



Fake Job Post Prediction Using Different Data mining Techniques

Faraz Ahmed (200547626)
Fareed Shahid (200551973)



Objective of the Paper:

The main goal of this research is to address the growing concern of job scam detection. The aim is to analyze the impacts of fraudulent job posts and contribute to the research field by tackling the challenges associated with detecting such posts. The primary focus is on predicting fake job posts using both traditional machine learning algorithms and a deep learning model.



Data Mining Techniques:

The paper employs various data mining techniques to achieve its objective. The techniques include traditional machine learning algorithms such as Decision_tree, Random_forest, SVM, Logistic Regression and a Deep learning model, specifically a Deep Neural Network (DNN).



Datasets:

The study utilizes the EMSCAD dataset for experimentation, which contains real-life examples of fake job posts. The dataset is processed using Google Colab. For conventional machine learning algorithms, hold-out cross-validation is employed, with 80% of the data used for training and 20% for testing. Specific details are provided for the KNN model, where a range of K values is explored. For the DNN model, a 10-fold cross-validation approach is used, with 60% for training, 20% for validation, and 20% for testing.




LIBRARIES

- PANDAS
- NUMPY
- SEABORN
- SKLEARN
- WORDCLOUD
- NLTK




STEP 1: Data cleaning and handling the missing values.

STEP 2: Split the dataset into training, validation, and testing sets.

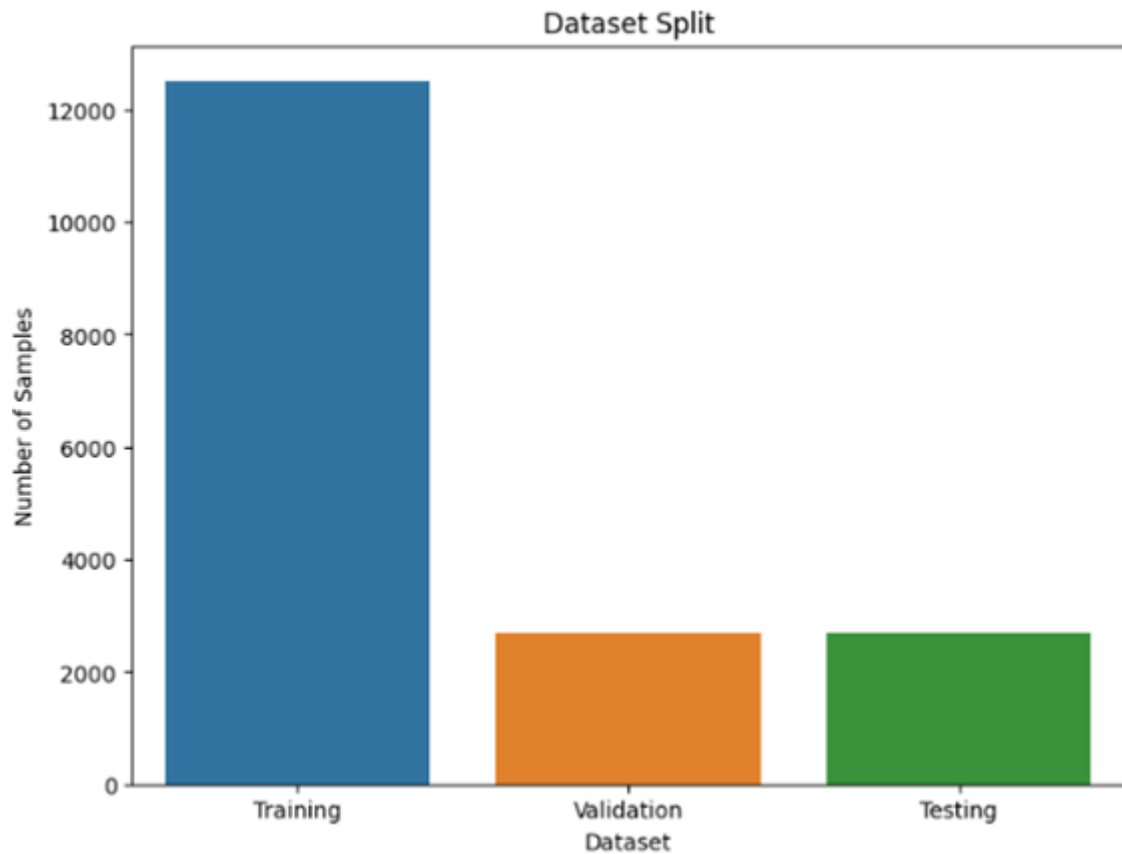


```
# Print the shape of the training, validation, and testing sets
print("Training set shape:", X_train.shape, y_train.shape)
print("Validation set shape:", X_val.shape, y_val.shape)
print("Testing set shape:", X_test.shape, y_test.shape)
```



```
Training set shape: (12515,) (12515,)
Validation set shape: (2682,) (2682,)
Testing set shape: (2682,) (2682,)
```

