CIS 520 Fall 2019, Project Report

Incorporating Machine Learning into Fake JobPosting Prediction

Team Members:

- Xi He; Email: xhe93@seas.upenn.edu
- Yihang Zhou; Email: yihangzh@seas.upenn.edu

Abstract

Considering the increasing number of online job posting scams, the incorporation of machine learning algorithms shows significance in fake jobPosting prediction. In this project, we aim to predict the authenticity (fake/real) of job postings based on text description. We implement natural language processing techniques (Countvectorizer and Tfidfvectorizer) to effectively victorize the words. Then, we construct multiple supervised machine learning classifiers and their results are compared to find out the best one for identifying fake Internet job ads. As a result, SVM with linear kernel model shows the highest f1 score–0.855 and is considered as the best.

1 Motivation

With the development of technology and the internet, companies and hiring managers tend to post job advertisements on the internet. Many people use the benefits of job board websites to spread fake job postings and then, collect the applicants' information for Robot call, sending spammy emails, or even criminals [1]. In USA, more than 3700 employment scam cases were reported in 2018, which was nearly a double of that number in 2017 [2]. Impressively, the case number rises at a significant speed in 2020, due to the Covid-19 pandemic [3]. As graduate students who are looking for summer internship opportunities right now, we are interested in building some classification machine learning models to help us to indicate the authenticity of job online postings based on jobs' description and some other information. Therefore, the goal of the project is to understand what some key aspects will lead to a fake/real job posting and construct some machine learning models to classify the authenticity of job posting. These machine learning models may help people who are looking for jobs to find out real job postings online, to save lots of time on writing job applications, and to avoid disclosure. Besides, big companies also benefit since some fake job postings represent these famous corporations to fraud job seekers, which draw significant loss to their credibility.

2 Related Work

Multiple related works have been down to detect email spam [4], review spam [5] and fake news [6]. These applications have some same features as the job scam tool introduced later in this paper–they are all related to natural languages and the target is whether yes or no.

Regarding to email spams, since these frauds can bring impressive financial damage to both individuals and companies, they are among one of the most annoying problems nowadays. Gmail, Yahoo, and Outlook all combined multiple machine learning classifiers–especially neural networks to detect email spams, based on the content, case, previous likeness, etc. [7].

In terms of review spams, due to customers' more reliance on reviews of certain products and services. Manufacturers, retailers, and service providers also pay attention to these online comments. However, this phenomenon gives wrongdoers chances to fake reviews to devalue or promote products and services, as a kind of unfair competition. Hence, supervised learning classifiers, like SVM, unsupervised classifiers, like semantic language model (SLM), and semi-supervised learning, like PU-Learning, are all considered recently [8].

Fake news detection, which is one of the most popular applications of machine learning, is widely analyzed. Thanks to the development of social media, the public have more access to online news, which also results

to an increase of fake news generated for injustice reasons–especially during the 2020 US election. Based on social context, news context and a great number of points of views, CNN, RNN and other algorithms are applied [9].

Job posting classification has also been tried before [10]. Key features are selected with some of them are extracted from Natural Language Processing (NLP) after preprocessing. Then, machine learning techniques like neural net and K-nearest Neighbor Classifier are applied and compared to choose the best one. This project will be held on a dataset from the University of the Aegean [11], analyzing the accuracies of multiple machine learning methods a bit different from previews works.

3 Data set

3.1 Initial Data Statistics and Visualization

This project is using a job posting information dataset from the Laboratory of Information Communication Systems Security, University of the Aegean [12].The dataset contains 17,880 real-life job ads with 17,014 legitimate and 866 fraudulent published between 2012 to 2014, mostly in the USA. Each sample is described by 17 features with one target–fraudulent or not. These 17 features consist of both textual information and meta-information about the jobs, including the title, location of the companies, required education, and the job description.

Before running machine learning codes, the detailed data inside is analyzed. Considering the 866 fake ads is less than 5% of the total, as shown in Figure 1, a large imbalance is unable to be ignored. An imbalanced dataset loses information in the process. Balancing methods need to be used in order to ensure the accuracy of minority class is not very low due to the imbalance.

