Learning to predict the stance of news articles

Justin Ty z5232245

Introduction

The prevelance of fake news in the media has been an increasing issue in recent times. A Fake News Challenge (FNC) was created in 2017 in an attempt to address this issue by using automated systems. The goal of the challenge was to build an AI system that could automatically detect the stance of news articles. This project for COMP9417 was an attempt to build an end-to-end process which could take news headlines and articles to predict their stance.

Implementation

Dataset and Labels

The provided dataset from the challenge contained bodies and headlines from news articles. The goal of the challenge was to build a classifier that could detect the stance of the news article. The stance of the article can be one of the following:

- Agree - If the headline and article body agree on the topic
- Disagree - If the headline and article body disagree
- Discusses - If there was a discussion of the topic
- Unrelated - If the headline and body were not related.

The nature of the problem could have been modelled as classifications with multiple stages. However, Chaudhry (2018) noted that this does not improve results and only adds complexity. As such, the models in this project treated this as a one stage 4-class classification problem.

Feature Engineering

The main approach for the features was to use a bag of words representation of words in the article headlines and bodies. A TFIDF Vectorizer was used to generate a bag-of-words feature set. This tranformation made more frequent words and stop words less important, while giving less occuring words more weight.

A large vocabulary with a one-hot-encoding resulted in a very sparse dataset. In order to improve performance and reduce noise, dimensionality reduction was applied to the bag of words using Latent Semantic Analysis.

Additionally, some feature engineering scripts were borrowed from the original challenge. Cooccurence of words between were the main features created by the original utility functions from the challenge. The borrowed features from the original repository include the count of refutes and overlaps between the article body and headline.

Modelling and cross validation

A simple benchmark was a Simple Decision Tree. To avoid the overfitting of a single decision tree learner, validation plots were used to tune its hyper parameters using cross validation. The hyper parameters include the tree's max depth and minimum items for a leaf. Validation and learning curves can be seen in the appendix.

Ensemble methods that were implemented included XGBoost, Random Forest, and Gradient Boosting model. The baseline from the original challenge was a Gradient Boosting Model with 200 base estimators.

In addition, XGBoost was also trained and tested. It has the benefit of being fast and scalable with the focus on speed and performance. It also contains parameters which help against overfitting such as early stopping. In this project, if the training error did not reduce after 5 iterations, the training process stops. This prevents overfitting from the training dataset alone. If the data is structured properly, it can be very efficient for sparse data which is suitable for a bag-of-words approach.

Neural networks and gradient boost proved to be popular and effective in the competition. SOLAT in the SWEN (Sean et al. 2017) by Talos Intelligence was the winner of the FNC which combined both Gradient Boosting and a Convolutional Neural Network with Google's pretained Word2Vec. The second, Athene (Hanselowski 2017) also used an ensemble of Multilayer Linear Perceptrons.

A fully connected multilinear perceptron neural network was implemented in this project. The implementation was a fully connected neural network that generates outputs through a softmax activation function. No ensembling of neural networks was done in this project, but this would be a good extension.

Scoring

Some scoring utility functions were borrowed from the challenge repository in order to compare this project's models against the baselines. This is referred to as the FNC scores.

Hanselowski (2018) noted that the original scoring functions were not appropriate for validating the document-level stance detection task. Instead, it was their recommendation to use F1 score with macro averaging. This is so that the scores are not affected by majority classes.

Findings by Hanselowski (2018) mentioned that some classifiers do not create predictions for the Disagree class while yielding high scores. The majority classes were Discuss and Unrelated which the FNC score tend to favour.

For this project, the FNC, F1, and accuracy scores, and training times were recorded. The detailed logs of the application were also saved to in the archive that shows their confusion matrices.

Experimentation

A summary of FNC scores, F1 Scores, and Accuracy is listed below.

Scores from this project

System	FNC	F1 Macro	Accuracy
Simple Decision Tree	52.66%	31.96%	57.55%
Gradient Boosting Model (FNC baseline parameters)	64.75%	39.08%	77.41%
Gradient Boosting Model (Tuned parameters)	65.23%	40.22%	80.05%
XGBoost	65.84%	38.66%	80.34%
Random Forest	40.14%	21.92%	72.52%
Dense Neural Network	61.13%	38.15%	75.10%

For comparison, Hanselowski (2018) recorded the FNC and F1 macros scores of the challenge participants as follows:

Scores of the original challenge participants

System	FNC	F1 Macro	Accuracy
Talos Comb	82.00%	58.20%	NA
Talso Tree	83.00%	57.00%	NA
Talos CNN	50.20%	30.80%	NA
Athene	82.00%	60.40%	NA

The FNC and F1 scores of this project do not exceed that of the winners of the competition. This can be attributed to the competition winners use pretrained vectors and better feature engineering from an NLP background.