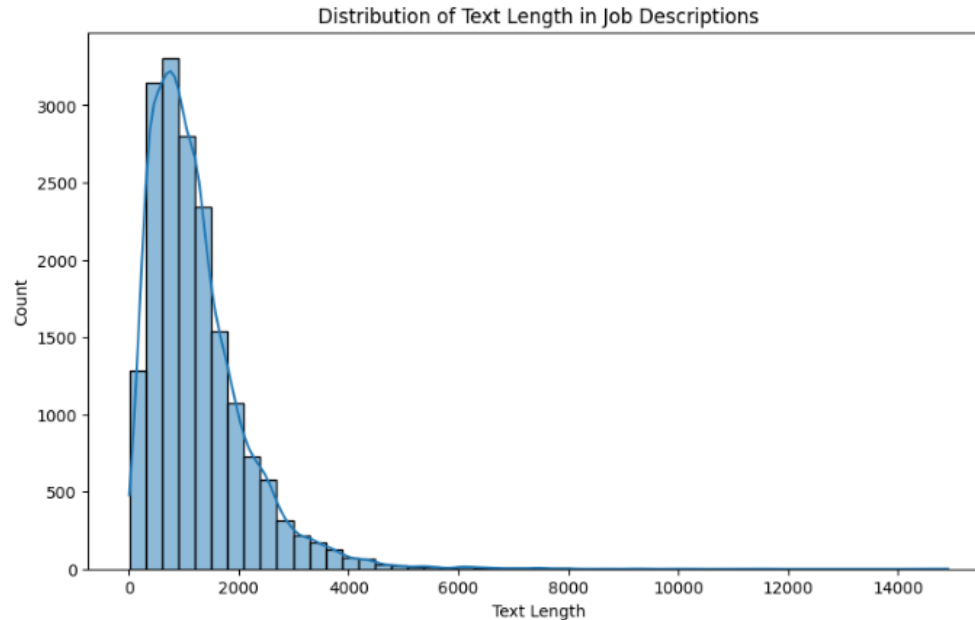
Data Split:




STEP 3: Generate Word Clouds of Legit and Fake Job Postings.



Distribution of text lengths in job descriptions





STEP 4: Feature extraction using the Term Frequency-Inverse Document Frequency (TF-IDF) vectorizer.

```
# Step 6: Model Training
# Instantiate the classification models
decision_tree = DecisionTreeClassifier()
random_forest = RandomForestClassifier()
svm = SVC(probability=True) # Set probability=True for later ROC curve
logistic_regression = LogisticRegression()

[22] # Train the models using the vectorized training data
decision_tree.fit(X_train_vectorized, y_train)
random_forest.fit(X_train_vectorized, y_train)
svm.fit(X_train_vectorized, y_train)
logistic_regression.fit(X_train_vectorized, y_train)
```

STEP 5: Train the Model.



