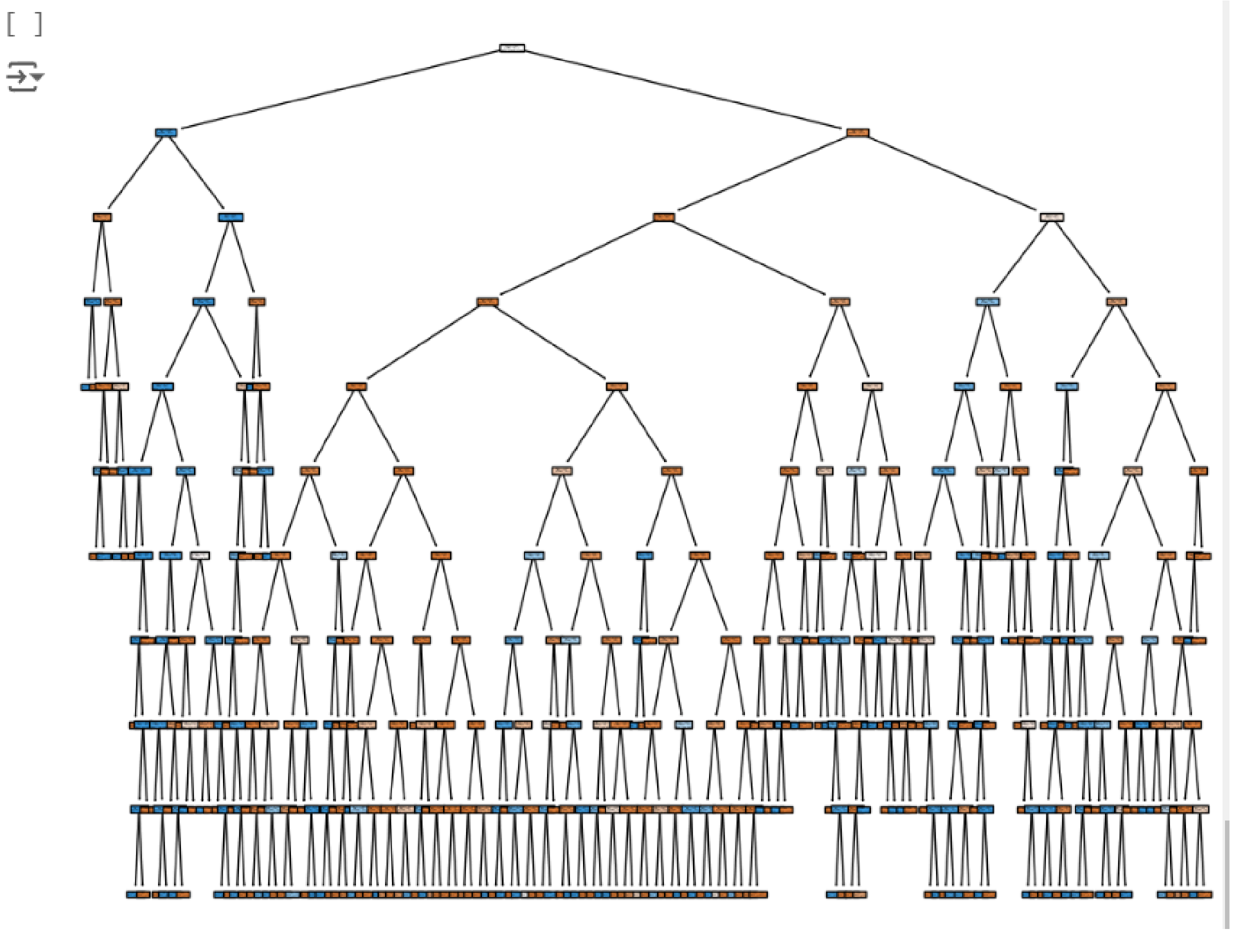
Decision Tree Model for Fraud Detection



1. Dataset and Preprocessing

- **Data Source and Characteristics**: The dataset used is from Kaggle and contains credit card transactions.Total transactions: 284,807 over two days.Fraudulent transactions: 492 (0.172% of total), making it highly imbalanced.Features: 31 columns, where 28 (V1–V28) are transformed features (due to confidentiality).Target: 'Class' variable with 1 indicating fraud and 0 indicating genuine transactions.

