COMP 550 Assignment 1
Distinguishing Reals and Fakes Within Facts About Cities

Tian Yu Hao

McGill University School of Computer Science

Abstract

In this study, we explore the ability of linear classifiers to distinguish between real and fake facts about cities. Our objective is to evaluate the effectiveness of different preprocessing techniques and linear classifiers in identifying fake information generated about various cities, here we use Tokyo as a case study. Initial experiments showed that our linear classifiers performed well on the Tokyo dataset, but its performance did not generalize as effectively to a larger dataset covering multiple cities. Our findings suggest that more robust models may be needed for larger datasets. Namely, we propose to explore neural networks in a further study.

Dataset

Our project consists of two separate datasets. The first one, where we used facts about Tokyo as a case study, was generated by the ChatGPT 4o model and appended to a text file. The dataset is composed of 315 fakes and 315 facts about Tokyo, each distinct from one another. The format followed what was specified by the assignment instructions (each being about one line in length).

The second dataset was provided by a fellow student on Ed discussion named Julien Claveau. The dataset itself is comprised of 10 different cities with 50 fakes and 50 facts about each. All in all, totalling 1000 entries. The reason why we chose to test on a second, bigger dataset was to see if our initial models, which performed well on the one-city Tokyo dataset, would generalize to a more developed dataset about different places. Also, by doing this, we could test for any bias introduced by using the single prompt to generate the different facts about Tokyo. This point will be discussed further in the Results and Analysis section of our report.

The training/train split we went with is 80/20 ratio. Also keep in mind that we ran all of our experiments with 5-fold cross-validation to reduce over-fitting.

About Pre-processing

We ran two sets of experiments to evaluate the performance of Naive Bayes, SVM, and logistic regression models. In the first set, we applied pre-processing techniques such as stopword removal and lemmatization, while in the second set, we cleaned the data but did not apply these techniques. Other pre-processing techniques we used all throughout are decapitalization, number and punctuation removal, and bagging.

For the model with all pre-processing techniques applied, the test accuracy for Naive Bayes was 0.9571, for SVM was 0.9524, and for Logistic Regression was 0.9548. These results indicate a good baseline performance.

Interestingly, when we skipped stopword removal and lemmatization, we observed higher test accuracies for Naive Bayes (0.9889), SVM (0.9667), and Logistic Regression (0.9611). This result suggests that removing stopwords and lemmatization might have removed important information, especially given that our dataset consists of relatively short text samples.

The other possibility is that in generating our Tokyo dataset, we introduced an enormous amount of bias by using the same model and technique all throughout. However, that seems to be disproven when we acknowledge that the accuracy also went up immensely in our second phase of testing on generalizing the models with the limited pre-processing applied.

Note: All experiments were run with the TD-IDF strategy.

Hyperparameter tuning

We performed hyperparameter tuning on our models using Sklearn's GridSearch function on the validation set. We tested for a range of 20 different values for the alpha parameter for Naive Bayes, and our regularization strength in SVM and logistic regression. We found the best parameters, appended them to a dictionary and used those in the final training and testing of our models with the (experimentally found) best hyperparameters.

Results and Analysis

First, we did try to run our hand-coded models on the dataset, however due to the slower runtime and overall lower accuracy, we reverted back into using Sklearn's built-in multinomial Naive Bayes, SVM, and logistic regression models.

In the Tokyo dataset, the best test accuracies achieved were 0.9889 for Naive Bayes, 0.9667 for SVM, and 0.9611 for Logistic Regression. However, when we applied the same models to a larger, more diverse dataset, the test accuracies dropped significantly. Naive Bayes achieved 0.8267, SVM reached 0.8400, and Logistic Regression obtained 0.8233.

These results indicate that while the models performed well on the Tokyo dataset, they failed to generalize effectively to the broader dataset. The significant drop in accuracy across all models suggests that our feature extraction and model tuning were likely not suitable or not adequate enough to a larger dataset. In theory, when thinking about linear classifiers, they are indeed limited by the amount of information they can encapsulate. We did try looking into why the larger dataset ended up with worse accuracy by looking at the highest frequency features in our dataset, but we didn't find anything out of the ordinary.

As to the earlier segment about the higher accuracy when using less pre-processing, we think it might be linked to the way that GPT uses stopwords, punctuation, and un-lemmatized words in the generation of its facts and fakes where we simply fail to capture the inherent bias and lose that extra information on how LLM's generate certain types of data. When they are present, our models then take advantage of the inherent bias to achieve better prediction accuracy. (Which is not necessarily a good thing since we want our models to be able to generalize over all matters of speech bias and not just ones in algorithmically generated statements)

Conclusion

Overall, the results of the study were as expected. We found that across the Tokyo dataset, Naive Bayes outperformed the other models, likely due to the small dataset size which suits its simplicity. However, on the larger and more diverse generalization dataset, we found that the best performing model was SVM. Note that the accuracy on the generalization dataset is still much worse than even the worse performing model on the simpler Tokyo dataset. This would suggest that using linear classifiers is not suitable if our goal is to generalize to even bigger and more complex datasets.

Another inherent problem in linguistics is that in a sentence, a difference of merely one word, one symbol of punctuation, or even just changing the order of the words can change a fact to a fake statement. More broadly even, to try to determine the veracity of any and all statements would need one to be omniscient, as a statement can be true one second and fake the next. With this in mind, we think that in a future experiment, it would be interesting to test using a different data generation method, something along the lines of: "List facts and fakes about a certain city that tread the line as much as possible between real and fake". Then run this on the state-of-the-art transformer models we have and see how they perform. However, even then, we doubt we would get anything even remotely usable in industry for detecting real world facts and fakes due to the sheer complexity of the task.

References

1. Presenting the final experiment results in a clearer format (sorry not enough space for a proper table)