STEP 6: Model Evaluation

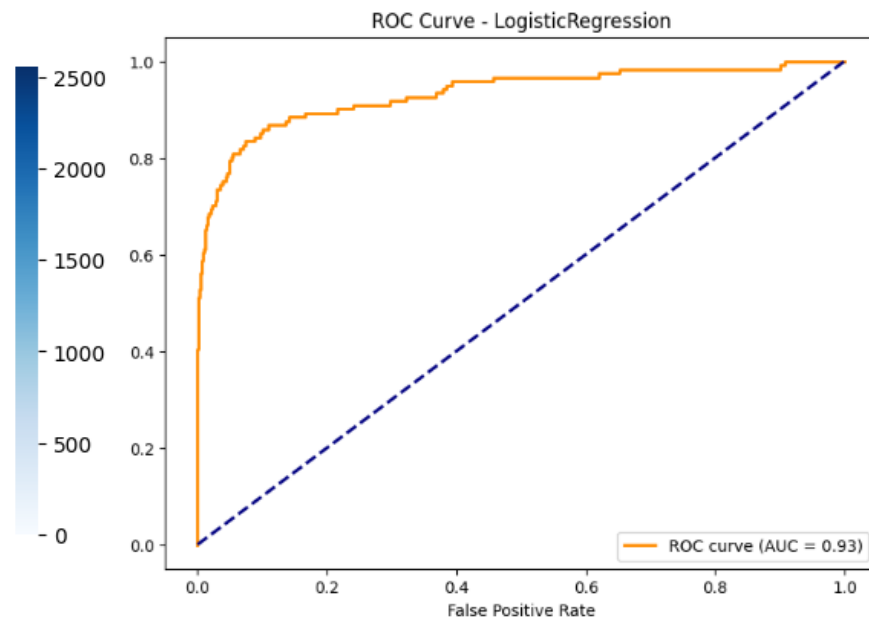
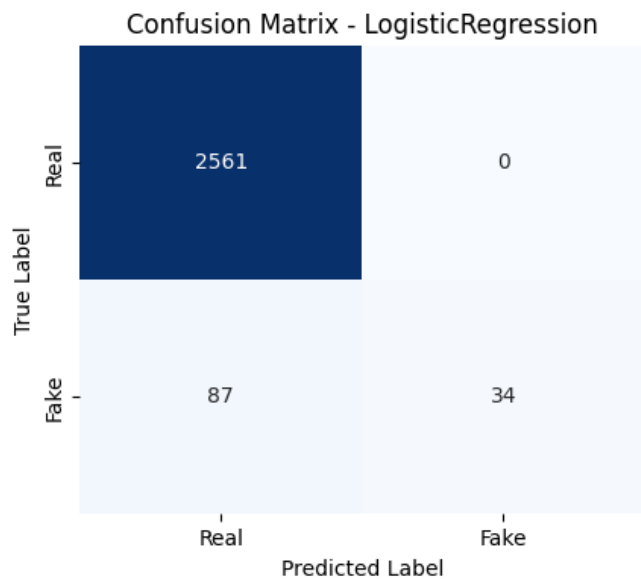
```
Model: DecisionTreeClassifier
Accuracy: 0.9713, Precision: 0.6930, Recall: 0.6529, F1 Score: 0.6723
Confusion Matrix:
[[2526   35]
 [  42   79]]

Model: RandomForestClassifier
Accuracy: 0.9806, Precision: 1.0000, Recall: 0.5702, F1 Score: 0.7263
Confusion Matrix:
[[2561    0]
 [  52   69]]

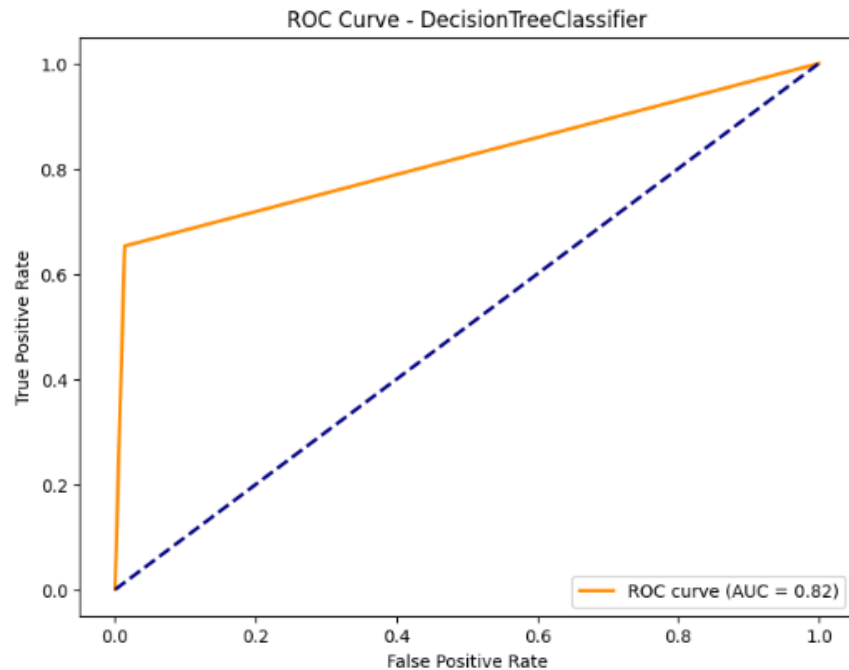
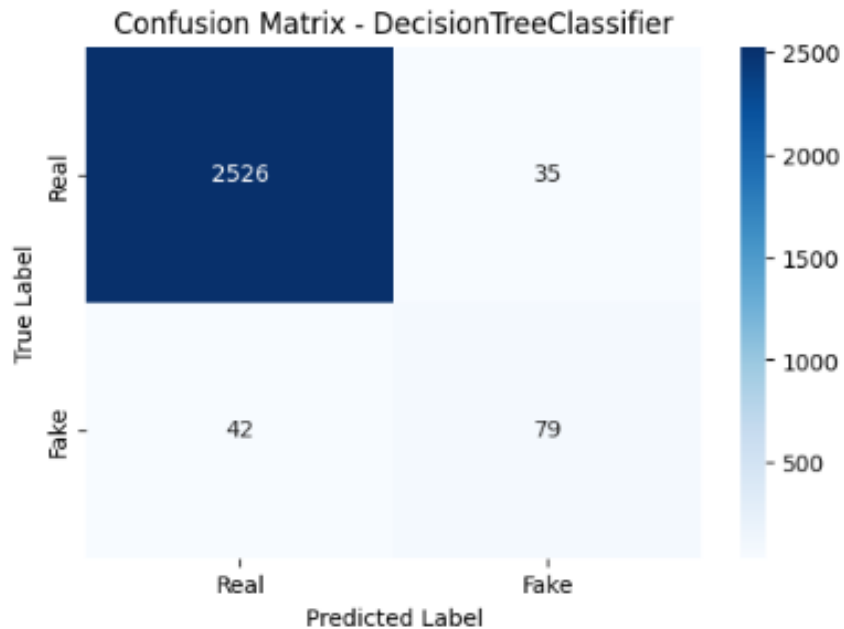
Model: SVC
Accuracy: 0.9795, Precision: 1.0000, Recall: 0.5455, F1 Score: 0.7059
Confusion Matrix:
[[2561    0]
 [  55   66]]

Model: LogisticRegression
Accuracy: 0.9676, Precision: 1.0000, Recall: 0.2810, F1 Score: 0.4387
Confusion Matrix:
[[2561    0]
 [  87   34]]
```