- The 'Time' column was dropped, and the 'Class' column (indicating fraudulent vs. non-fraudulent transactions) was designated as the target variable.
- **Data Splitting**: The dataset was split into training and test sets with a 67-33 split.
- **Class Imbalance**: Since the dataset was imbalanced, with many more non-fraud instances, a sampling technique was applied:
  - **Random Undersampling**: A sample of 100,000 data points from the majority class (non-fraud) was taken.
  - **SMOTE (Synthetic Minority Over-sampling Technique)**: SMOTE was used to oversample the minority class to further balance the data.

2. Model Initialization and Training

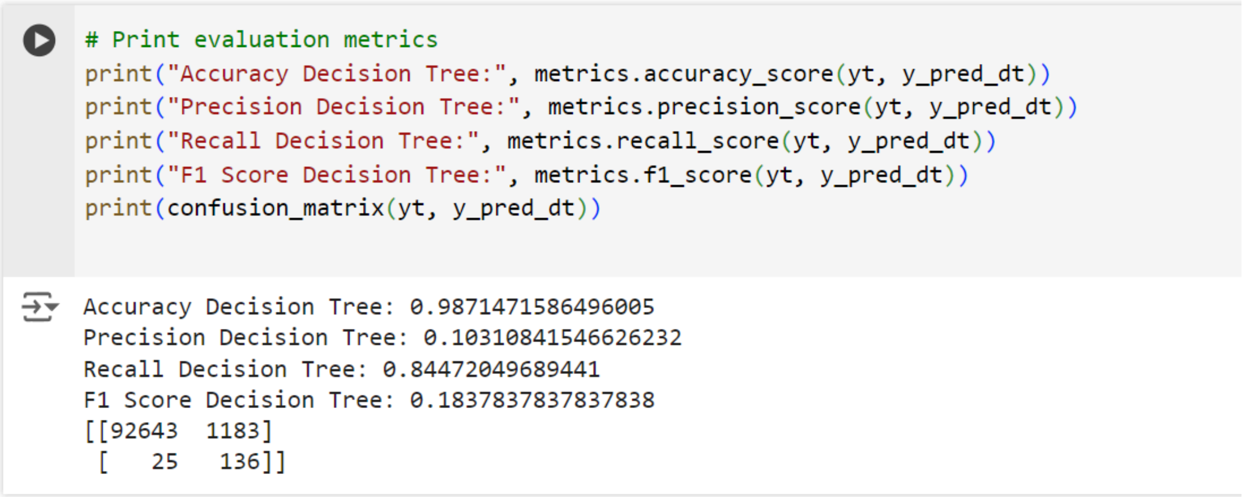
- **Model Type**: A Decision Tree Classifier was initialized with criterion='gini' and a maximum depth of 10.
- **Training**: The model was trained on the resampled training set.

3. Model Evaluation Metrics

The Decision Tree model was evaluated on several metrics, which were as follows:

- **Accuracy**: Represents the percentage of correctly classified instances.
- **Precision**: Measures how many of the predicted fraud cases were actually fraud.
- **Recall**: Reflects how well the model identified actual fraud cases.
- **F1 Score**: The harmonic mean of precision and recall, giving a balanced evaluation of model performance.
- **Confusion Matrix**: Showed the number of true positives, true negatives, false positives, and false negatives.

Here are the evaluation metrics obtained:



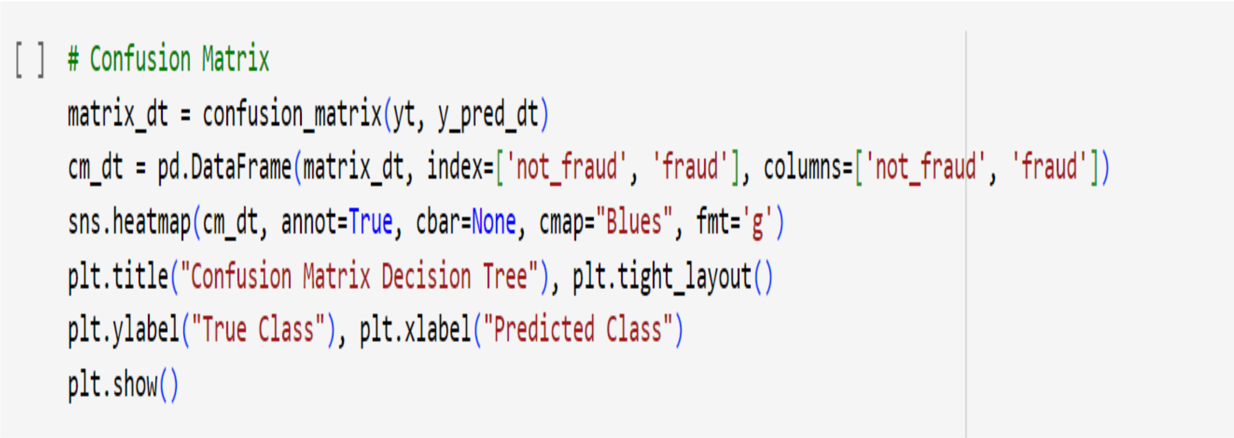
```
# Print evaluation metrics
print("Accuracy Decision Tree:", metrics.accuracy_score(yt, y_pred_dt))
print("Precision Decision Tree:", metrics.precision_score(yt, y_pred_dt))
print("Recall Decision Tree:", metrics.recall_score(yt, y_pred_dt))
print("F1 Score Decision Tree:", metrics.f1_score(yt, y_pred_dt))
print(confusion_matrix(yt, y_pred_dt))
```

```
Accuracy Decision Tree: 0.9871471586496005
Precision Decision Tree: 0.10310841546626232
Recall Decision Tree: 0.84472049689441
F1 Score Decision Tree: 0.1837837837837838
[[92643  1183]
 [   25   136]]
```

