Programming Assignment 1: Real vs. Fake Facts About Cities

Hamza Rashid
COMP 550, Fall 2024

Due: September 27th, 2024

I. Problem Setup

The aim of this assignment is to test the effectiveness of linear classifiers in distinguishing real from fake facts about cities, and determine if the choice of classifier is non-trivial. We evaluate the performance of a Linear Support Vector Machine, Multinomial Naive Bayes Classifier, Logistic Regression model, and Linear Regression model, and build a training pipeline to evaluate the performance of these classifiers under different preprocessing techniques.

II. Dataset Generation and Procedure

Using ChatGPT, we generated real and fake facts about Saint Petersburg, Russia. Three prompts were used:

- ”I would like to create a dataset containing facts about cities. I am particularly interested in Saint Petersburg, Russia. List 100 verified facts about Saint Petersburg, Russia.”
- ”now the reason i am asking for this data is to conduct a simple NLP project. As such, for pedagogical purposes, can you generate 100 fake facts about Saint Petersburg, Russia.”
- (to match the uniform distribution of data provided by another student) ”Give me 150 more real and fake facts”

We created two files: `facts.txt` (real facts) and `fakes.txt` (fake facts), each containing 2,750 samples, uniformly distributed across 11 cities. The dataset was divided into training, validation, and testing sets with a 70/10/20 split. We utilized sklearn’s relative term frequency (with inverse document frequency) vectorizer (`tfidf`), enabled to detect unigrams, bigrams, and trigrams (city names, and names of historical events tend to be split into 1-3 words). Three major preprocessing techniques explored and tested separately on the classifiers were: Porter-Stemmer, Lemmatization (NLTK, WordNet database), and supplementing the WordNet Lemmatizer with Part of Speech tagging to improve its accuracy. Minor preprocessing decisions applied to each of the aforementioned techniques included: converting text to lowercase, and removing all but alphanumeric and whitespace characters.

III. Parameter Settings and Models

We experimented with four linear classifiers: Linear Support Vector Machine, Multinomial Naive Bayes Classifier, Logistic Regression model, and Linear Regression model. Due to the simplicity of our dataset, we tested hyperparameters that address overfitting and how the models behave with unseen data. We tested the inverse-regularization parameter for the SVM and Logistic model with values {0.25, 0.5, 1.0}. As for the Naive Bayes classifier, we compared Laplace smoothing with varying degrees of Lidstone smoothing, testing values {0.25, 0.5, 1.0}. No hyperparameter testing was performed for the linear regression classifier. We also explored several train, validation, test split ratios.

IV. Results and Conclusions

The preprocessing methods, Porter-Stemmer, Lemmatization, and POS-supplemented Lemmatization yielded a similar number of features: 40,951, 41,986, 41,227, respectively. And, while the Logistic model performed best with C:=1.0 independently of the preprocessing method, we obtained mixed results with the SVM; C:=0.5 was favorable under Porter-Stemmer and Lemmatization, while the POS-supplemented Lemmatizer caused little bit more overfitting, requiring C:=0.25 for optimal performance. The overfitting might be a result of the POS-supplemented Lemmatizer generating the most features out of the three methods, causing an adverse effect in the SVM. As for the smoothing parameter, we did not find any noticeable differences across the tested values, which was expected given the simplicity of the dataset, even split of the classes, and the independence assumption making the Naive Bayes classifier less prone to overfitting. The relatively poor R^2 score of $\sim$ 80% from the linear regression classifier may be attributed to a feature vector's tendency to join a cluster of other data points in its class, due to the simplicity of our dataset. On the other hand, the high accuracy scores, in the neighborhood of 98%, across the SVM, Naive Bayes Classifier, and Logistic Regression model may be attributed to such clustering of classes, and similar number of features. Thus, the choice of linear classifier is not trivial, as its performance depends on the distribution of the feature vectors and classes (in a discrete problem).

V. Limitations and Generalization

The primary limitation of this study is the simplicity of the dataset. The fake documents use a particular set of words frequently (e.g., 'secret', 'underground', 'rumored') that do not appear as much in the counterpart class, and tend to be shorter in length than fact documents. As a result, the classes are easily distinguishable with linear models, as they tend to form separable clusters of feature vectors in a low dimensional space. However, fake facts can utilize the prominent patterns of a real fact to make a claim that can only be proved false with prior knowledge on the given subject – often the case when working with real-world data. Therefore, while it was reasonable to assume that ChatGPT would provide real facts, it was not reasonable to assume that the generated fake facts would be representative of false information that occurs in a real-world setting. Therefore, we cannot generalize the results of this study to the real world; the overall problem of distinguishing real from fake facts is not necessarily simple enough to be modelled by a linear classifier.