Predicting Fake Job Listings from Real Ones using Machine Learning Models

Ronit Munshi

3/23/24

Abstract

Nowadays, there are more and more fake job positions appearing online, whether that be companies listing more job positions than they actually need or malicious actors listing out job positions that are meant to take personal information. The main way this problem is going to be solved is through the creation of models that can predict whether these job listings are real or fake. The procedure involves finding the dataset needed to train the model. [2]After the preprocessing, the different NLP models were tested out to see which one would end up giving the best prediction, with the only ones that ended up working in the end being a Logistic Regression Model and a BERT Model. The accuracy of the Logistic Regression model ended up being 99.48%,. And after running the data through Random Forest Hyperparameter Tuning, the model was left with a mean absolute error of 3.37% (low values are good!).[8] The accuracy of the BERT model ended up being 99.81%.Overall, the accuracy of the logistic regression model on its own was excellent in separating the fake job listings from the real ones, showing its capability in that regard. Though, possible potential ways to further improve the accuracy to a greater level would be through implementing a LSTM model, given how it can be more accurate than Logistic Regression and as accurate as a BERT model.[11] Though, both require their own amount of preprocessing of the data that is separate from the main preprocessing already completed. Alongside that, it would be beneficial to have the models be able to take in input and be easily accessible so that anyone who is concerned about whether or not a job position is real or fake can check through imputing the information into a search box and submitting it, giving the models more data to use to improve their accuracy along with giving the public peace of mind to know that their data isn't going to get stolen.

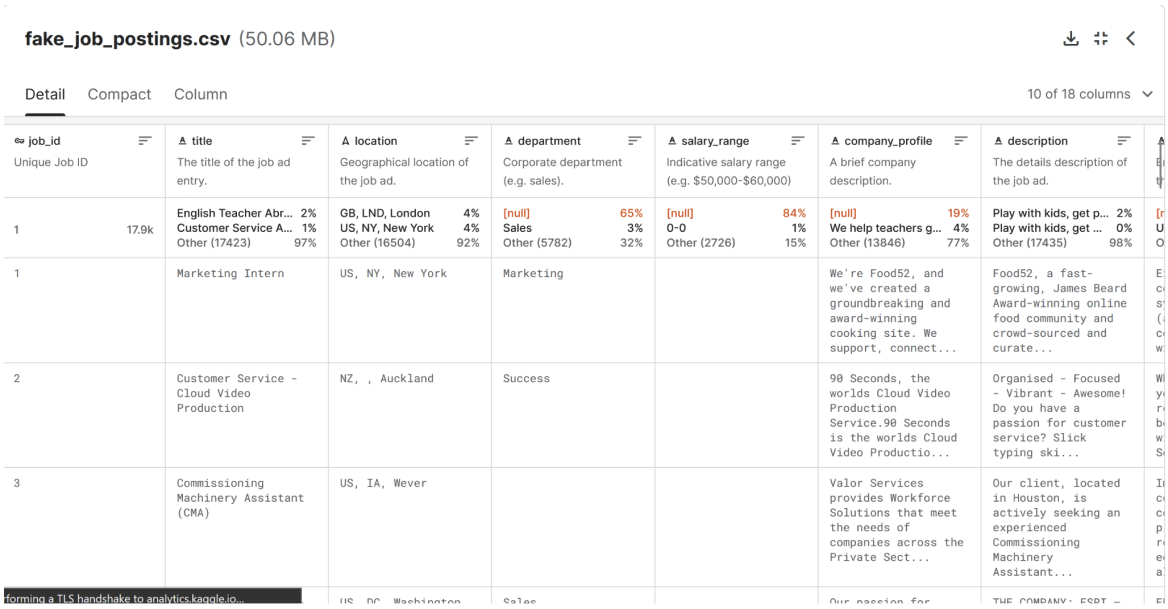
1. Introduction

In recent times, there have been many issues in regards to potential employees seeing and applying for job listings that are fake. Such reasons for this involve scammers and malicious actors putting up fake job listings in order to get personal information from potential employees looking for a job. [1]The nature of this model development itself will be supervised, primarily through the use of NLP models, given the fact the data primarily consists of categorical data and that the categories of information in the dataset have to be compared to the one column in the data itself that states whether or not the job listings are real or fake. The output of the project itself should consist of the prediction accuracy, the mean value error, as well as the specific numbers provided from the confusion matrix generated from the models, which would consist of the number of correctly predicted real jobs, incorrectly predicted real jobs, correctly predicted fake jobs, and incorrectly predicted fake jobs.