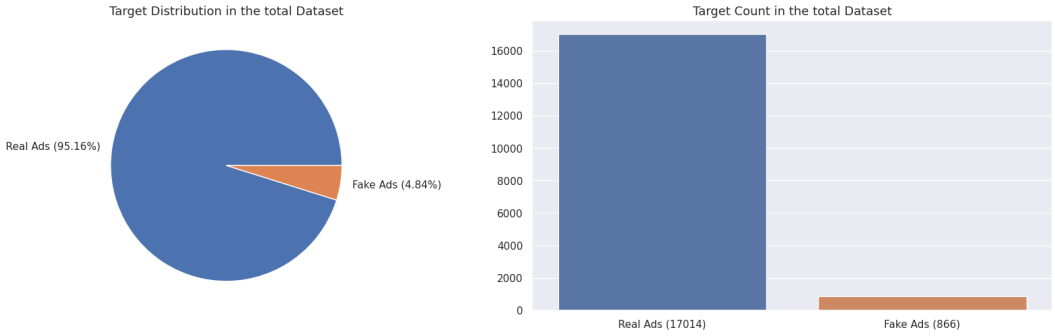

Figure 1: Sample imbalance in terms of the target

Another key factor is the existence of missing data, shown in Table 1. The missing values are replaced by blankets. Besides, multiple preprocesses are conducted, which are explained in the below Problem Formulation Section. Since some features do not have too much related to the prediction, they are deleted to simplify training. More specifically, the job_ID is not necessary since it just counts from 1 to 17880. The salary_range has too much missing data. Additionally, whether telecommuting, has the company logo, has questions, and has experienced are not important and deleted as of a result, which is proven by a trial taking these features into account but with negligible improvements in output results. In this project, the combination of the title, location, department, company_profile, description, requirements, benefits, employment_type, required_education, industry, and function is used for machine learning.

Column name	Number of missing
job_id	0
title	0
location	346
department	11547
salary_range	15012
company_profile	3308
description	1
requirements	2695
benefits	7210
telecommuting	0
has_company_logo	0
has_questions	0
employment_type	3471
required_experience	7050
required_education	8105
industry	4903
function	6455
fraudulent	0

Table 1: Number of missings in each column

The word cloud that includes the summary of different classes is shown in Figure 2 and Figure 3 from which we could easily summarize some words that seem to be critical in each class. For example, in real job posts, "bachelor degree" and "years experience" frequently occur. In the fake job posts, "high school" and "data entry" frequently occur. Something clear is that differences do exist but small. Both real and fake ads pay attention to customer service and experience. However, real jobs require a higher education level, more communication skills, focusing more on information technology. This is crucial in the later implementation of machine learning codes.

Figure 2: Most frequent words in real posts

Figure 3: Most frequent words in fake posts

3.2 Data Pre-processing

In this project, we only consider text variables in the job posting dataset. We combine those selected text variables, including the title,location and description listed in section 2.1 into a single text variable called content.

We perform the following pre-processing steps on our content variable.The work flow is briefly summarized as following:

- Fill missing text variable with an empty string
- Convert the words to lower cases
- Remove the stopping words in sentences
- Remove all punctuation

More details are shown in the Problem Formulation section.

4 Problem Formulation

4.1 Target Variable Y and Supervised Learning

We formulate a Fake Job Posting Prediction as a supervised binary classification problem. Y-target in this problem is the authenticity (fake/real) of job posting. The dataset contains 17014 real jobs and 866 fake jobs. In this project, 0 represents real job posts and 1 represents fake job posts.

4.2 Embedding and X Features

The original X feature in this project is the text (job description) of the job posts. For the non-deep learning models, text data points are vectorized by using two methods: CountVectorizer and TfidfVectorizer. For the deep learning model (BiDirectional LSTM), we first use the Tokenization method to split the text, then use the Word Embedding method to vectorize text data points to real numbers.

CountVectorizer method vectorizes the sentence according to the word frequency (count). One benefit would be: all sentences are having the same length and could be a good suit for the input format of machine learning models we will later use. It is a simple and straightforward embedding method.

TfidfVectorizer method vectorizes the sentence according to the words TF-IDF score. TF-IDF balances out the term frequency with its inverse document frequency. Therefore, one benefit would be: TF-IDF will give higher scores to the words that occur frequently in one text but occur rarely in other texts. TfidfVectorizer method will give higher scores to these words and thus they will be the ones that the model identifies as important and tries to learn.

Tokenization is a way of separating a piece of text into smaller units called tokens. Tokens can be either words, sub-words, or characters. In this project, we use words as tokens. Word Embedding method maps words to vectors of real numbers. Words that have the same meaning have a similar representation. We use it to vectorize x features and to reduce the dimensionality of word vector spaces for our deep learning model.

After embedding the text, the x features are word vectors. Based on the machine learning models, we use either TfidfVectorizer, CountVectorizer, or Word Embedding method to perform word vectorization.

4.3 Imbalanced Dataset

In this dataset, about 90 percent of the samples are real job posts. This dataset is highly imbalanced. Standard classifier algorithms, such as Decision Tree and Logistic Regression, has a bias toward classes. For example, machine learning classifiers tend to predict the majority class data when the dataset is highly imbalanced. To handle class imbalance, we use the Random Oversampling method to make the number of samples belonging to each class equal in the training set. The Random Over-sampling method randomly increases the number of samples from minority class. We perform the oversampling method after splitting the dataset into training and test set to avoid repeating samples in both training and test sets. We only perform the oversampling method on the training set to meet the real-life situation.

