Detecting Fake Job Postings with Text Classification

Lanqi Fei

Abstract

This paper aims to predict fake job postings with binary text classification techniques, based on company profile, job description and some complementary features. The main techniques involved in the paper include Logistic Regression, Support Vector Machine (SVM), and Convolutional Neural Networks (CNN). Multimodal approaches that combine textual and meta features to make inferences are also explored to compare with classical models. All source code could be found in Github via https://github.com/MSIA/lfq4864_msia_text_analytics_2020

1 Introduction

As social media becomes popular and more platforms arise for job search, nowadays people have easy access to a wide range of job postings. With the ease of companies and individuals posting jobs on websites, it is essential to identify which of them could be fraudulent. Platforms that offer job posting services are also placed great responsibility for filtering out fraudulent job postings to keep user safe from theft of personal information and maintain healthy user retention, and therefore it would be of great help if machine learning can help automatically flag and filter out fraudulent jobs for audiences by identifying important signals. In this paper, several text classification models are explored and compared to build a fake job posting detector.

2 Related Work

Text classification, being one of the most popular areas of natural language processing, has experienced great advances in industry and research in a wide range of topics. Consequently, it is no surprise that there is already research on predicting fake jobs using machine learning. Ensemble classifiers are found to be the most effective approach (Bandyopadhyay et al., 2020), and data cleaning is also essential for improving performance (Abuta et al., 2021) in this area.

Despite the fact that there is some existing study, there is still not extensive research on this topic, and specifically, there is not much research that uses an imbalanced dataset, while in reality, only a small fraction of the job postings will be fraudulent. Moreover, most of the dataset used by earlier research mainly consists of features like job description, but incorporating some meta data like salary level, and whether the job post comes with a company logo could be useful.

In terms of text classification model architectures, vast research has been done. In addition to Logistic Regression, which serves as a good baseline model for most classification problems, Support Vector Machine is another model that proves to be able to enable automation in various fields. In recent years, more researchers are interested in Convolutional Neural Network for sentence classification, due to its nature of local focus which can also be applied to text. Kim states in his paper that a comparatively simple CNN along with hyperparameter tuning and static vectors could yield great performance on multiple benchmarks (Kim, 2014). In this paper, the goal is to conduct experiments on various popular methods based on these previous research and compare their performance and apply to fake job postings detection.

3 Dataset

The dataset consists of 17,880 observations, and only 5% is flagged as fraudulent. The main textual features include the job title, company profile, job description, job requirements, and some additional meta features including expected salary, required education are also provided if they exist.

In this paper, the three textual columns, company profile, job description, and job requirements will be used as input features for classification. Job title here is not used to avoid potential bias, since the classifier should not depend on the title of position such as “Data Entry Specialist” to make inference of whether a job is fake or not. In the end, some methods that combine textual and meta features to make predictions are further explored.

4 Method

4.1 Preprocessing

Since the dataset is imbalanced, upsampling for the minority class is performed, resulting in 1:1 class ratio. Three textual columns of focus, company profile, job description and job requirements are concatenated as one string named column `text`. If all three columns are missing, then the `text` column is marked as ‘Missing’. Punctuation, special characters, stopwords are removed from text, stemming is performed so that each word will be transformed to its corresponding base form, and lastly, all words are turned into lowercases.

To perform classical text classification algorithms, feature vectorization needs to be performed in advance before feeding to the models. In this paper, TF-IDF vector representation is used.

For basic models including Logistic Regression, Support Vector Machine, unigram, bigram and mixed-gram (combining unigrams and bigrams) representations are created before feeding into the TF-IDF transformer.

In addition, to make different models comparable, the data is split into training and testing beforehand, with 70% training and 30% testing data. The models will be trained on the same training data and tested on the same testing data.

4.2 Exploratory Data Analysis

It is always helpful to understand the dataset better and perform some initial exploratory data analysis before modeling.

Segmenting the job postings by job function, one can examine which job categories are the ones with high fraudulent frequencies. Based on the following chart, it indicates that Administrative, Engineering and Customer Services are the top 3 fraudulent job functions.

