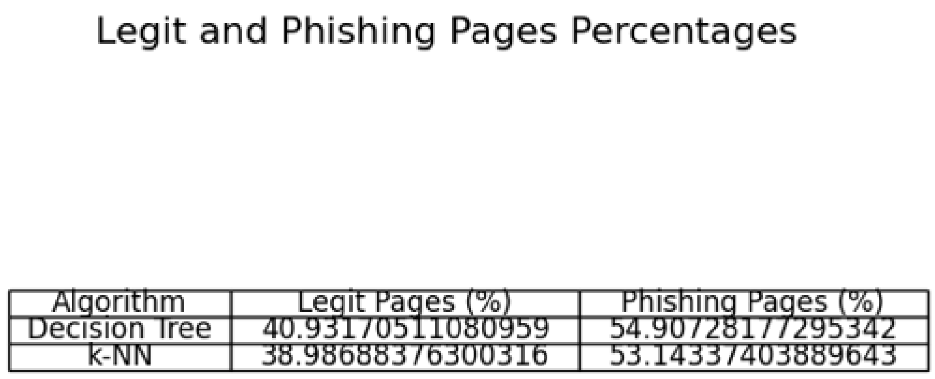
Project Report

Psarros Filippos

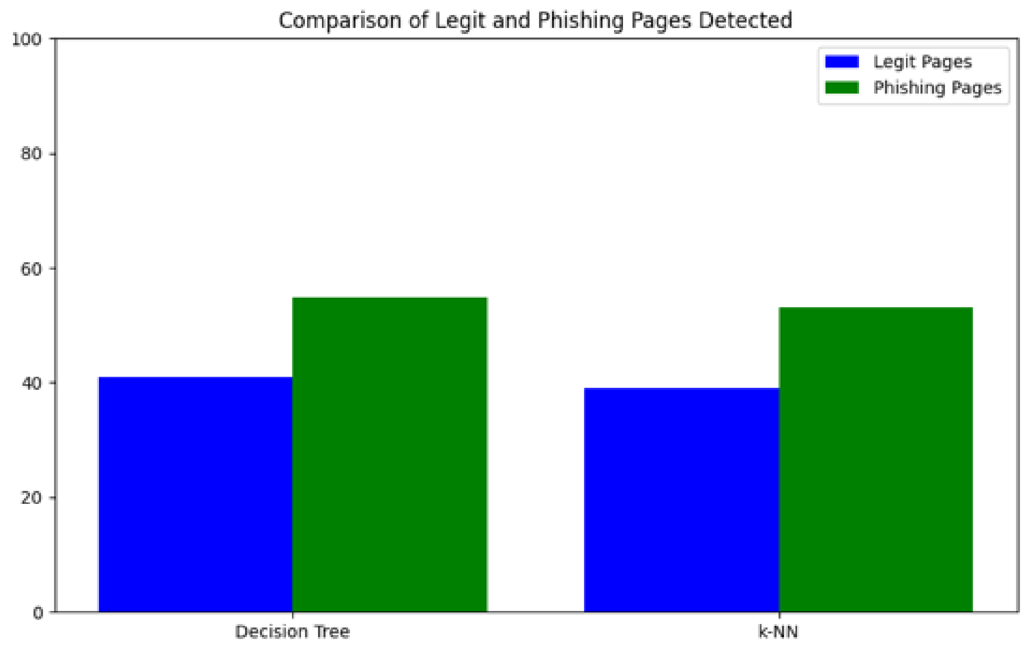
A full analysis of the project is provided in the Phishing_Website_Detection_using-Decision_Trees_and_k_NN_report.pdf file.

Below we can see the percentage of phishing and legit websites.



Legit and Phishing Pages Percentages

Algorithm	Legit Pages (%)	Phishing Pages (%)
Decision Tree	40.93170511080959	54.90728177295342
k-NN	38.98688376300316	53.14337403889643



By observing the results, we can see that the algorithms produce similar outcomes with only slight differences. The ratio between the Legit and Phishing categories does not show significant deviation, and the difference between the two algorithms is less than 5%. This indicates that the dataset is relatively balanced, as the observed differences are minimal.

