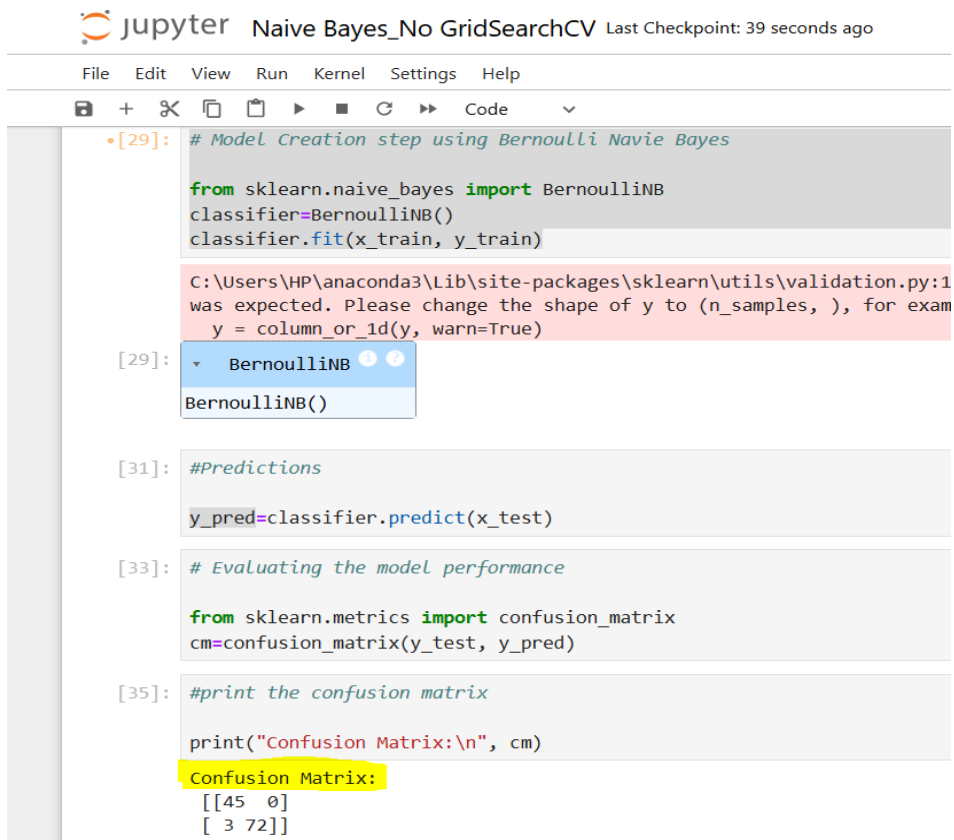
```
[47]: 0.9995555555555555
```

roc_auc_score =0.99

3. Naive Bayes

	precision	Recall	F1-score	Support
False	0.94	1.00	0.97	45
True	1.00	0.96	0.98	75
Accuracy			0.97	120
Macro Avg	0.97	0.98	0.97	120
Weighted Avg	0.98	0.97	0.98	120