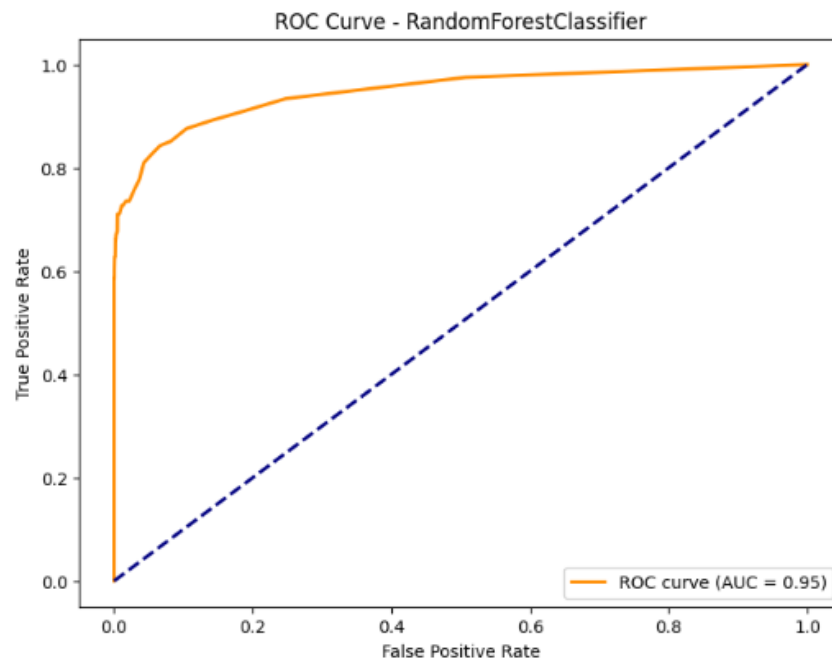
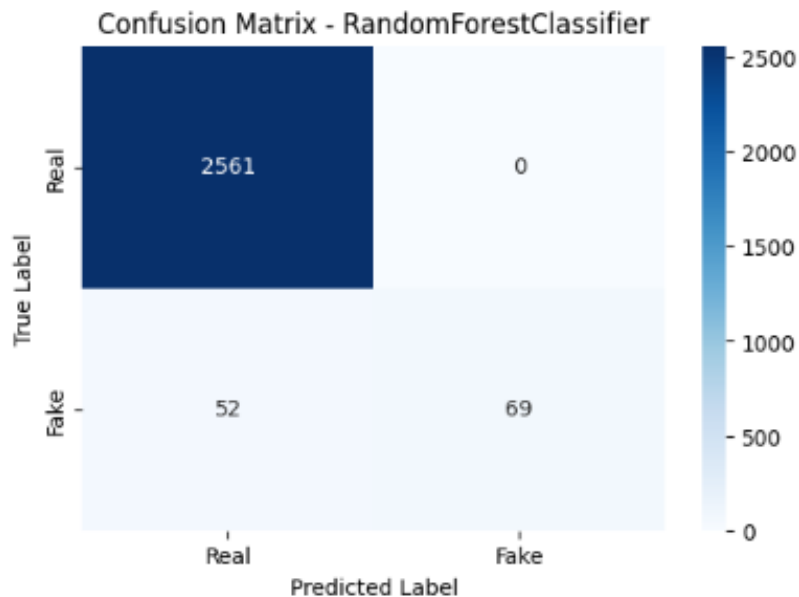
ROC-CURVE: Logistic Regression