Normalization

Normalization was applied to the data. Specifically, the StandardScaler class from the sklearn.preprocessing library was used, which normalizes the features so that they have a mean of 0 and a standard deviation of 1. Since the k-NN algorithm relies on distance measurements (e.g., Euclidean distances), normalization is especially important. Without normalization, features with larger numerical scales would dominate the distance calculations, negatively affecting the model's accuracy. Therefore, normalization improves feature comparison and the overall performance of the algorithm.
In contrast, normalization is not necessary for the Decision Tree algorithm, as it does not rely on distance calculations. However, normalization was applied for consistency and uniform preprocessing across the dataset.

Training Method

In this code, the Train-Test Split method was applied using the train_test_split function from the sklearn.model_selection library. The dataset was split into training and test sets using an 80%–20% ratio.

The reasons this method was used are:

1. It is simple and fast, making it suitable for building a basic evaluation system.
2. The test set remains independent from the training process, ensuring an objective assessment of model performance.
3. The problem involves classifying websites as "Legit" or "Phishing." Splitting the data allows the algorithms (Decision Tree and k-NN) to train on the training set and be evaluated on unseen test data, ensuring good generalization.

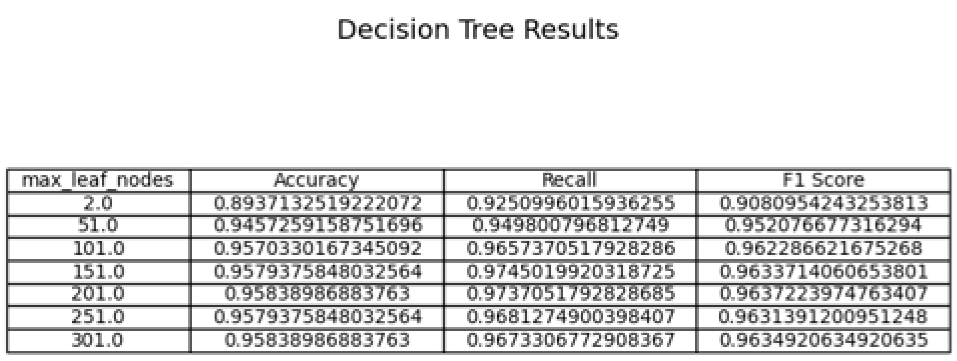
max_leaf_nodes Parameter for Decision Trees

Specific values for the max_leaf_nodes parameter were tested in the code as part of optimizing the Decision Tree algorithm's accuracy. The tested values were:

- 2, 51, 101, 151, 201, 251, 301

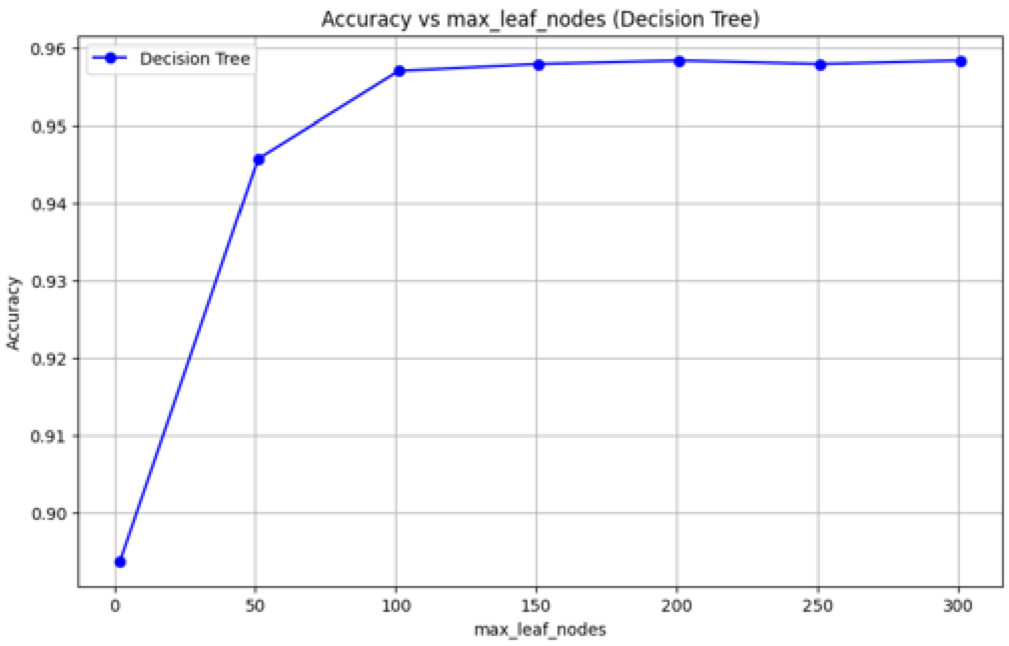
These values were chosen based on the logic of gradually increasing the number of leaves to assess how tree complexity affects accuracy. From the analysis of the results for these values, the optimal max_leaf_nodes value was selected — the one that maximized accuracy on the test set.