4.4 Evaluation

For measuring the accuracy of models, we will use F1 score since it clearly shows the summary of prediction outcome including the numbers of True Positives, True Negatives, False Positives and False Negatives under actual class and predicted class.

$$F_1 = \frac{2 \cdot Recall \cdot Precision}{Recall + Precision}$$

F1 score is a suitable method to measure the accuracy of an imbalanced dataset since it finds an equal balance between precision and recall which is useful for dealing with imbalanced datasets. Thus, we use F1-score as our evaluation metrics.

4.5 Loss Function

For this project, we will use the binary cross-entropy loss function for our deep learning model (Bidirectional LSTM) since this project is a binary classification problem and Bidirectional LSTM's output will be probability values between 0 and 1. Other lost functions will be depended on the model used.

5 Methods

The machine learning models used are Binary Logistic Regression, Decision Tree, Random Forest, SVM, and Bidirectional LSTM.

5.1 Baseline: Binary Logistic Regression

Logistic Regression is widely used in the binary classification method. It is a supervised machine learning model. It is a special class of generalized linear model[13]. The Binary Logistic Regression model is simple and very efficient to train. We use both TF-IDF and Count representation of x features to train the binary logistic regression model. We use it as a baseline for this project.

5.2 Baseline: Support Vector Machine

Support Vector Machine(SVM) is similar to the logistic regression, capable of solving classification problems. But SVM tries to find the best margin that separates classes, different from what logistic regression does. This makes SVM having less overfitting risk, and a 2% or 3% accuracy higher than logistic regression, but aslo apprantly, more time-consuming. In this program, sklearn.svm.SVC() is used, which is a specific application of SVM, based on libsvm. It is more suitable dealing with classification problems. TF-IDF and Count representation of x features are also applied in this SVC method.

5.3 Baseline: Decision Tree

Decision Tree is one of the most popular classifications and supervised learning models. It is easy to use and explains with simple math. It is very intuitive. We use both TF-IDF and Count representation of x features to train the Decision Tree model. We also perform hyperparameter-tunning and cross-validation by using the Random Search Method to find the best model. We use it as another baseline for this project. The Random Search method is used since it is great for getting hyperparameter combinations randomly.

5.4 Random Forest

Random forest [14] is a bagging method that combines a set of decision tree to make a final prediction using vote. This model is a more advanced decision tree model. This ensemble model lowers the risk of overfitting comparing with the Decision Tree model and runs efficiently on a large dataset. We use both TF-IDF and Count representation of x features to train the Random Forest model. We also perform hyperparameter-tunning by using Random Search and cross-validation to find the hyperparameters that give the highest F1-score.

5.5 BiDirectional Long Short-Term Memory Networks

Bidirectional Long Short-Term Memory Networks (BLSTM) is a special recurrent neural network which works well in sequence classification problems with feedback connections. In this deep learning method, the Tokenization method is applied to split the text with then, the Word Embedding method to vectorize it. Hyperparameter tuning with validation test and early stopping are used to build the best model.

6 Experiments and Results

In the experiment, the training set is 70 percent of the original dataset and the test set split is 30 percent of the original dataset. We use the oversampling method to balance the training set. As a result, there are 11910 training examples in each class (total of 23820 training samples in the training set) and 5364 test samples. We use F1-score as an evaluation metric to measure the accuracy of the models.

6.1 Binary Logistic Regression

We train the binary logistic regression model with both TF-IDF and Count representation of x features. The performance metric is F1-score.

Model	Train F1-score	Test F1-score
Logistic Regression with Count	0.999	0.793
Logistic Regression with TF-IDF	0.996	0.814

Table 2: The F1 score of logistic regression models

The detailed classification report on the test set using count and TF-IDF representation is shown in Figure 4 and 5. One interesting fact is that F1-score for class 0 (real job posts) is much higher than the F1-score for class 1 (fake job posts).

	precision	recall	f1-score	support
0	0.99	0.99	0.99	5104
1	0.78	0.81	0.79	260
accuracy			0.98	5364
macro avg	0.88	0.90	0.89	5364
weighted avg	0.98	0.98	0.98	5364

Figure 4: Classification Report on the test set(Count)

	precision	recall	f1-score	support
0	0.99	0.99	0.99	5104
1	0.76	0.88	0.81	260
accuracy			0.98	5364
macro avg	0.88	0.93	0.90	5364
weighted avg	0.98	0.98	0.98	5364

Figure 5: Classification Report on the test set(TF-IDF)

6.2 Support Vector Machine

We train the Support Vector Machine model, based on sklearn.svm.SVC(), with both TF-IDF and Count representation of x features.The performance metric is F1-score.