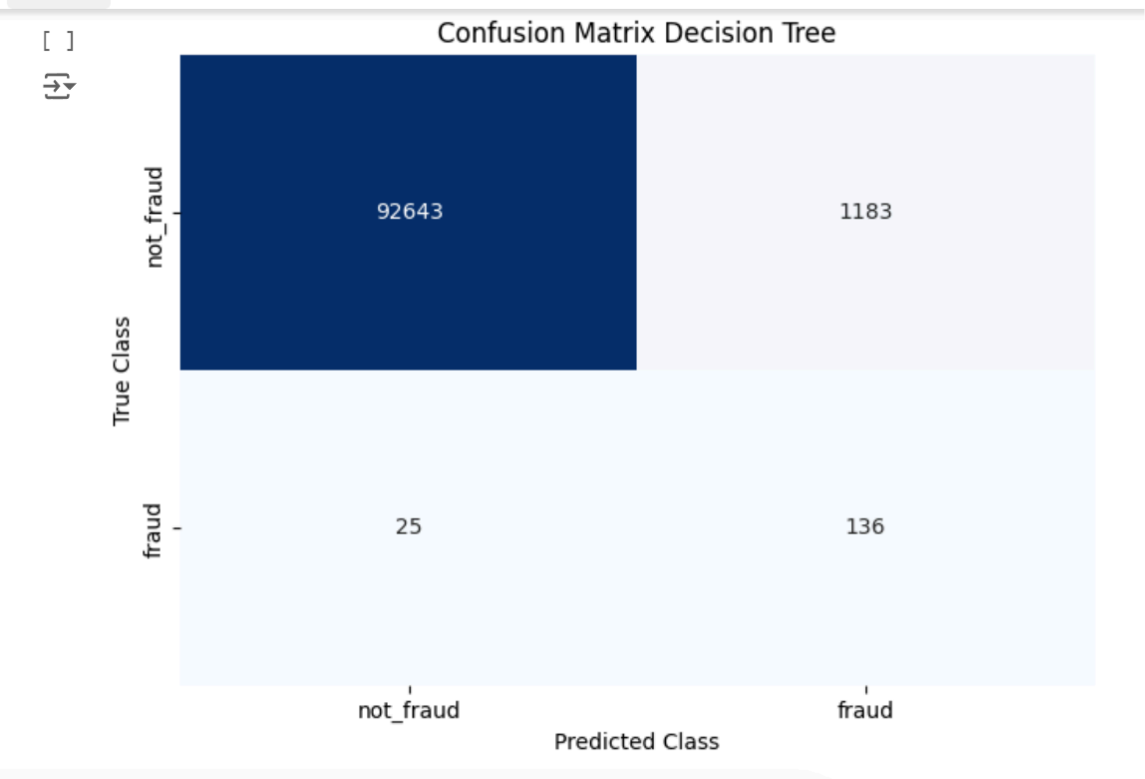
4. Confusion Matrix

The confusion matrix was visualized using a heatmap to show the breakdown of predictions:

- **Non-fraud (True Negative)**
- **Fraud (True Positive)**



```
# Confusion Matrix
matrix_dt = confusion_matrix(yt, y_pred_dt)
cm_dt = pd.DataFrame(matrix_dt, index=['not_fraud', 'fraud'], columns=['not_fraud', 'fraud'])
sns.heatmap(cm_dt, annot=True, cbar=None, cmap="Blues", fmt='g')
plt.title("Confusion Matrix Decision Tree"), plt.tight_layout()
plt.ylabel("True Class"), plt.xlabel("Predicted Class")
plt.show()
```