The results are shown below:



Decision Tree Results

max_leaf_nodes	Accuracy	Recall	F1 Score
2.0	0.8937132519222072	0.9250996015936255	0.9080954243253813
51.0	0.9457259158751696	0.949800796812749	0.952076677316294
101.0	0.9570330167345092	0.9657370517928286	0.962286621675268
151.0	0.9579375848032564	0.9745019920318725	0.9633714060653801
201.0	0.95838986883763	0.9737051792828685	0.9637223974763407
251.0	0.9579375848032564	0.9681274900398407	0.9631391200951248
301.0	0.95838986883763	0.9673306772908367	0.9634920634920635

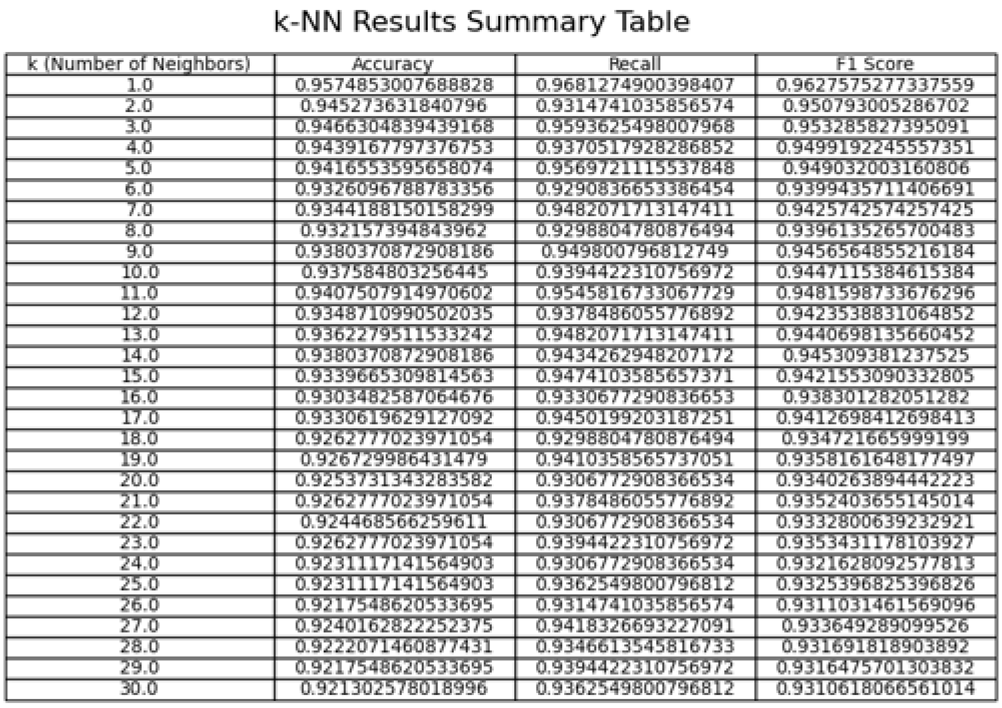


k Parameter for k-NN

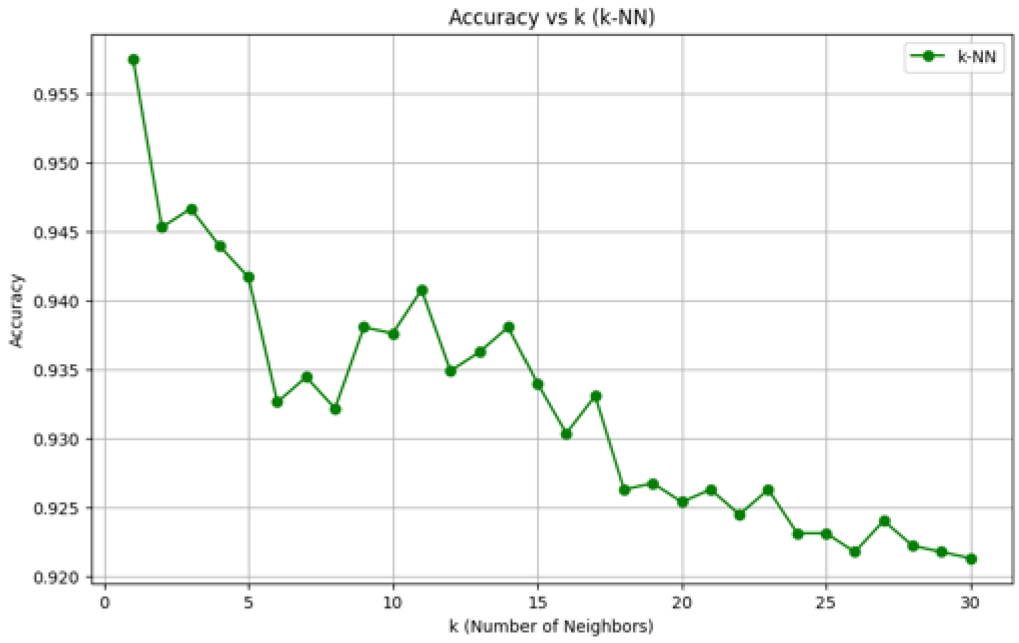
In the code, the k parameter for the k-Nearest Neighbors (k-NN) algorithm was optimized by testing values from 1 to 30. The appropriate value of k was selected by measuring the accuracy for each tested value using the test set.

The optimal k was chosen as the one that maximized the algorithm's accuracy on the test set. This process ensures that the model generalizes better to unseen data. Additionally, lower values of k can lead to overfitting, while higher values may lead to underfitting. Thus, this approach avoids both extremes.

The results for the k-NN algorithm are shown below:


k-NN Results Summary Table

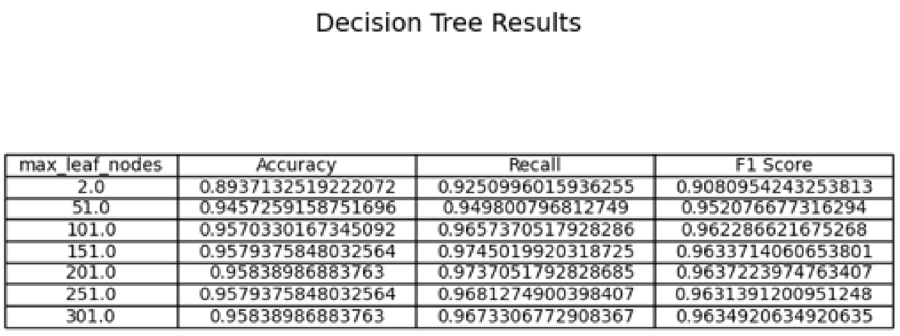
k (Number of Neighbors)	Accuracy	Recall	F1 Score
1.0	0.9574853007688828	0.9681274900398407	0.9627575277337559
2.0	0.945273631840796	0.9314741035856574	0.950793005286702
3.0	0.9466304839439168	0.9593625498007968	0.953285827395091
4.0	0.9439167797376753	0.9370517928286852	0.9499192245557351