As expected, the simple decision tree showed overfitting where its training scores were very high yet its validation scores scored low. The learning curves in the appendix confirm this.

In terms of performance and accuracy, boosting models delivered the most accurate results with reasonable performance. Boosted ensemble methods seem to work very well especially for an unbalanced dataset such as this. Both the gradient boosting classifier and XGBoost ranked top of the models from this project.

Random forest, while scored high on accuracy, predicted tend to pick majority classes over less occuring Agree and Disgree classes.

The simple neural network in this project does not score as well as the other ensemble models. This may be due to the lack of quality features. This could be futher improved with more background in NLP feature engineering. Ensembling networks could also improve results as done by other submissions to the challenge.

Hanselowski (2018) noted that neural networks tend to score well for the FNC score but do not create as much predictions for the Disgree class, yielding lower F1 scores. These observations were also made on the neural network trained on this project.

Conclusion and further recommendation

This project has shown an end-to-end process of feature engineering and predicting the stance of news articles without the use of pretrained materials. The process included transforming raw text data into features and training a predictive model from them .

This project was meant for education purposes without using pretained materials. In terms of FNC and F1 scores, the models from this project did not score as high as the winners of the original challenge.

On the other hand, the test accuracy of the models from this project were reasonably good. Ensemble estimators from this project have drawn out the best results. GBM and XGBoost were able to predict the correct stance of 80% of the test data.

Futher improvements could be done with this project. Areas of improvement can be done with better featuring engineering from an NLP background.

References

Andreas Hanselowski , et al (2018) *A Retrospective Analysis of the Fake News Challenge Stance Detection Task*, : Research Training Group AIPHES Computer Science Department, Technische Universität Darmstadt. https://arxiv.org/pdf/1806.05180.pdf

Melanie Tosik, et al (2018) *Debunking Fake News One Feature at a Time*, : Department of Computer Science New York University. https://arxiv.org/pdf/1808.02831.pdf

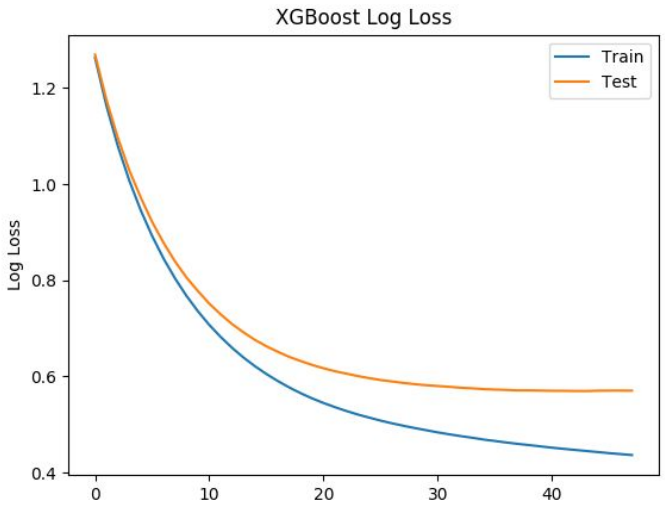
Ali K. Chaudhry, Darren Baker, Philipp Thun-Hohenstein (2018) *Stance Detection for the Fake News Challenge: Identifying Textual Relationships with Deep Neural Nets*, : Stanford. https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1174/reports/2760230.pdf

Sean , Baird, Doug Sibley, Yuxi Pan (2018) *Talos Targets Disinformation with Fake News Challenge Victory,* Talos Intelligence https://blog.talosintelligence.com/2017/06/talos-fake-news-challenge.html