2. Background

One approach that has been used to solve the research problem was done by Mehadi Hasan, who created a Bi-Directional LSTM model that would accomplish the same goal as the different models that will be explained further down below, but to preview the models that will be created are a Logistic Regression Model as well as a BERT model. The pros of his approach mainly involve that LSTM models are in general more accurate than a simple Logistic Regression Model and around on the same level of accuracy as a BERT model. But the accuracy of his model ended up being only around 97% in terms of predicting which job listings were real and which ended up being fake, which is still very high.[16] But as reviewed in the results, the accuracy of the Logistic Regression model on its own ended up being more accurate, with an accuracy score of 99.45%. Another approach taken was by KZ Data Lover who tried to solve the same problem using a KNIME workflow, which yielded him an overall accuracy score of 96%, again which is high but not as much as the previewed 99.45% accuracy score of the logistic regression model on its own. [15] These other approaches relate to this project's approach given that it involves creating multiple NLP models to see which one ends up being more accurate in the end. In addition, both previous approaches employ preprocessing techniques that will be used in this project, mainly processes that ocean up the data like a lemmatization process, bag of words, stopword filter, etc.

3. Dataset



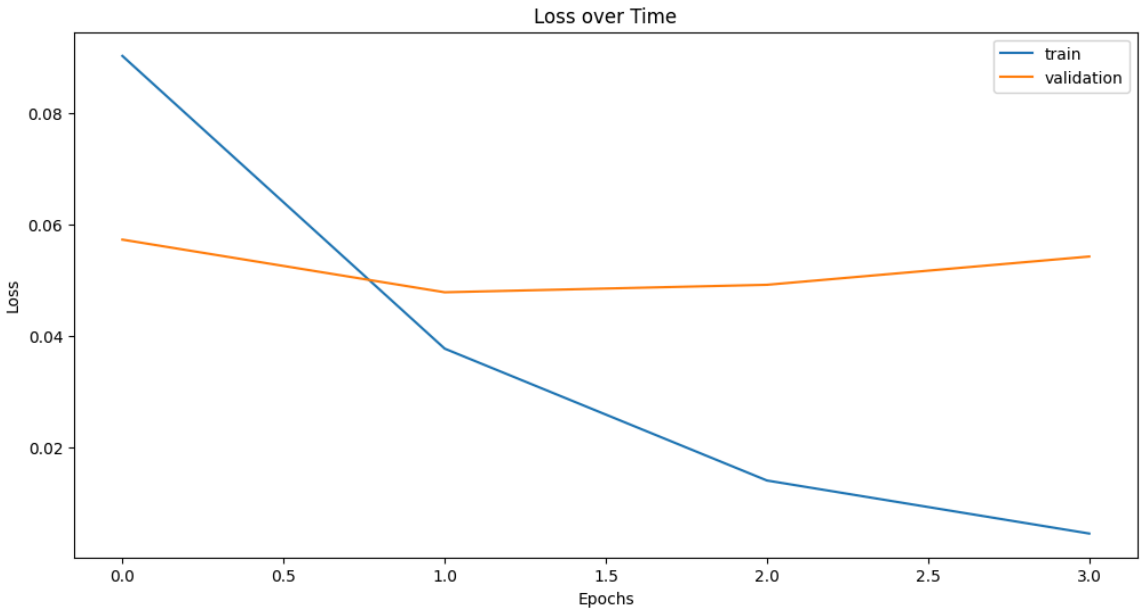
The dataset used, called Fake/Real Job Position predictions,[1] contains the data of 18,000 different job listings, with 800 of those being fake job listings . The dataset technically