5.0	0.9416553595658074	0.9569721115537848	0.949032003160806
6.0	0.9326096788783356	0.9290836653386454	0.9399435711406691
7.0	0.9344188150158299	0.9482071713147411	0.9425742574257425
8.0	0.932157394843962	0.9298804780876494	0.9396135265700483
9.0	0.9380370872908186	0.949800796812749	0.9456564855216184
10.0	0.937584803256445	0.9394422310756972	0.9447115384615384
11.0	0.9407507914970602	0.9545816733067729	0.9481598733676296
12.0	0.9348710990502035	0.9378486055776892	0.9423538831064852
13.0	0.9362279511533242	0.9482071713147411	0.9440698135660452
14.0	0.9380370872908186	0.9434262948207172	0.945309381237525
15.0	0.9339665309814563	0.9474103585657371	0.9421553090332805
16.0	0.9303482587064676	0.9330677290836653	0.938301282051282
17.0	0.9330619629127092	0.9450199203187251	0.9412698412698413
18.0	0.9262777023971054	0.9298804780876494	0.934721665999199
19.0	0.926729986431479	0.9410358565737051	0.9358161648177497
20.0	0.9253731343283582	0.9306772908366534	0.9340263894442223
21.0	0.9262777023971054	0.9378486055776892	0.9352403655145014
22.0	0.924468566259611	0.9306772908366534	0.9332800639232921
23.0	0.9262777023971054	0.9394422310756972	0.9353431178103927
24.0	0.9231117141564903	0.9306772908366534	0.9321628092577813
25.0	0.9231117141564903	0.9362549800796812	0.9325396825396826
26.0	0.9217548620533695	0.9314741035856574	0.9311031461569096
27.0	0.9240162822252375	0.9418326693227091	0.933649289099526
28.0	0.9222071460877431	0.9346613545816733	0.931691818903892
29.0	0.9217548620533695	0.9394422310756972	0.9316475701303832
30.0	0.921302578018996	0.9362549800796812	0.9310618066561014