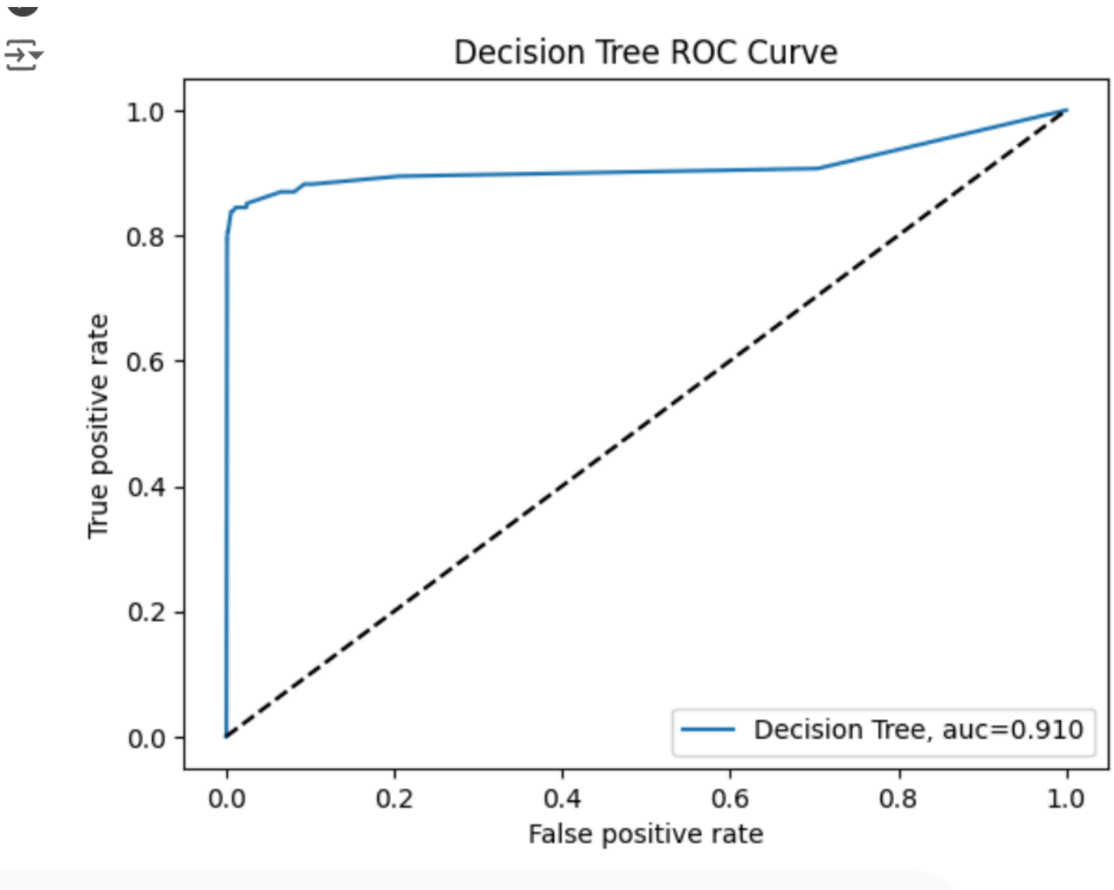


5. ROC and AUC

- **ROC Curve**: Plotted to show the trade-off between true positive rate (sensitivity) and false positive rate.

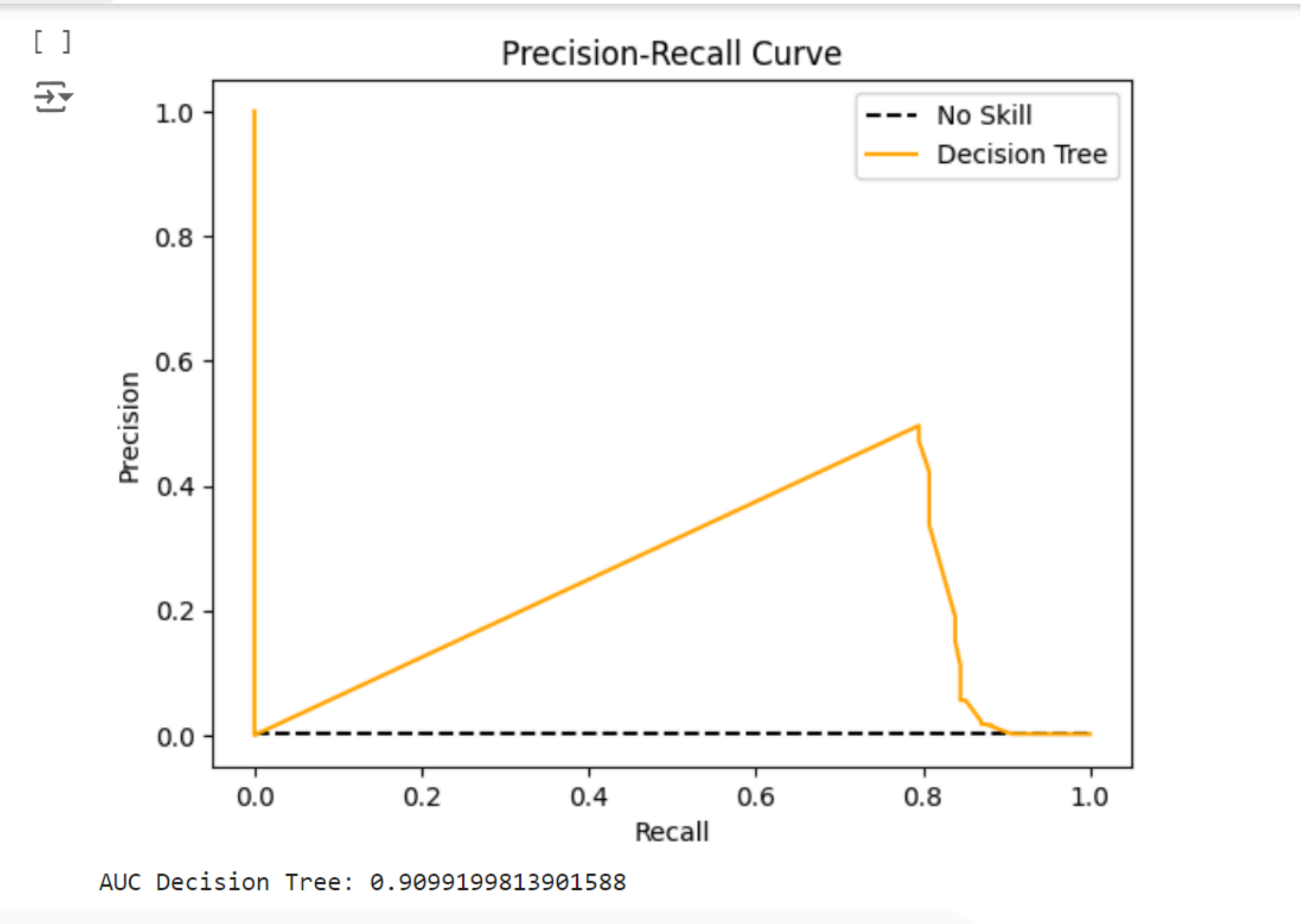
- **AUC (Area Under Curve)**: AUC provides a single metric indicating the overall ability of the model to distinguish between fraud and non-fraud cases. The closer the AUC to 1, the better the model.

