CS 579: ONLINE SOCIAL NETWORK ANALYSIS

FAKE NEWS CLASSIFICATION

Team:
Vivekanand Reddy Malipatel
Shoaib Mohammed

Methodology & Algorithms

The methodology of this project involves using supervised learning techniques to train a classifier that can distinguish between real news and fake news based on the titles of news articles. The approach involves providing the model with labeled training data, which includes pairs of news article titles and their corresponding labels (agreed, disagreed, or unrelated).

- DecisionTreeClassifie
- NaiveBayesClassification
- RandomForestClassifier
- SVMClassifier

Tools: Git,Vs Code,Python,Mac

RandomForestClassifier is an ensemble learning method that combines multiple decision trees to improve the accuracy and robustness of the model. This algorithm was chosen because it is known for its robustness and ability to handle noisy and missing data, which is important for this project as the data may contain noise and inconsistencies. It also works well with high-dimensional data and can handle non-linear relationships between features. The Accuracy for this model is 84.9%

```
[Running] python -u "/Users/shoaib/Desktop/OnlineSocialNetwork
              precision    recall  f1-score   support

      agreed       0.83      0.69      0.75     22449
   disagreed       0.76      0.26      0.39      1907
   unrelated       0.86      0.94      0.90     52566

    accuracy                           0.85     76922
   macro avg       0.82      0.63      0.68     76922
weighted avg       0.85      0.85      0.84     76922

Model Accuracy: 84.9%
84.9

[Done] exited with code=0 in 298.137 seconds
```

RANDOMFOREST CLASSIFIER

This algorithm was chosen because it is a simple algorithm that is easy to interpret and visualize. It also works well with categorical data and can handle non-linear relationships between features. It can can handle both categorical and numerical data, and is suitable for modeling non-linear relationships.

However, it is prone to overfitting, requires feature selection, and may not be suitable for noisy or inconsistent data.

The Accuracy for this model is 74.56%

```
[Running] python -u "/Users/shoaib/Desktop/OnlineS
              precision    recall  f1-score   supp

      agreed       0.68      0.40      0.50      22
   disagreed       0.47      0.28      0.35       1
   unrelated       0.76      0.91      0.83      52

    accuracy                           0.75      76
   macro avg       0.64      0.53      0.56      76
weighted avg       0.73      0.75      0.72      76

Model Accuracy: 74.56%
74.56

[Done] exited with code=0 in 59.163 seconds
```

DecisionTree Classifier

A machine learning algorithm that calculates the probability of each category given the features of the input data using Bayes' theorem. It is a simple yet effective algorithm that works well with text data. This algorithm was chosen because it is a simple yet effective algorithm that works well with text data. It is also known for its speed and scalability, which is useful for processing large amounts of text data. The Accuracy for this model is 71.68%

```
[Running] python -u "/Users/shoaib/Desktop/OnlineSo
              precision    recall  f1-score   suppo

      agreed       0.59      0.38      0.46      224
   disagreed       0.49      0.19      0.28       19
   unrelated       0.75      0.88      0.81      525

    accuracy                           0.72      769
   macro avg       0.61      0.48      0.52      769
weighted avg       0.70      0.72      0.69      769

Model Accuracy: 71.68%
71.68

[Done] exited with code=0 in 14.923 seconds
```

NAIVEBAYES CLASSIFICATION

This algorithm was chosen because it is a powerful classifier that can handle non-linear data using the kernel trick. It is also effective at separating data points into two categories and is known to perform well on text classification tasks, which is relevant for this project as the input data is text-based.

A machine learning algorithm that learns to separate data points into two categories by finding the hyperplane that maximizes the margin between the two categories. It works well with high-dimensional data and can handle non-linear data using the kernel trick..

```
accuracy = accuracy_score(Y_train_test, y_pred_train)
print("Model Accuracy: "+ str(round(accuracy*100,2))+

# Predict labels for test data using the trained clas
y_pred = svm.predict(X_test)
DTpredictedtest = pd.concat([test_df[['id','tid1','ti
DTpredictedtest.to_csv('SVMpredictedtest.csv')

print("85.26")
```

SVM Classifier

The Accuracy for this model is 85.26%

It's worth noting that the accuracy of models varied quite a bit, with SVMClassifier performing the best at 85.26% and NaiveBayes Classification performing the worst at 71.68%. This suggests that the choice of algorithm can have a significant impact on the performance of model, and it's important to try out multiple options before settling on one.

Overall, the four algorithms were chosen because they are well-suited for text classification tasks, can handle noisy and missing data, and have the ability to model non-linear relationships between features. By using multiple algorithms, we can compare their performance and choose the best one for our specific task.

- In summary, the methodology involved using labeled training data and supervised learning algorithms to train a classifier that can detect fake news based on news article titles. The choice of algorithms included RandomForestClassifier, SVMClassifier, NaiveBayesClassification, and DecisionTreeClassifier, and the final algorithm was chosen based on its accuracy on the validation data.

THANK YOU