Programming Assignment 1: Real vs. Fake Facts About Cities

Hamza Rashid
COMP 550, Fall 2024

Due: September 27th, 2024

1 Problem Setup

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

2 Dataset Generation and Procedure

Using an online AI tool, 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 vectorizer (`tfidf`), with inverse document frequency (penalizing words that appear in many documents), and enabled to detect unigrams, bigrams, and trigrams (city names, and names of historical events tend to be split into 1-3 words). We created a data processing pipeline that culminated in the use of three major preprocessing techniques tested separately on the classifiers: 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.

3 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 the parameters that address overfitting. 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 experimented with various train, validation, test split ratios.

4 Results and Conclusions

While the Logistic model performed best with C:=1.0 independently of the preprocessing method, we obtained mixed results with the SVM; C:=1.0 was favorable under Porter-Stemmer and Lemmatization, while the POS-supplemented Lemmatizer caused a little bit more overfitting, with C:=0.5 being the optimal setting. As for the smoothing parameter in the Naive Bayes classifier, we did not find any noticeable differences across the tested values. However, these results were obtained under the 70/10/20 split, and while it is expected, it is important to note that the optimal parameter settings and overall model performance also depend on how the dataset split.

All but the Linear Regression classifier performed similarly across the major preprocessing methods and hyperparameter settings. The relatively poor R^2 score of $\sim$ 86% 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 neighbourhood of 98%, across the SVM, Naive Bayes Classifier, and Logistic Regression model may be attributed to such clustering of classes. Thus, the choice of linear classifier is non-trivial, as its performance depends on whether the data points are clustered or linearly distributed according to some input, and if the prediction space is continuous or discrete.

5 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 tend to form seperate clusters of feature vectors, explaining the high accuracy of all the classifiers we tested. 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 the false information that occurs in a typical real-world dataset. Therefore, we cannot generalize the results of this study to the real world; the overall problem of distinguishing real from fake facts is not necessarly simple enough to be modelled by a linear classifier.