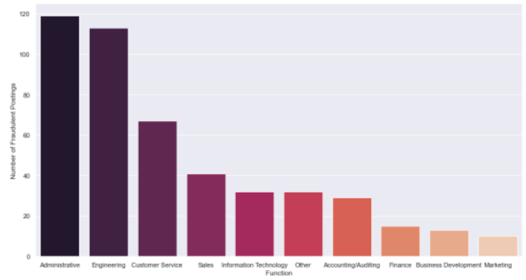

Figure 1: Top 10 Fraudulent Job Functions

Even though the column `title` also indicates the function of the job posted, it turns out that most of the titles only occur once in the dataset. In this case, graphing the word cloud for fraudulent job titles could help depict which job titles are associated with high fraudulent risks.

Figure 2: Fraudulent Job Title Word Cloud

Based on Figure 2, one can notice that some jobs with high risk of being fraudulent include data entry, assistant, and so forth. This also aligns with our intuition since many fraudulent jobs are also the ones that are less technical or lower paid.

4.3 Logistic Regression

Logistic regression usually serves as a good benchmark model to compare with other models. In addition, it requires minimal training time and computation power.

To test the performance of logistic regression, with different bag-of-words representation, penalty function and regularization strength (C), twelve experiments are performed, and some results are summarized in the following table. Note that all metrics are rounded to 2 decimal places.

Feature	Penalty	C	Accuracy	F1
unigram	elasticnet	1	0.89	0.89
unigram	l2	.5	0.89	0.89
bigram	elasticnet	.5	0.76	0.75
mixed	elasticnet	1	0.86	0.86

Table 2: Test Score for Logistic Regression

4.4 Support Vector Machine (SVM)

Support vector machine is another popular text classifier as it is computationally efficient and requires less amount of text to train.

To test the performance of SVM, with different bag-of-words representation, loss and regularization strength (alpha), eighteen experiments are performed, and some results are summarized in the following table.

Feature	loss	alpha	Accuracy	F1
unigram	squared hinge	1e-6	0.94	0.94
unigram	hinge	1e-6	0.94	0.94
bigram	squared hinge	1e-6	0.80	0.79
mixed	hinge	1e-6	0.90	0.90

Table 2: Test Score for SVM

4.5 Convolutional Neural Network (CNN)

Convolutional neural network, a popular deep learning architecture in image classification, proves to be robust in text classification as well. It utilizes layers with convolving filters that are applied to local features (LeCun et al., 1998).

To test the performance of CNN, with different embedding size, number of filters and kernel size, twelve combinations of parameters are experimented, and the model with the best performance is able to reach a test accuracy of 100% within 5 epochs.

4.6 Multimodal Approach

As is previously mentioned, since the dataset used for experimentation contains some numerical and categorical features in addition to textual features, it could be potentially useful to combine features of various types to make inferences, so that the classifier does not only depend on job descriptions but also information such as salary range, whether there is company logo, education requirements and so forth.

There are multiple ways for combining the features together. One popular approach is to use embeddings as feature extractors. For instance, BERT, as the state-of-the-art approach, could be used as a feature extractor by combining embeddings and other features and then fitting a classical machine learning model, or as a model for further fine-tuning (Tunstall et al. 2022). Here, to be consistent, we use TF-IDF vectors as the textual features and append additional useful features to make inferences.

The first step would be preprocessing and imputation for non-textual columns. For all categorical features including employment type, required education, required experience, location, department, industry, and function, missing values are replaced by 'Unspecified', because missing could also have information. There is another textual column named `benefits`, however, since many of the values are missing, here it is transformed into a binary feature, and 1 indicates there is benefit description in the job posting, and 0 otherwise. For required education, some lower-level categories are regrouped into one to reduce dimension. Lastly, there exists a salary range feature, in the form of 'lower bound – upper bound'. In this case, the lower bound and upper bound are both extracted, normalized and placed into two separate columns. The missing values and invalid values in each column are imputed by the medians, respectively.

For the multimodal approach, two classical machine learning models are tested: logistic regression, and SVM. Difference combinations of parameters for both models are experimented, together with different number of features (k) in the TF-IDF vector through chi-squared selection criteria. Some results are summarized blow.