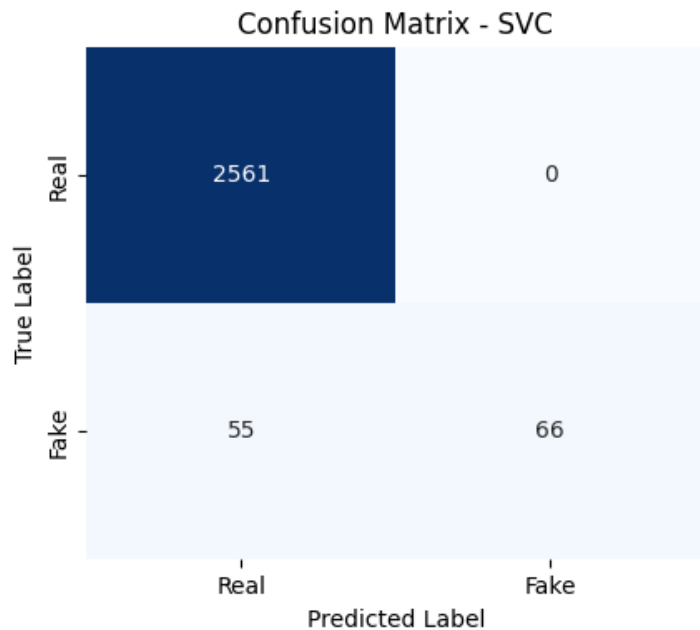


Decision Tree Classifier

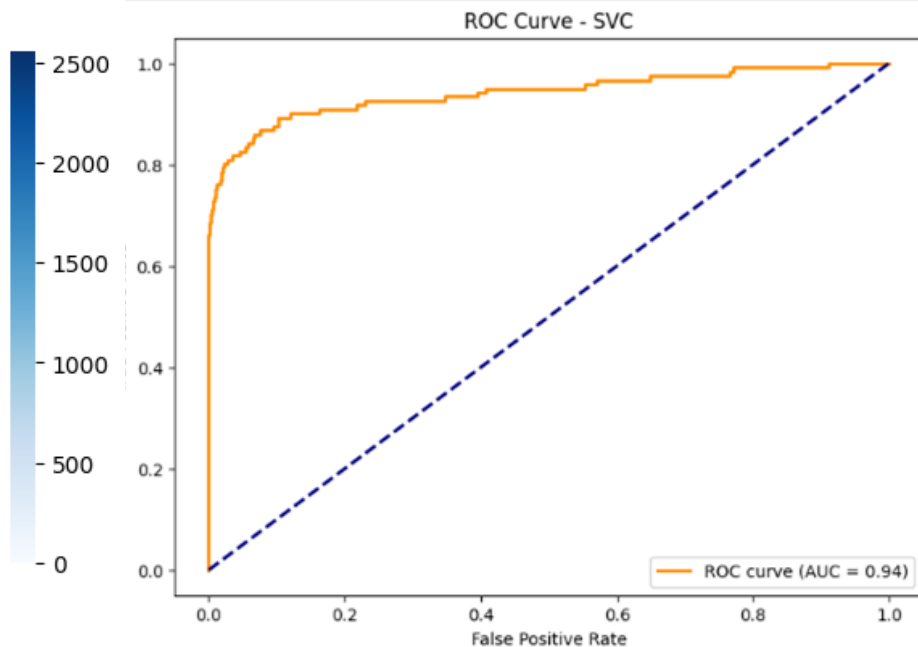


Random Forest Classifier

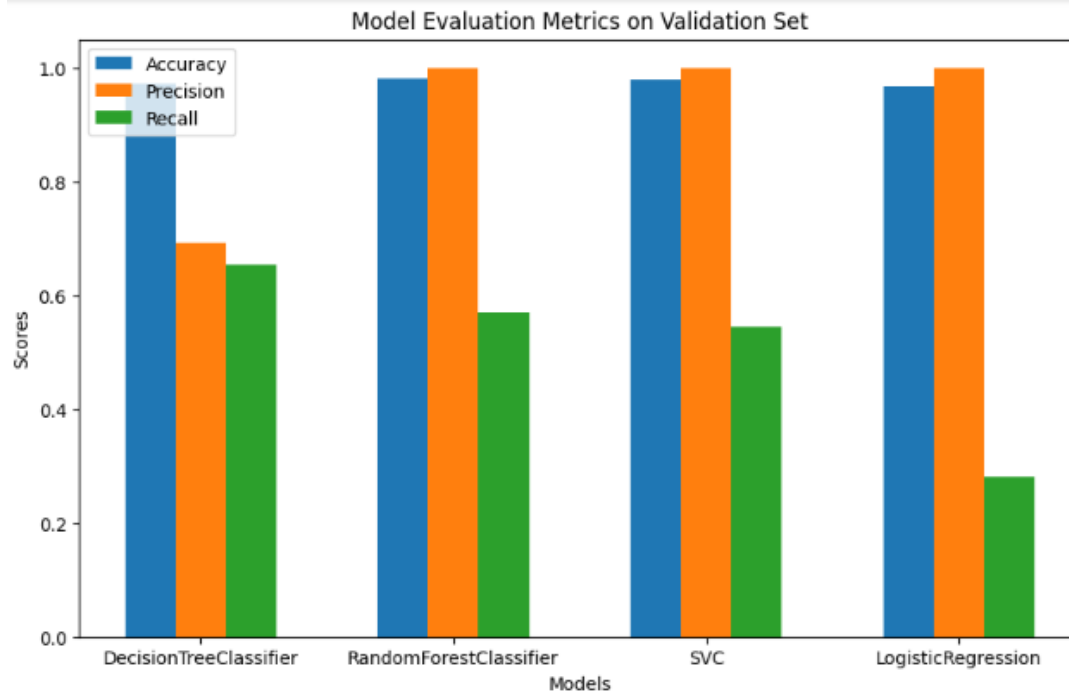




SVC



Model Evaluation





Model Evaluation on test set

```
Final Testing Accuracy (DecisionTreeClassifier): 0.9724  
Final Testing Accuracy (RandomForestClassifier): 0.9780  
Final Testing Accuracy (SVC): 0.9746  
Final Testing Accuracy (LogisticRegression): 0.9638
```

Generates word clouds for "Legit Jobs" and "Fake Jobs" based on the text data in the training set.