Confusion Matrix Screenshot



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

```
•[29]: # Model Creation step using Bernoulli Navie Bayes

from sklearn.naive_bayes import BernoulliNB
classifier=BernoulliNB()
classifier.fit(x_train, y_train)

C:\Users\HP\anaconda3\Lib\site-packages\sklearn\utils\validation.py:1
was expected. Please change the shape of y to (n_samples, ), for exam
y = column_or_1d(y, warn=True)

[29]: ▾ BernoulliNB ⓘ ⓘ
      BernoulliNB()

[31]: #Predictions

y_pred=classifier.predict(x_test)

[33]: # Evaluating the model performance

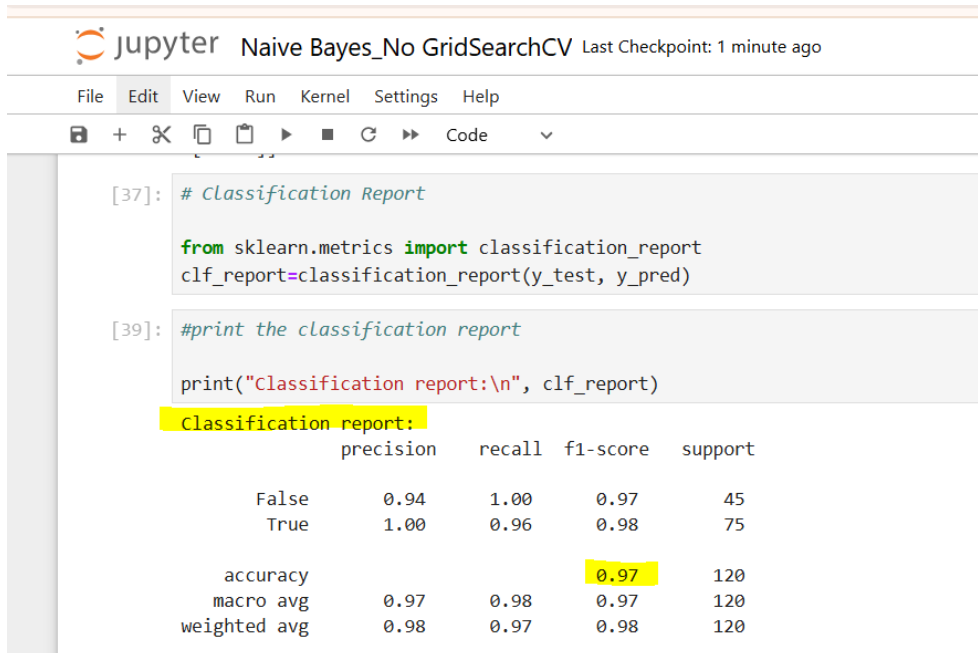
from sklearn.metrics import confusion_matrix
cm=confusion_matrix(y_test, y_pred)

[35]: #print the confusion matrix

print("Confusion Matrix:\n", cm)

Confusion Matrix:
[[45  0]
 [ 3 72]]
```

Classification Report Screenshot



A Jupyter Notebook interface with the title 'Naive Bayes_No GridSearchCV' and 'Last Checkpoint: 1 minute ago'. The menu bar includes File, Edit, View, Run, Kernel, Settings, and Help. The toolbar shows icons for saving, adding, deleting, copying, pasting, running, and code execution. The code cell [37] contains the following Python code:

```
[37]: # Classification Report

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

The code cell [39] contains the following Python code:

```
[39]: #print the classification report

print("Classification report:\n", clf_report)
```

The output of the code cell [39] is a classification report:

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

      False      0.94      1.00      0.97         45
       True      1.00      0.96      0.98         75

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

Accuracy = 0.97

ROC AUC Score Screenshot



A Jupyter Notebook interface with the title 'Naive Bayes_No GridSearchCV' and 'Last Checkpoint: 1 minute ago'. The menu bar includes File, Edit, View, Run, Kernel, Settings, and Help. The toolbar shows icons for saving, adding, deleting, copying, pasting, running, and code execution. The code cell [73] contains the following Python code:

```
[73]: # Predicting the probability

y_pred_proba=classifier.predict_proba(x_test)[:,-1]
```

The code cell [75] contains the following Python code:

```
[75]: # Evaluation Metrics

from sklearn.metrics import roc_auc_score
roc_auc_score(y_test,y_pred_proba)
```

The output of the code cell [75] is:

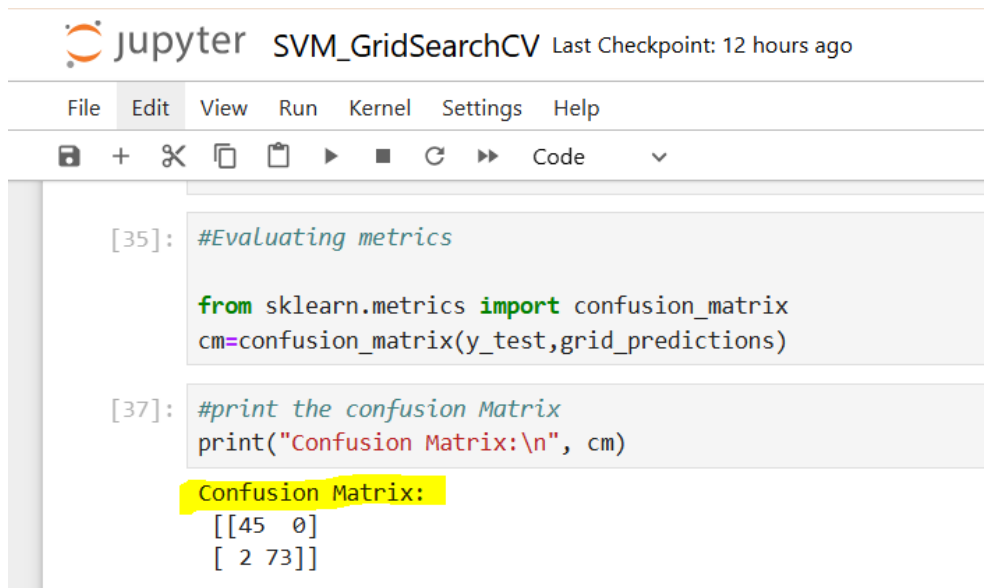
```
[75]: 1.0
```

roc_auc_score =1.0

4. Support Vector Machine

	precision	Recall	F1-score	Support
False	0.96	1.00	0.98	45
True	1.00	0.97	0.99	75
Accuracy			0.98	120
Macro Avg	0.98	0.99	0.98	120
Weighted Avg	0.98	0.98	0.98	120

Confusion Matrix Screenshot



The screenshot shows a Jupyter Notebook window titled "SVM_GridSearchCV" with a "Last Checkpoint: 12 hours ago" status. The interface includes a menu bar (File, Edit, View, Run, Kernel, Settings, Help) and a toolbar with icons for saving, adding, deleting, copying, pasting, and running code. The code cell contains the following Python code:

```
[35]: #Evaluating metrics

from sklearn.metrics import confusion_matrix
cm=confusion_matrix(y_test,grid_predictions)

[37]: #print the confusion Matrix
print("Confusion Matrix:\n", cm)
```

The output of the code is displayed below the cell:

```
Confusion Matrix:
[[45  0]
 [ 2 73]]
```

Classification Report Screenshot

Jupyter SVM_GridSearchCV Last Checkpoint: 12 hours ago

File Edit View Run Kernel Settings Help

+ ✂ 📄 📄 ▶ ■ 🔁 ⏭ Code ▾

```
[ 2 73]]
```

```
[39]: #classification report
      # Evaluation metrics

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

```
[41]: #print the classification report
      print("classification report:\n",clf_report)
```

classification report:

	precision	recall	f1-score	support
False	0.96	1.00	0.98	45
True	1.00	0.97	0.99	75
accuracy			0.98	120
macro avg	0.98	0.99	0.98	120
weighted avg	0.98	0.98	0.98	120

Accuracy = 0.98

ROC AUC Score Screenshot

Jupyter SVM_GridSearchCV Last Checkpoint: 12 hours ago

File Edit View Run Kernel Settings Help

+ ✂ 📄 📄 ▶ ■ 🔁 ⏭ Code ▾

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

```
[41]: #print the classification report
      print("classification report:\n",clf_report)
```

classification report:

	precision	recall	f1-score	support
False	0.96	1.00	0.98	45
True	1.00	0.97	0.99	75
accuracy			0.98	120
macro avg	0.98	0.99	0.98	120
weighted avg	0.98	0.98	0.98	120

```
[57]: # Evaluation Metrics

      from sklearn.metrics import roc_auc_score
      roc_auc_score(y_test,grid.predict_proba(x_test)[:,:1])
```

```
[57]: 0.9997037037037036
```