We observe that the best value is k = 1.

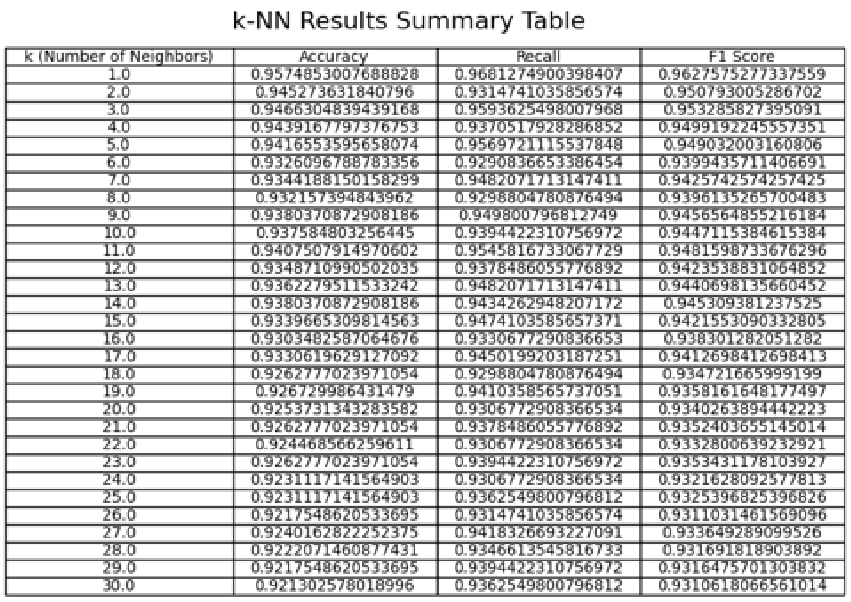
Accuracy and Performance

Below we present the performance metrics of the two algorithms (Accuracy, Recall, and F1 Score):