The ROC curve for the Decision Tree model rises steeply towards the top-left, indicating a strong ability to differentiate between fraud and non-fraud cases. The AUC score of 0.910 suggests excellent model performance, as it’s close to the ideal score of 1. The curve is well above the diagonal baseline (random guessing line), confirming that the model reliably distinguishes fraudulent transactions from legitimate ones.



6. Precision-Recall Curve

- The precision-recall curve further evaluated the model’s performance by plotting precision against recall, especially useful for imbalanced datasets. The baseline "no skill" line represented the probability of guessing fraud cases randomly.
- The PR curve shows the decision tree classifier's precision-recall trade-off, with precision on the y-axis and recall on the x-axis. The decision tree achieves high precision for low recall but drops sharply as recall increases.
- The AUC score is 0.91, indicating strong performance, as a score of 1.0 would be perfect. The "No Skill" model, represented by the dashed line, serves as a baseline, showing constant precision regardless of recall for random guessing. The decision tree clearly outperforms the no-skill baseline.

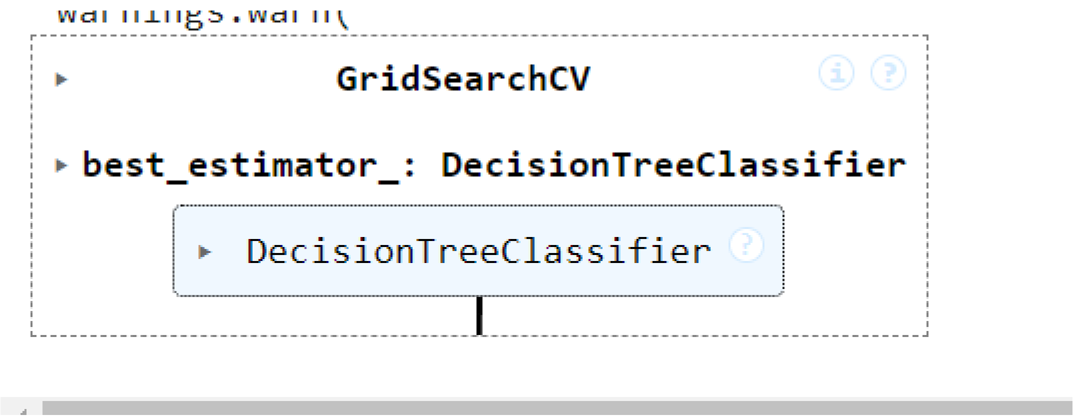


```
AUC Decision Tree: 0.9099199813901588
```

7. Hyperparameter Tuning with GridSearchCV

A **GridSearchCV** was used to fine-tune the Decision Tree's parameters, searching through various combinations of `criterion`, `splitter`, `max_depth`, and `max_features` to find the best performing configuration.

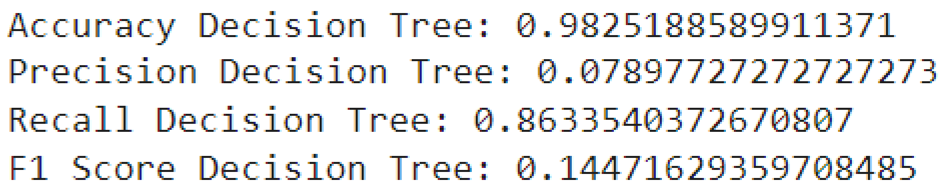
- **Best Parameters**: Found by evaluating the model's accuracy using 5-fold cross-validation.