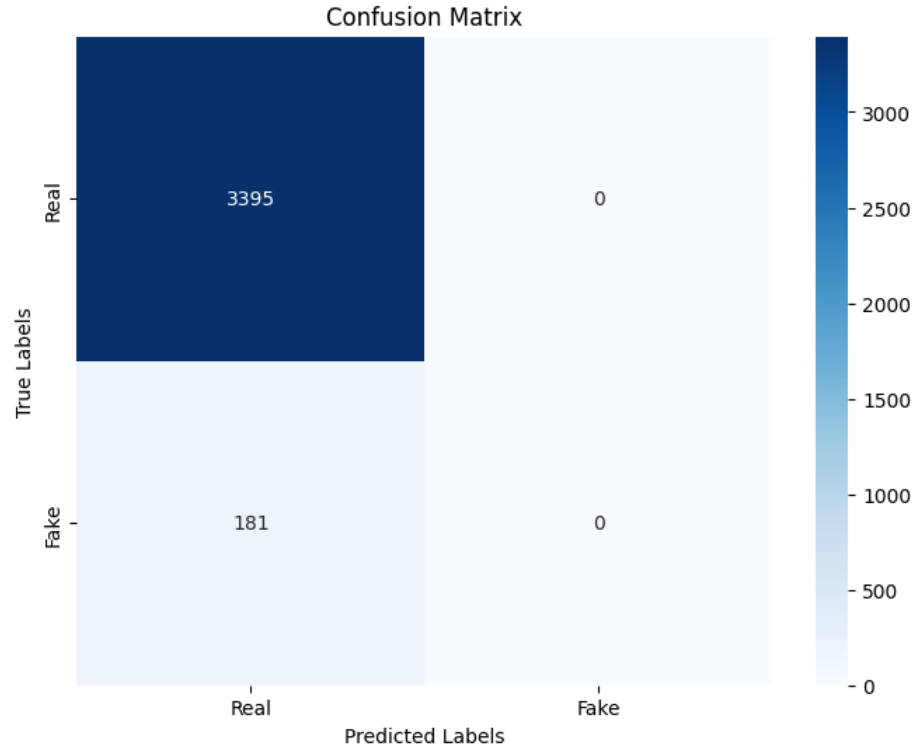
Word Cloud - Legit Jobs



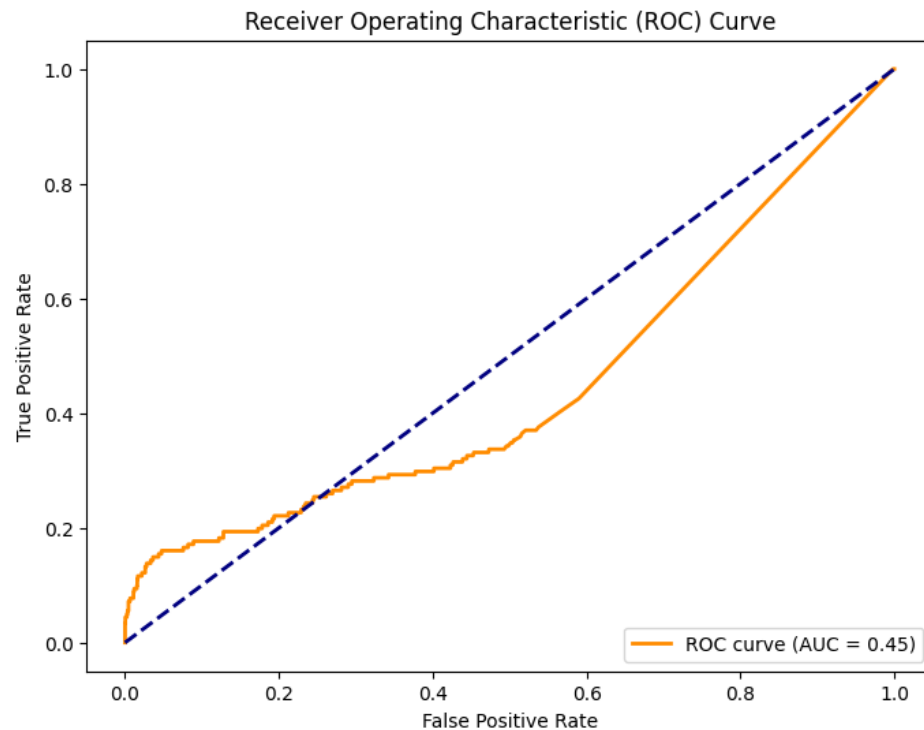
Word Cloud - Fake Jobs



Deep Neural Network (DNN)



ROC Curve of DNN:





Evaluation Metrics:

The paper evaluates the performance of the models using various metrics. For traditional machine learning algorithms, metrics such as classification accuracy, precision, recall, and F1 score are considered. For the DNN model, accuracy, precision, and recall are analyzed for each fold. The paper emphasizes the importance of assessing the model's performance on a class-imbalanced dataset and considers precision and recall in addition to accuracy.



Results and Findings:

The research reports the highest classification accuracy for the Random Forest Classifier among traditional machine learning algorithms. The DNN model achieves a 99% accuracy for specific folds, with an average classification accuracy of 97.7% across 10 folds. The paper highlights the importance of considering precision and recall, especially in a class-unbalanced dataset.



Challenges and Solutions:

The implementation faces challenges related to the class imbalance in the dataset. The use of precision and recall is emphasized to address this issue. Additionally, a dropout layer is introduced in the DNN model to reduce overfitting and improve generalization.