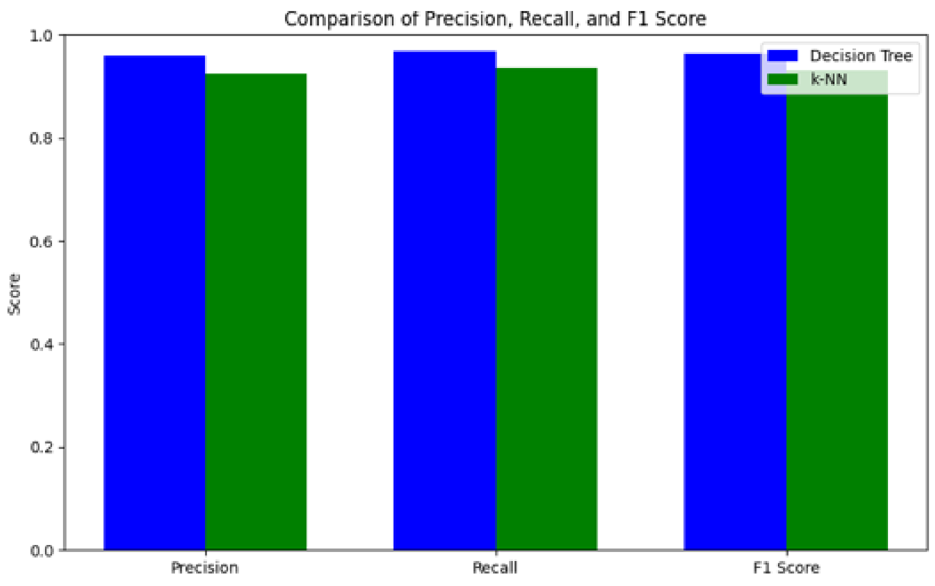
Decision Tree Results

max_leaf_nodes	Accuracy	Recall	F1 Score
2.0	0.8937132519222072	0.9250996015936255	0.9080954243253813
51.0	0.9457259158751696	0.949800796812749	0.952076677316294
101.0	0.9570330167345092	0.9657370517928286	0.962286621675268
151.0	0.9579375848032564	0.9745019920318725	0.9633714060653801
201.0	0.95838986883763	0.9737051792828685	0.9637223974763407
251.0	0.9579375848032564	0.9681274900398407	0.9631391200951248
301.0	0.95838986883763	0.9673306772908367	0.9634920634920635



k-NN Results Summary Table

k (Number of Neighbors)	Accuracy	Recall	F1 Score
1.0	0.9574853007688828	0.9681274900398407	0.9627575277337559
2.0	0.945273631840796	0.9314741035856574	0.950793005286702
3.0	0.9466304839439168	0.9593625498007968	0.953285827395091
4.0	0.9439167797376753	0.9370517928286852	0.9499192245557351
5.0	0.9416553595658074	0.9569721115537848	0.949032003160806
6.0	0.9326096788783356	0.9290836653386454	0.9399435711406691
7.0	0.9344188150158299	0.9482071713147411	0.9425742574257425
8.0	0.932157394843962	0.9298804780876494	0.9396135265700483
9.0	0.9380370872908186	0.949800796812749	0.9456564855216184
10.0	0.937584803256445	0.9394422310756972	0.9447115384615384
11.0	0.9407507914970602	0.9545816733067729	0.9481598733676296
12.0	0.9348710990502035	0.9378486055776892	0.9423538831064852
13.0	0.9362279511533242	0.9482071713147411	0.9440698135660452
14.0	0.9380370872908186	0.9434262948207172	0.945309381237525
15.0	0.9339665309814563	0.9474103585657371	0.9421553090332805
16.0	0.9303482587064676	0.9330677290836653	0.938301282051282
17.0	0.9330619629127092	0.9450199203187251	0.9412698412698413
18.0	0.9262777023971054	0.9298804780876494	0.934721665999199
19.0	0.926729986431479	0.9410358565737051	0.9358161648177497
20.0	0.9253731343283582	0.9306772908366534	0.9340263894442223
21.0	0.9262777023971054	0.9378486055776892	0.9352403655145014
22.0	0.924468566259611	0.9306772908366534	0.9332800639232921
23.0	0.9262777023971054	0.9394422310756972	0.9353431178103927
24.0	0.9231117141564903	0.9306772908366534	0.9321628092577813
25.0	0.9231117141564903	0.9362549800796812	0.9325396825396826
26.0	0.9217548620533695	0.9314741035856574	0.9311031461569096
27.0	0.9240162822252375	0.9418326693227091	0.933649289099526
28.0	0.9222071460877431	0.9346613545816733	0.931691818903892
29.0	0.9217548620533695	0.9394422310756972	0.9316475701303832
30.0	0.921302578018996	0.9362549800796812	0.9310618066561014