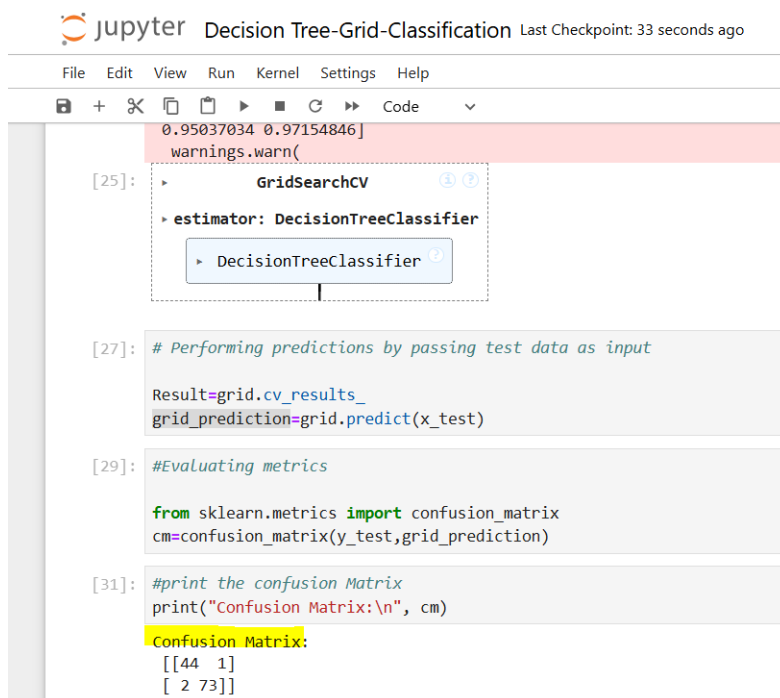
```
[ ]:
```

roc_auc_score =0.99

5. Decision Tree

	precision	Recall	F1-score	Support
False	0.96	0.98	0.97	45
True	0.99	0.97	0.98	75
Accuracy			0.97	120
Macro Avg	0.97	0.98	0.97	120
Weighted Avg	0.98	0.97	0.98	120

Confusion Matrix Screenshot



The screenshot shows a Jupyter Notebook interface with the title "Decision Tree-Grid-Classification" and a last checkpoint of 33 seconds ago. The notebook contains several code cells. The first cell shows a warning message: "0.95037034 0.97154846] warnings.warn()". The second cell, [25], shows a GridSearchCV object with an estimator of DecisionTreeClassifier. The third cell, [27], shows the execution of grid_predictions = grid.predict(x_test). The fourth cell, [29], shows the evaluation of metrics using sklearn.metrics.confusion_matrix. The fifth cell, [31], shows the printing of the confusion matrix, which is displayed as:

```
Confusion Matrix:
[[44  1]
 [ 2 73]]
```

Classification Report Screenshot

jupyter Decision Tree-Grid-Classification Last Checkpoint: 1 minute ago

File Edit View Run Kernel Settings Help

+ ✂ 📄 📌 ▶ ■ ↺ ⏩ Code ▼

```
[33]: #classification report
      # Evaluation metrics

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

[35]: #print the classification report
      print("classification report:\n",clf_report)
```

classification report:

	precision	recall	f1-score	support
False	0.96	0.98	0.97	45
True	0.99	0.97	0.98	75
accuracy			0.97	120
macro avg	0.97	0.98	0.97	120
weighted avg	0.98	0.97	0.98	120

Accuracy = 0.97

ROC AUC Score Screenshot

jupyter Decision Tree-Grid-Classification Last Checkpoint: 1 minute ago

File Edit View Run Kernel Settings Help

+ ✂ 📄 📌 ▶ ■ ↺ ⏩ Code ▼

```
[39]: # Evaluation Metrics

      from sklearn.metrics import roc_auc_score
      roc_auc_score(y_test,grid.predict_proba(x_test)[:,:1])

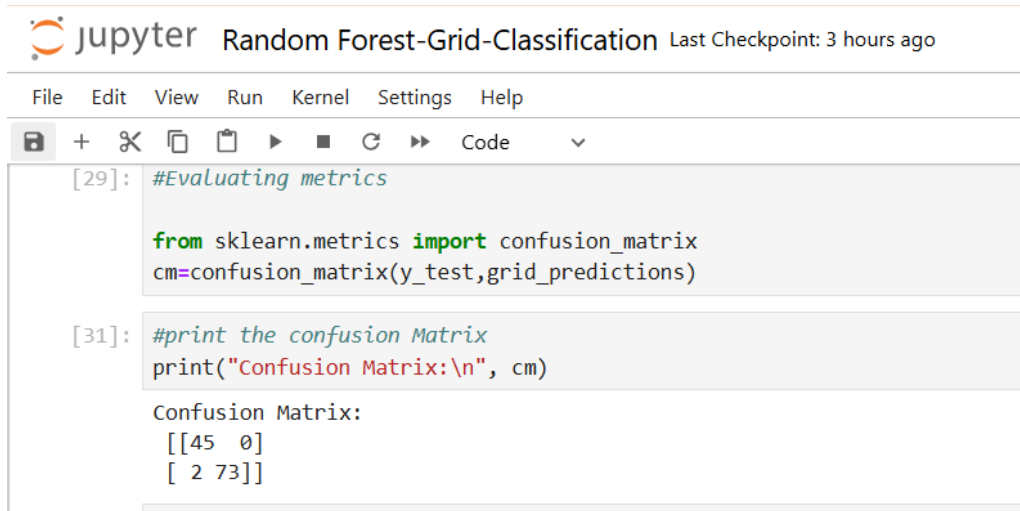
[39]: 0.9755555555555556
```