Model	Parameters	Accuracy	F1
Logistic Regression	unigram, k=1000, penalty=elasticnet, C=1	0.92	0.92
SVM	unigram, k=1000, loss=squared hinge, alpha=1e-5	0.92	0.92

Table 3: Test Score for Multimodal Learning

The best multimodal approach using logistic regression performs slightly better than before, whereas the best multimodal model using SVM is slightly worse.

4.7 Imbalance Correction

Since the dataset is highly imbalanced, we took upsampling approach before fitting a model. However, this will result in a situation where probabilities predicted for fraudulent jobs will be much higher. However, in reality, the probabilities of fake job postings are very small. Therefore, imbalance correction using Bayes Classifier could be a good approach to correct the probability distributions.

The steps for imbalance correction are as follows:

1. Calculate $p = P(y = 1)$ for the entire original training set. In this case, $p = 0.05$
2. Balance the training data to have the desired fraction $p_s = 0.5$
3. Calculate the population odds $O = \frac{p}{1-p}$ and $O_s = \frac{p_s}{1-p_s}$ for the original and balanced samples, respectively
4. Fit your classification model $p_s(x) = P[y = 1|x]$ to the balanced data
5. Recover the corrected classification model by $p(x) = P[y = 1|x] = \frac{p_s(x)O}{O_s - p_s(x)(O_s - O)}$

Bayes classifier concepts provide a convenient way to correct for artificial balancing. This is not always necessary if one only wants scoring direction. In this paper, the probabilities generated are corrected through the above algorithm.

5 Result

The best models for each model architecture with its corresponding parameter setting are summarized in table 4.

Model	Paramsters	Accuracy
Logistic Regression	unigram, k=1000, penalty=elasticnet, C=1	0.90
Logistic Regression (Multimodal)	unigram, k=1000, penalty=elasticnet, C=1	0.92
SVM	unigram, k=1000, loss=squared hinge, alpha=1e-5	0.94
CNN	emb_size=64, num_filters=16, kernel_size=4	1.00

Table 4: Best Performing Models

6 Discussion

Based on the results, we noticed that all models are able to perform reasonably well, and among them, CNN performs best, with almost perfect test accuracy.

In this study, combing non-textual features does increase performance for logistic regression but not for SVM. However, further feature engineering and feature selection might be helpful when additional non-textual features are considered.

There is still a lot of room for improvement. First, upsampling is probably not the best approach for dealing with imbalanced data, especially because upsamping simply repeatedly draws observations from the same sample. Even though the word embeddings are numerical in nature, SMOTE is neither considered a good approach, as in high-dimensional settings, the various samples are almost uniformly distant from each other, negatively affecting the proper definition of neighborhood (Maldonado et al., 2019). There are already some research exploring this field, some scientists suggest that focal loss, which is a popular method in image classification, could be well applied to natural language processing. Another option is to text augmentation libraries like NLPAUG to generate synthetic data.

Since all word representations are TF-IDF vectors in this paper, some future steps could also include experimentations using word2vec embeddings and BERT embeddings, to give a more comprehensive comparison study.

References

C. Anita, P. Nagarajan, G. Sairam, P. Ganesh. 2021. Fake Job Detection and Analysis Using Machine Learning and Deep Learning Algorithms. Revista Gestao Inovacao e Technologias.

Y. Kim. 2014. Convolutional Neural Networks for Sentence Classification. arXiv preprint. arXiv: 1408.5882.

L. Tunstall, L. Werra, T. Wolf. 2022 (Expected). Natural Language Processing with Transformers. O'Reilly Media, Inc.

S. Bandyopadhyay, S. Dutta. 2020. Fake Job Recruitment Detection Using Machine Learning Approach. International Journal of Engineering Trends and Technology.

S. Maldonado, J. Lopez, C. Vairetti. 2019. An alternative SMOTE oversampling strategy for high-dimensional datasets. Applied Soft Computing Journal, 76 (2019) 380-389

Y. LeCun, L. Bottou, Y. Bengio, P. Haffner. 1998. Gradient-based learning applied to document recognition. In Proceedings of the IEEE, 86(11):2278– 2324, November.