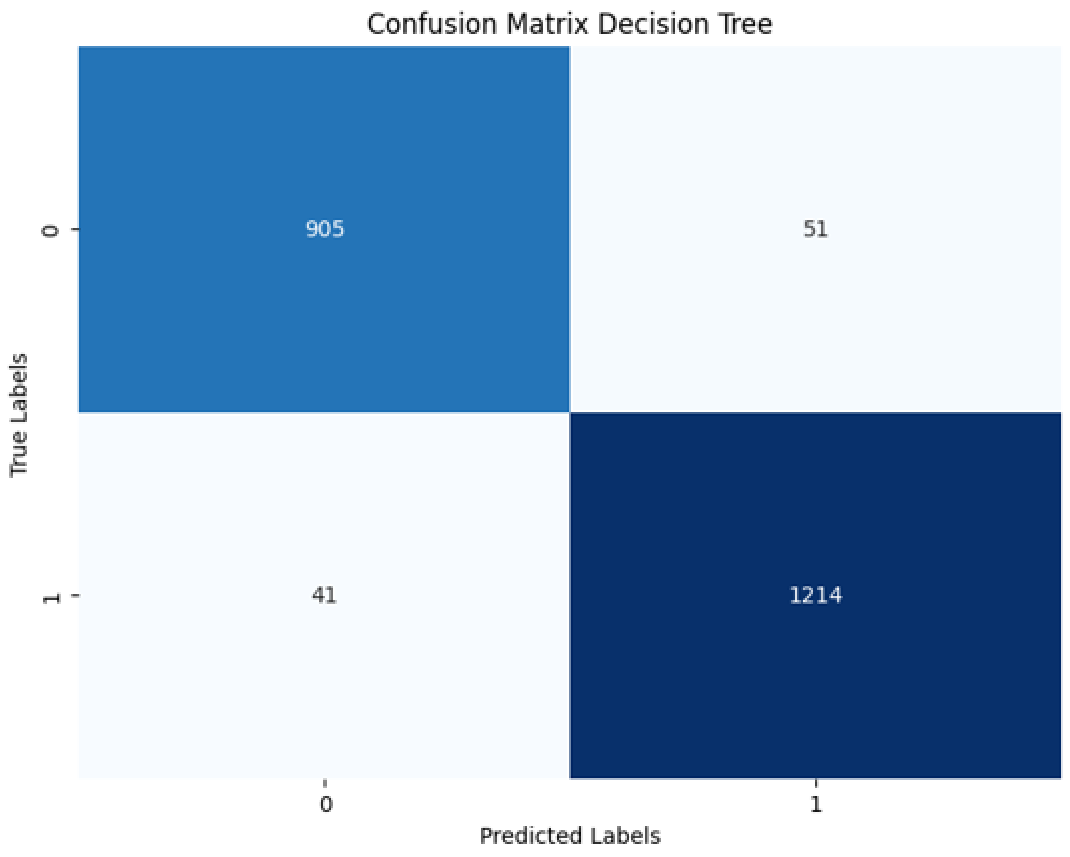
Based on the results presented in the figures, we can compare the Accuracy, Recall, and F1 Score for the two algorithms (Decision Tree and k-NN):

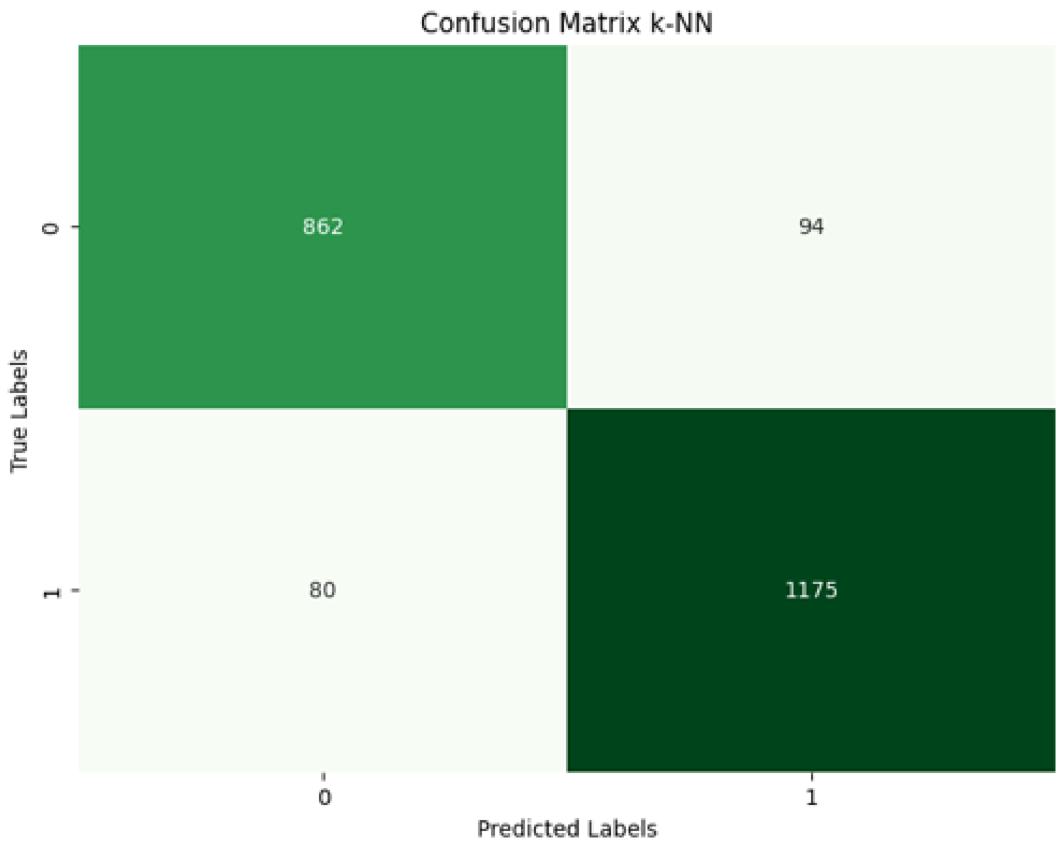
1. **Accuracy**:
   - The Decision Tree algorithm achieved slightly better accuracy (0.9583 for max_leaf_nodes=201) compared to k-NN, which had a maximum accuracy of 0.9574 for k=1.
   - The difference is minimal, indicating that both algorithms perform well in classifying the data.
2. **Recall**:
   - The Recall of the Decision Tree was higher (0.9737 for max_leaf_nodes=201) compared to k-NN (0.9681 for k=1).
   - This means that the Decision Tree is slightly more effective in identifying actual "Phishing" pages.
3. **F1 Score**:
   - The F1 Score, which combines both precision and recall, was also slightly higher for the Decision Tree (0.9637 for max_leaf_nodes=201) compared to k-NN (0.9627 for k=1).
   - This small difference suggests that both algorithms perform exceptionally well.