involved both language and numerical data, with most of the language data encompassing the job titles, as well as the different aspects of the job itself, like the description, company profile, salary range, etc, with the numerical data mainly encompassing the part of the dataset that depicts whether or not the job position is real or if it is fake. Before any preprocessing, the shape of the data was (17880, 18). In addition, before preprocessing the data, an inspection over the whole dataset was taken to see which aspects of it would be beneficial to the training of the model given that not all the different information provided about each job would be useful in differentiating which are real and which are fake. The specific description columns of the dataset that ended up being used were the job description, company profile, and job requirements, along with the column denoting if the job position was real or fake. From here, the preprocessing of the data began. First, the three columns of data were combined into one and ran through a dropna that would remove any null values, leading to a final shape of (12631, 2).[13] Next, the combined column was ran through a lemmatization process to change certain words within the dataset to their more common forms.[10]Then, a train test split was set up, with it being a 80 to 20 split between testing and training, the X_train and X_test using the combined column, while the y_train and y_test using the column representing whether or not the job listing is real or fake.[4]Finally, the X_train and X_test data were run through a count vectorizer in order to turn the language data into numerical data so that it could properly be compared with the numerical data of 1s and 0s being used to represent if the data is real or fake.[3] The preprocessing of the data for the BERT model will be expanded on later on in this paper.

4. Methodology / Models

The main two models that used to solve the problem were a Logistic Regression Model and the BERT model. The way the logistic regression model primarily works is through primarily finding the relationship between the data in the x_train and y_train, and then applying that relationship data to make predictions about the relationships between the data in the x_test and y_test. The way the model learning procedure went, as mentioned in the dataset section of this paper, for the Logistic Regression Model primarily revolved around a 80% testing and 20% training split, with 20% of the combined column of data being used to train the model for understanding which job positions are real and fake when comparing it to the 20% of the y-train containing the data about real or fake. .[4] The data preprocessing steps again included lemmatization of both X_train and X_test data, removal of null values using dropna, and conversion of the data into 1s and 0s using a count vectorizer. This made for easy comparisons with the y_train and y_test data.[3,4,10] To set up the Logistic Regression Model, many Python libraries such as sklearn and pandas were used among others. [5] After the main model was created, the model was run through a hyper parameter tuning process, mainly through that of random forest, with the results of said tuning being described after this section.[8]The way the BERT model works that makes it different from other models is