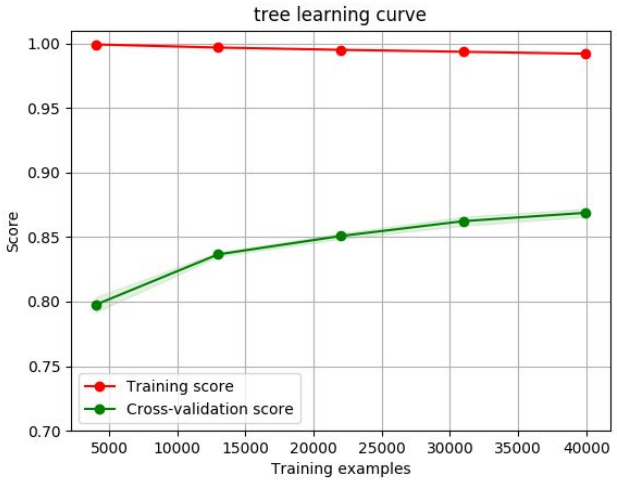
Fake New Challenge http://www.fakenewschallenge.org/

Tianqi Chen and Carlos Guestrin. (2016). *XGBoost: A Scalable Tree Boosting System*. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16). ACM, New York, NY, USA, 785-794. DOI: https://doi.org/10.1145/2939672.2939785

Appendix

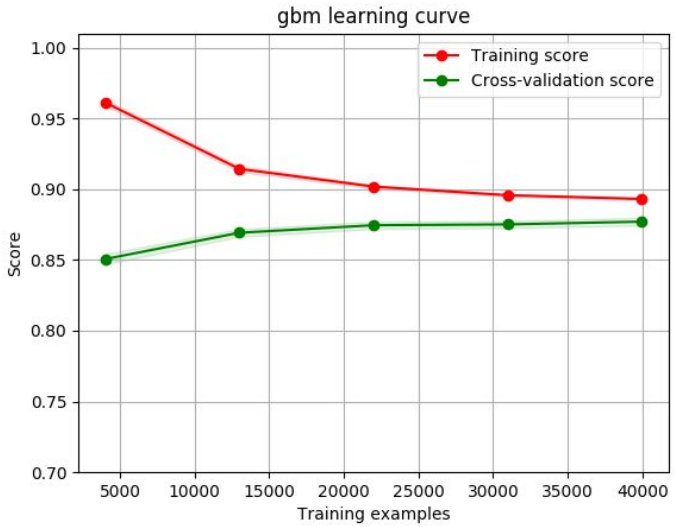


An early stop[1] was applied to XGBoost. This prevents anymore learning to avoid overfitting. If the log loss (cross entropy) does not decrease after 5 rounds, then the training stops.

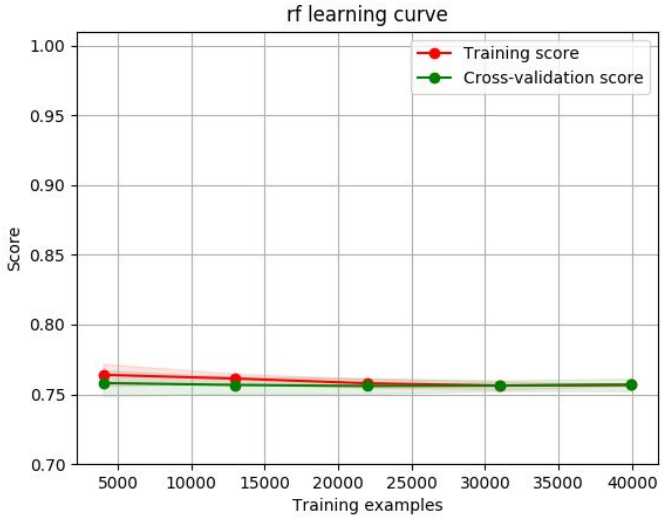


[1] https://machinelearningmastery.com/avoid-overfitting-by-early-stopping-with-xgboost-in-python/

The learning curve[2] that the Simple Decision Tree overfits the training set. It scores almost 100% on the training set while it's cross validation score remained low.

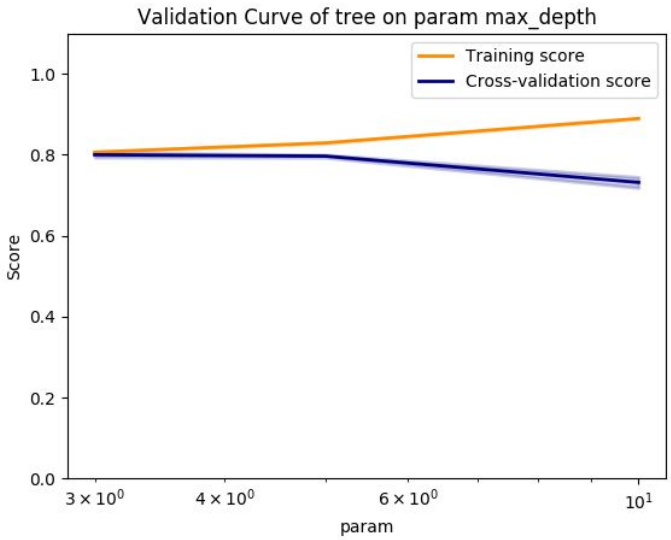


On the otherhand, a gradient boosting model had a good learning curve. It is a sign that the model does not overfit.