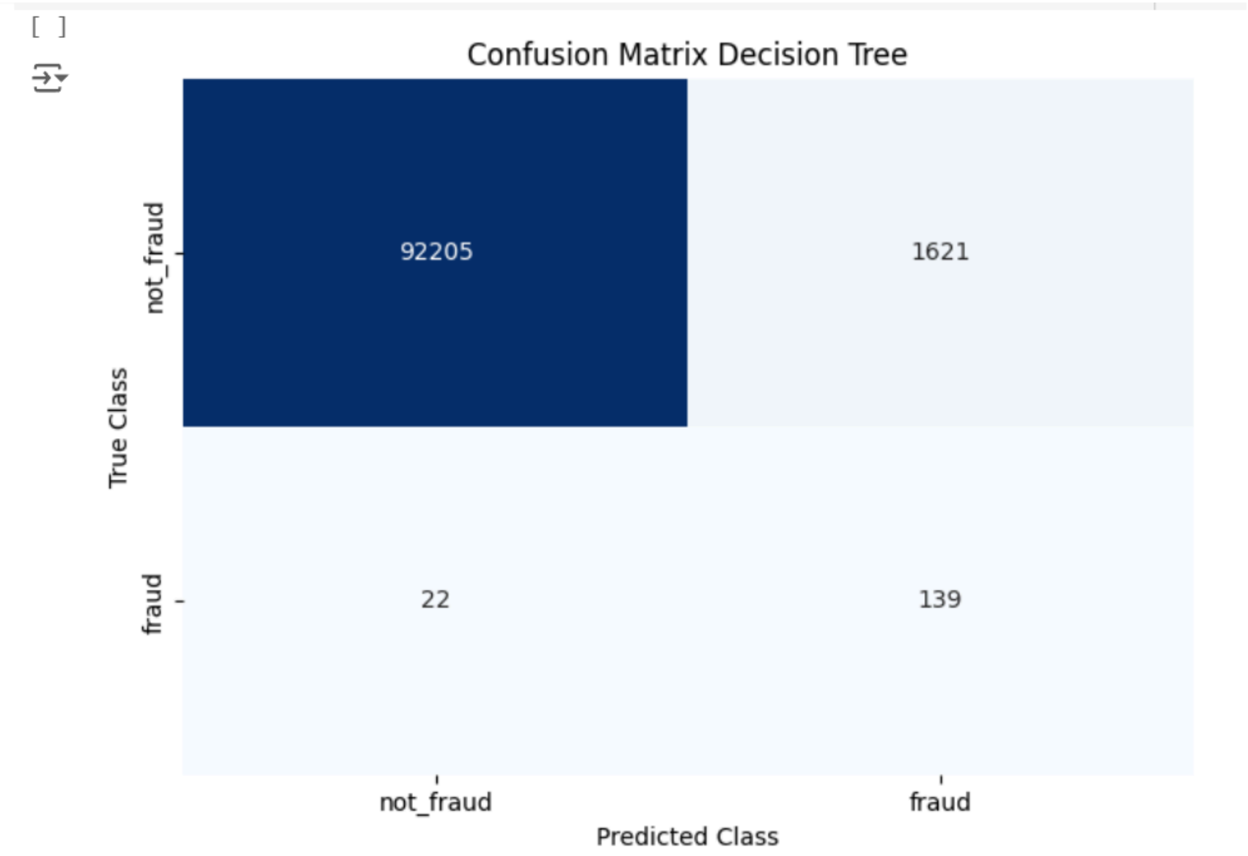
```
warnings.warn(
▸ GridSearchCV
▸ best_estimator_: DecisionTreeClassifier
    ▸ DecisionTreeClassifier
```

8. Final Results After Tuning

The model's performance was re-evaluated using the optimal hyperparameters. The final metrics were recalculated and the confusion matrix was visualized again to assess improvements.



```
Accuracy Decision Tree: 0.9825188589911371
Precision Decision Tree: 0.07897727272727273
Recall Decision Tree: 0.8633540372670807
F1 Score Decision Tree: 0.14471629359708485
```



Confusion Matrix Decision Tree