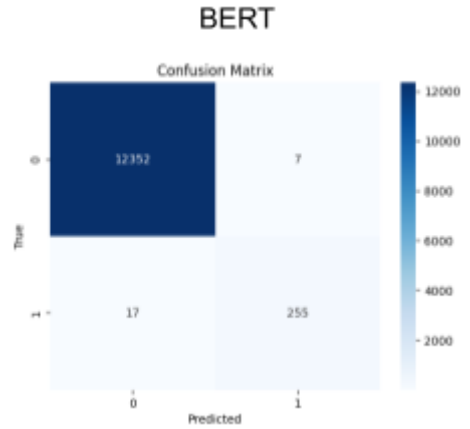
that it reads data bidirectionally in comparison to most other NLP models that only can read data from one direction, right or left. This gives BERT models the context of the sentence data it receives from what precedes it to what comes after it, giving it a richer idea of what a word means in one part of a sentence in comparison to other use cases of the word in other parts of the data. Now going into the model development process, the preprocessing first involved tokenizing the combined column data in order to be able map them to an index with BERT's vocab using the Bert Tokenizer from the Transformer python library along with converting the sentence data into numerical data. From there, the tokenized data was then modified in order to be able to be used by BERT. This was done mainly through applying the CLS and SEP tokens to the start and end of each sentence respectively, padding and truncating the sentences so that they are all the same length, along with using attention masks to be able to differentiate the padding data from the real sentence data. Then, train test splits were set up, with the data being split into training and validation data, with there being a 85% validation and a 15% training split in the data. The same thing was done with the attention masks, with those being split up into training and validation masks, with there being a 85% validation and a 15% training split in the data as well. The data was also converted into tensors so that it could be easily read and understood by the BERT model. In addition, a data loader was made in order for the feeding of the data into the model to be more streamlined, which completes the preprocessing. [17]From here, the training of the model begins. The BERT model used in this case was a BertForSequenceClassification model, which itself is a standard BERT model that just adds a single linear layer on top. During training, the pre-trained BERT model and the additional untrained classification layer learn from the data as it is fed into them in order to help understand what the main classification job of the model is. Before the training begins, though, hyperparameters also have to be considered, mainly the learning rate and epochs, with the learning rate being set to 2e-5 and the epochs to 4 in order to avoid overfitting and underfitting, which could have occurred if these two hyperparameters weren't set up correctly.After this, the model training began, with the training going through the steps of unpacking the tokenized data inputs as well as their labels as they are coming through the data loaders, clearing out any weights and biases from the previous pass, running the data through a forward pass that feeds the data through the network, running the data through a backward pass (backwards propagation,)and finally having the hyperparameters fully updated accordingly to the two passes. These steps were done 4 times through the 4 epochs.[12] After the training was completed, the training loss and validation loss, which showed the error the models made in classifying, were graphed over the 4 epochs, as shown below, with the training loss at each epoch being 0.09, 0.04, 0.01, and 0.00, and the validation loss being 0.06, 0.05, 0.05, and 0.05.



In addition, the validation accuracy in each epoch was 0.98, 0.99, 0.99, and 0.99. From there, the accuracy score was found, which will be discussed along with the Logistic Regression model's results in the next section of this paper.

5. Results and Discussion

The outcome of the models overall was positive, with there being a 99.45% accuracy score for the logistic regression model as well as a mean absolute error of 3.27% after running the model through a random forest hyper parameter process.The BERT model has an accuracy score of 99.81% . The Logistic Regression model's confusion matrix showed that from the testing data, 2461 real jobs were predicted correctly as real, 53 fake jobs were predicted correctly as fake, 9 real jobs were predicted wrongly as fake, and 4 fake jobs were predicted wrongly as real.[9]The confusion matrix of the BERT model showed that based on the testing data, 12,352 real jobs were predicted correctly as real, 255 fake jobs were predicted correctly as fake, 17 real jobs were predicted correctly as fake, and 7 fake jobs were predicted correctly as real. Other than these models, Also attempted to be implemented were a KNN and Decision Tree classifier model to act as additional sources of data.[6] [7]But, both models faced the same



errors that that weren't fully able to be rid off in order to get a tangible result, given that both would experience an error in regards to the shape, stating that they " Found input variables with inconsistent numbers of samples: [10104, 2527] ," with the shape being "odd", in the sense that there wasn't an even number of data points from the dataset that could be compared between each other. Given that data points shouldn't have been removed didn't want to have to remove data points, But in terms of looking at the results of the models themselves, they both performed well in separating, for the most part, the fake job listings from the real job listings given the high accuracy scores and the lower mean absolute errors after the random forest hyper parameter tuning. [8]

6. Conclusions

Overall, the main purpose of this project was to create at least one machine learning model that can predicts whether or not job listings that are posted online are real or fake, primarily based on the dataset, Real/Fake Job Posting Prediction, which contains the information of 18,800 job listings, 800 of those being fake jobs. The main significance of this project is to provide the foundation for solving the problem of the ease of people being tricked by fake listings and applying to those "jobs" and then subsequently having their data stolen by being able to identify and separate the real job listings from the fake job. Currently, the Logistic Regression and BERT models can only give predictions based on the dataset they were trained on, with neither being able to take in input. And overall, the results of those models were successful, with the Logistic Regression Model having an accuracy score of 99.45% and a mean absolute error of 3.372%, and the BERT model having an accuracy score of 99.81%. But in regards to the next steps for the research, a potential future would be to try to create more models using different machine learning algorithms in order to see how they compare to the BERT and the Logistic Regression Model, in addition to trying to use a platform like Streamlit in order to be able to deploy the model for the public to use, with the goal being to have a search box that anyone could easily plug in information from a job listing into in order to check if the job listing is likely real or if there is a higher chance of it being fake.