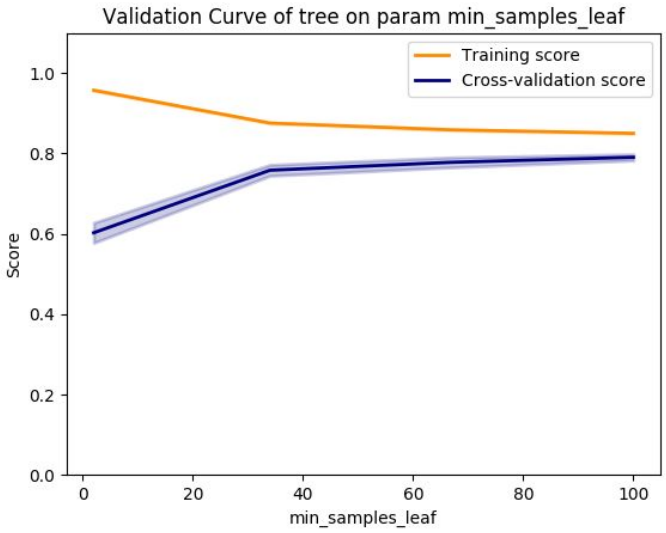


The random forest learning curves were similar for train and test. The accuracies were similar but the classifier was mostly predicting the majority classes.

[2] https://scikit-learn.org/stable/auto_examples/model_selection/plot_validation_curve.html



The validation plot[3] suggests that the max depth for a tree is around 5. Any deeper, the model will overfit the training set but degrade on cross validation.



This suggests that the minimum number of samples in the leaf should be around 50.

3 https://scikit-learn.org/stable/auto_examples/model_selection/plot_validation_curve.html#sphx-glr-auto-examples-model-selection-plot-validation-curve-py

```
Training a gbm model

Training time took 266.0019030570984 seconds

Trained a model using GradientBoostingClassifier(criterion='friedman_mse', init=None,
                           learning_rate=0.1, loss='deviance', max_depth=3,
                           max_features=None, max_leaf_nodes=None,
                           min_impurity_decrease=0.0, min_impurity_split=None,
                           min_samples_leaf=1, min_samples_split=2,
                           min_weight_fraction_leaf=0.0, n_estimators=200,
                           n_iter_no_change=None, presort='auto',
                           random_state=None, subsample=1.0, tol=0.0001,
                           validation_fraction=0.1, verbose=0,
                           warm_start=False)
Train FNC Score:
-------------------------------------------------------------
|           |   agree   | disagree  |  discuss  | unrelated |
-------------------------------------------------------------
|   agree   |   1906    |    101    |    805    |    866    |
-------------------------------------------------------------
| disagree  |    132    |    321    |    165    |    222    |
-------------------------------------------------------------
|  discuss  |    291    |    47     |   6965    |   1606    |
-------------------------------------------------------------
| unrelated |    208    |    40     |   1009    |   35288   |
-------------------------------------------------------------
Score: 18399.25 out of 22563.25 (81.54521179351379%)
Train F1 Score: 0.70412738590683
Train Accuracy Score: 0.8900984551348755

Test FNC Score:
-------------------------------------------------------------
|           |   agree   | disagree  |  discuss  | unrelated |
-------------------------------------------------------------
|   agree   |    140    |    28     |   1147    |    588    |
-------------------------------------------------------------
| disagree  |    25     |     2     |    336    |    334    |
-------------------------------------------------------------
|  discuss  |    548    |    57     |   2642    |   1217    |
-------------------------------------------------------------
| unrelated |    198    |    259    |   1023    |   16869   |
-------------------------------------------------------------
Score: 7536.5 out of 11651.25   (64.68404677609699%)
Test F1 Score: 0.3890477489877572
Test Accuracy Score: 0.7733443513162555
```

This is a sample output of a logs from a Gradient Boosting Classifier. The logs folder contain the output from the different models.