roc_auc_score =0.97

6. Random Forest

	precision	Recall	F1-score	Support
False	0.96	1.00	0.98	45
True	1.00	0.97	0.99	75
Accuracy			0.98	120
Macro Avg	0.98	0.99	0.98	120
Weighted Avg	0.98	0.98	0.98	120

Confusion Matrix Screenshot



The screenshot shows a Jupyter Notebook interface with the title "Random Forest-Grid-Classification" and a "Last Checkpoint: 3 hours ago" status. The notebook has a menu bar with "File", "Edit", "View", "Run", "Kernel", "Settings", and "Help". Below the menu bar is a toolbar with icons for saving, adding, deleting, copying, pasting, running, and other actions. The code editor shows two cells. The first cell, labeled "[29]:", contains the comment "#Evaluating metrics" and the code `from sklearn.metrics import confusion_matrix` and `cm=confusion_matrix(y_test,grid_predictions)`. The second cell, labeled "[31]:", contains the comment "#print the confusion Matrix" and the code `print("Confusion Matrix:\n", cm)`. The output of the second cell is displayed below the code, showing "Confusion Matrix:" followed by a 2x2 matrix: `[[45 0]` and `[2 73]]`.

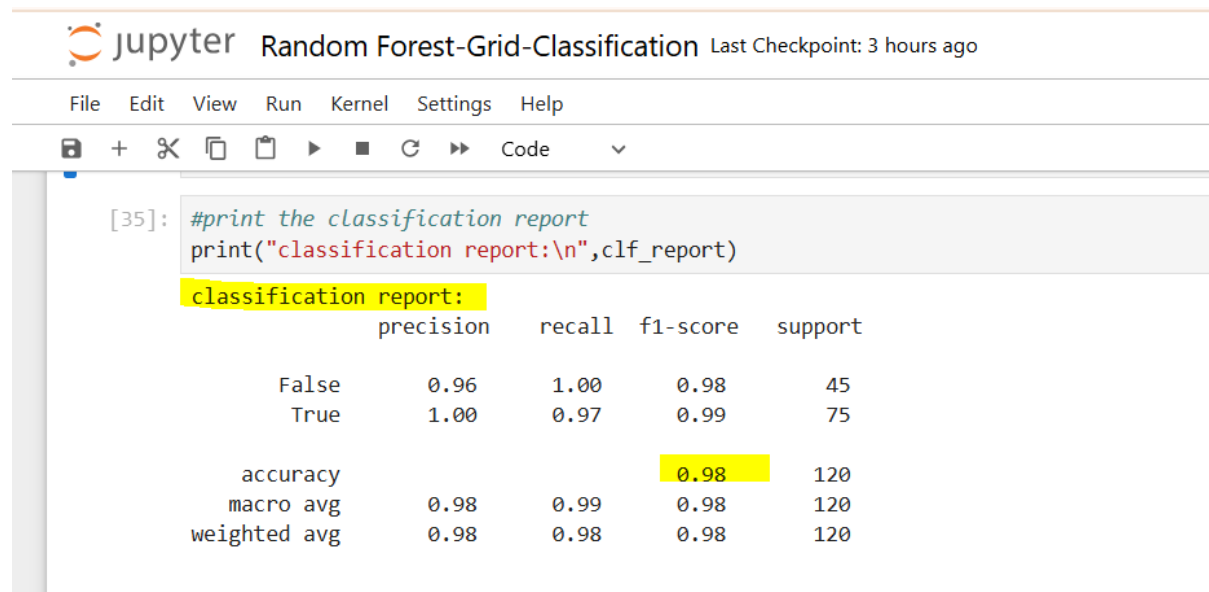
```
[29]: #Evaluating metrics

from sklearn.metrics import confusion_matrix
cm=confusion_matrix(y_test,grid_predictions)

[31]: #print the confusion Matrix
print("Confusion Matrix:\n", cm)

Confusion Matrix:
[[45  0]
 [ 2 73]]
```