Acknowledgments

I would like to thank Hassan Azmat who helped with all the issues I had throughout the development of the models in regards to guiding me throughout the creation of the two models themselves, as well as through the creation and writing of this research paper as well. I would also like to thank Inspirit AI for allowing me to connect with Hassan and giving me a chance to work on the project with assistance from him.

References

1. Bansal S. Real / Fake Job Posting Prediction. Kaggle.com. Published February 28, 2020. Accessed November 5, 2023. https://www.kaggle.com/datasets/shivamb/real-or-fake-fake-jobposting-prediction
2. scikit-learn. sklearn.linear_model.LogisticRegression — scikit-learn 0.21.2 documentation. Scikit-learn.org. Published 2014. https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html
3. Using CountVectorizer to Extracting Features from Text. GeeksforGeeks. Published July 15, 2020. https://www.geeksforgeeks.org/using-countvectorizer-to-extracting-features-from-text/
4. scikit-learn. sklearn.model_selection.train_test_split — scikit-learn 0.20.3 documentation. Scikit-learn.org. Published 2018. https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html
5. Galarnyk M. Logistic Regression using Python (scikit-learn). Medium. Published April 29, 2020. https://towardsdatascience.com/logistic-regression-using-python-sklearn-numpy-mnist-handwriting-recognition-matplotlib-a6b31e2b166a
6. scikit-learn. 1.10. Decision Trees — scikit-learn 0.22 documentation. Scikit-learn.org. Published 2009. https://scikit-learn.org/stable/modules/tree.html
7. scikit-learn . sklearn.neighbors.KNeighborsClassifier — scikit-learn 0.22.1 documentation. Scikit-learn.org. Published 2019. https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html
8. scikit-learn. 3.2.4.3.2. sklearn.ensemble.RandomForestRegressor — scikit-learn 0.20.3 documentation. Scikit-learn.org. Published 2018. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html
9. scikit-learn. sklearn.metrics.confusion_matrix — scikit-learn 0.21.3 documentation. Scikit-learn.org. Published 2019. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html
10. Pykes K. Stemming and Lemmatization in Python. www.datacamp.com. Published February 2023. https://www.datacamp.com/tutorial/stemming-lemmatization-python
11. Otten NV. How To Use LSTM In NLP Tasks With A Text Classification Example Using Keras. Spot Intelligence. Published January 11, 2023. https://spotintelligence.com/2023/01/11/lstm-in-nlp-tasks/
12. BERT. huggingface.co. Accessed March 20, 2024. https://huggingface.co/docs/transformers/model_doc/bert#transformers.BertForSequenceClassification
13. Ivanova I. Fake job listings are a growing problem in the labor market. www.cbsnews.com. Published March 30, 2023.

https://www.cbsnews.com/news/job-openings-fake-listings-ads-federal-reserve-jolts/
14. pandas.DataFrame.dropna — pandas 1.3.1 documentation. pandas.pydata.org. https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.dropna.html
15. Job Posting Prediction with KNIME (96% Accuracy). kaggle.com. Accessed March 21, 2024. https://www.kaggle.com/code/kzmontage/job-posting-prediction-with-knime-96-accuracy
16. Detect fake job news using BiLSTM 97 % accuracy. kaggle.com. Accessed March 21, 2024. https://www.kaggle.com/code/islamic/detect-fake-job-news-using-bilstm-97-accuracy
17. .Lendave V. A Guide to Text Preprocessing Using BERT. Analytics India Magazine. Published September 19, 2021. https://analyticsindiamag.com/a-guide-to-text-preprocessing-using-bert/