In summary, both algorithms achieve very high success rates, with minor differences in the above metrics. The Decision Tree appears to be the slightly better choice due to its more consistent performance across most metrics.

Confusion Matrices

Below we present the confusion matrices for each algorithm.





The confusion matrices for the Decision Tree and k-NN algorithms show the classification results between legitimate and phishing websites using the labels:

- **00**: Legitimate websites correctly classified
- **11**: Phishing websites correctly classified
- **01**: Legitimate websites misclassified as phishing
- **10**: Phishing websites misclassified as legitimate

1. **Decision Tree:**

- **00**: 905, correct classification of legitimate websites
- **11**: 1214, correct classification of phishing websites
- **01**: 51, legitimate websites misclassified as phishing
- **10**: 41, phishing websites misclassified as legitimate

The Decision Tree model achieves higher values in correct classifications (00 and 11) and lower values in incorrect ones (01 and 10), making it more accurate overall.

2. **k-NN:**

- **00**: 862, correct classification of legitimate websites
- **11**: 1175, correct classification of phishing websites
- **01**: 94, legitimate websites misclassified as phishing
- **10**: 80, phishing websites misclassified as legitimate

The k-NN model yields lower values in correct classifications and higher values in incorrect ones compared to the Decision Tree.

Conclusion:
The Decision Tree outperforms k-NN in classification accuracy, demonstrating higher precision in distinguishing between the two classes. k-NN produces more misclassifications across all categories, negatively impacting its performance.

Evaluation on the Performances of the Algorithms (e.g., Accuracy, Recall, F1 Score)

Based on the performance evaluation of the Decision Tree and k-NN algorithms using accuracy, recall, and F1 score, we observe the following:

- **Decision Tree:**
    - **Accuracy:** Higher than k-NN, indicating better overall performance
    - **Recall:** Excellent, showing that the model correctly identifies the majority of phishing pages
    - **F1 Score:** Slightly better than k-NN, indicating a balanced performance between precision and recall
- **k-NN:**
    - **Accuracy:** Slightly lower than Decision Tree, with more classification errors
    - **Recall:** Still good, but lower than the Decision Tree
    - **F1 Score:** Close to Decision Tree, but slightly inferior

Comparison and Suitability of Each Algorithm

- The Decision Tree is more suitable when a balance between accuracy and recall is needed. It is ideal in cases where misclassifications (e.g., labeling a phishing site as legitimate) must be minimized.
- The k-NN algorithm is appropriate for problems relying on distance-based or pattern recognition approaches, but its performance heavily depends on the choice of k and the normalization of data. In high-dimensional or noisy datasets, k-NN may underperform compared to Decision Trees.

In summary, the Decision Tree stands out in this case due to its superior overall performance across accuracy, recall, and F1 score. While k-NN remains reliable, it does not reach the performance levels of the Decision Tree for this specific website classification problem.