Classification Report Screenshot



The screenshot shows a Jupyter Notebook interface with the title "Random Forest-Grid-Classification" and a subtitle "Last Checkpoint: 3 hours ago". The notebook has a menu bar with "File", "Edit", "View", "Run", "Kernel", "Settings", and "Help". Below the menu bar is a toolbar with icons for saving, adding, deleting, copying, pasting, running, and code completion. The code cell [35] contains the following code:

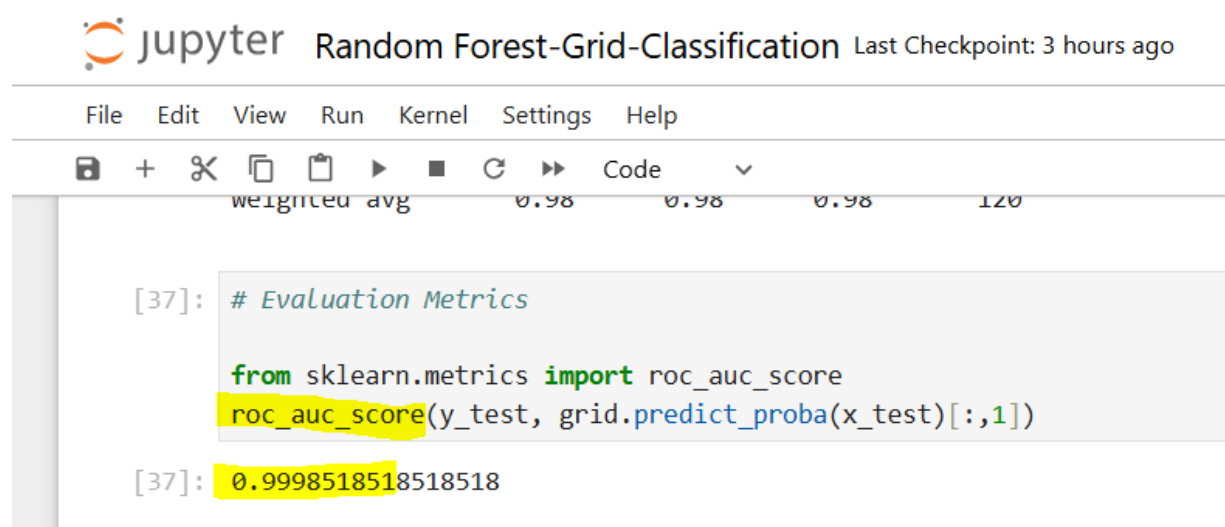
```
[35]: #print the classification report
print("classification report:\n",clf_report)
```

The output of the code is a classification report for a Random Forest model. The report is displayed as a table with the following data:

	precision	recall	f1-score	support
False	0.96	1.00	0.98	45
True	1.00	0.97	0.99	75
accuracy			0.98	120
macro avg	0.98	0.99	0.98	120
weighted avg	0.98	0.98	0.98	120

Accuracy = 0.98

ROC AUC Score Screenshot



The screenshot shows a Jupyter Notebook interface with the title "Random Forest-Grid-Classification" and a subtitle "Last Checkpoint: 3 hours ago". The notebook has a menu bar with "File", "Edit", "View", "Run", "Kernel", "Settings", and "Help". Below the menu bar is a toolbar with icons for saving, adding, deleting, copying, pasting, running, and code completion. The code cell [37] contains the following code:

```
[37]: # Evaluation Metrics

from sklearn.metrics import roc_auc_score
roc_auc_score(y_test, grid.predict_proba(x_test)[:,:1])
```

The output of the code is the ROC AUC score, which is 0.9998518518518518.

roc_auc_score =0.99

Final Model: The results indicate that **Logistic Regression** outperformed alternative AI models in terms of **evaluation metrics**, leading to its selection for deployment.