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 60/10/30 split. We utilized sklearn's relative term frequency vectorizer (`tfidf`), supplemented 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. We tested the reg-

ularization hyperparameter (inversely proportional to regularization strength) for the SVM and Logistic model with values in the range `0.25, 0.5, 1.0`, getting mixed results across the three major preprocessing techniques. As for the Naive Bayes classifier, we compared laplace smoothing with varying degrees of Lidstone smoothing, testing values in the range `0.25, 0.5, 1.0` for the smoothing parameter. No hyperparameter testing was performed for the linear regression classifier.

4 Results and Conclusions

All but the Linear Regression method performed similarly across all three hyperparameter settings and major preprocessing methods. One would expect improvement in accuracy going from Porter Stemmer to Lemmatization (as it is context-sensitive), and after supplementing `NLTK's WordNet Lemmatizer` with the Part of Speech tagger `pos_tag` (as the Lemmatizer would be more accurate). The relatively poor R^2 score of 86.7% from the linear regression classifier may be attributed to the fact that a feature vector tends to join a cluster of other data points in its class (for example, catergorizing someone as short or tall can be modelled more accurately by a linear regression model). On the other hand, the similarity in accuracy, in the neighbourhood of 98%, accross the SVM, Naive Bayes Classifier, and Logistic Regression model may be attributed to such clustering of classes. The choice of linear classifier is important, depending on whether the problem we aim to model is continuous or discrete, and if the data points are clustered or linearly distributed according to some input.

5 Limitations and Generalization

The primary limitation of this study is the simplicity of the dataset. The `fake` documents use a particular set of adjectives 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, only to make a claim that can be proved false only 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 data, it was not reasonable to assume that it's fake data would be representative of the fake data 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.