The gridsearch is not considered because SVMs usually take a long looping time. The simple SVC is st with linear kernel.

Model	Train F1-score	Test F1-score
SVC with Count	1.000	0.782
SVC with TF-IDF	1.000	0.855

Table 3: The F1 score of SVC models

It is clear that SVC with TF-IDF shows a higher F1 score comparing to the SVC with Count. Besides, the SVC with TF-IDF is also better than the model using Logistic Regression and TF-IDF, even gridsearch is not used. This proves that SVM is truly better than logistic regression considering the F1 score.

6.3 Decision Tree

We train the Decision Tree model with both TF-IDF and Count representation of x features. We perform hyperparameter-tunning on max-depth, min-samples-split, and criterion with the Random Search and the 5-fold cross-validation method. Figure 6 shows the result of hyperparameter tuning based on CountVectorizer. Using 2 for min-sample-split, None for max-depth, and entropy for criterion give the best model.

```
Tuned Decision Tree Parameters: {'min_samples_split': 2, 'max_depth': None, 'criterion': 'entropy'}
Best score is 0.985928535357958
```

Figure 6: Random Search Result based on CountVectorizer

Figure 7 shows the result of hyperparameter tuning based on TfidfVectorizer. Using 2 for min-sample-split, None for max-depth, and entropy for criterion give the highest f1-score on the validation sets.

```
Tuned Decision Tree Parameters: {'min_samples_split': 2, 'max_depth': None, 'criterion': 'entropy'}
Best score is 0.985928535357958
```

Figure 7: Random Search Result based on CountVectorizer

The f1-score on training and test sets are shown in the table4. Based on the result shown in table 4, using the TF-IDF representation has a better F1-score than Count representation does.

Model	Train F1-score	Test F1-score
(Count) Decision Tree	1.0	0.611
(TF-IDF) Decision Tree	1.0	0.634

Table 4: Decision Tree Result

6.4 Random Forest

We train the Random Forest model with both TF-IDF and Count representation of x features. We perform hyperparameter-tunning n-estimators, max-depth, min-samples-split, and min-samples-leaf. 8 shows the result of hyperparameter tuning based on CountVectorizer. Using 2 for min-sample-split, 602 for n-estimators, 200 for max-depth, and 1 for min-samples-leaf give the best model.

```
Average of the best f1-score in various folds during cross validation =  0.9988260835016227
The best parameters found during k-fold cross validation is =  {'n_estimators': 602, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_depth': 200}
```

Figure 8: Random Search Result based on CountVectorizer

9 shows the result of hyperparameter tuning based on TfidfVectorizer. Using 3 for min-sample-split, 602 for n-estimators, None for max-depth, and 1 for min-samples-leaf give the best model.

```
Average of the best f1-score in various folds during cross validation =  0.9988260835016227
The best parameters found during k-fold cross validation is =  {'n_estimators': 602, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_depth': 200}
```

Figure 9: Random Search Result based on TfidfVectorizer

The f1-score on training and test sets are shown in the table5. Based on the result shown in table 5, using the Count representation has a better F1-score than TF-IDF representation does.

Model	Train F1-score	Test F1-score
(Count) Random Forest	1.0	0.771
(TF-IDF) Random Forest	1.0	0.750

Table 5: Random Forest Result

6.5 BiDirectional Long Short-Term Memory Networks

Tokenization rather than TF-IDF and Count representation of x features is applied in this deep learning algorithm. Besides, the train-test-split process and the random-over-sample process remain same. The model is constructed sequentially with the embedding layer at first, then the SpatialDropout layer to reduce overfitting. The BLSTM layer comes next with two dense layers following by(Relu and Sigmoid serving as activation functions). The loss function selected is binary cross entropy and the metric used is accuracy instead of f1 score. In the model fitting process, 10 epochs, batch size 64 and 20% data for validation are used.

After 4 epochs(with early stopping–the monitor is set as the validation loss and the patience is 2), the f1 score in testing is 0.771. It is surprising that Bidirectional LSTM has a relative low f1 score comparing with the Logistic Regression.

Model	Train F1-score	Test F1-score
BiDirectional LSTM	1.0	0.772

Table 6: BiDirectional LSTM Result

The classification report for the test set are shown below.

	precision	recall	f1-score	support
0	0.98	1.00	0.99	5104
1	0.88	0.69	0.77	260
accuracy			0.98	5364
macro avg	0.93	0.84	0.88	5364
weighted avg	0.98	0.98	0.98	5364

Figure 10: BiDirectional LSTM Classification Report

7 Conclusion and Discussion

In this project, we explore the best model to detect fake online job postings using a dataset from the University of the Aegean. The implementations of several pre-processing steps, including three different word conversion methods, the stopping-word-removal, and the punctuation-removal are done at first. Then, we try three baseline machine learning algorithms–the logistic regression, the support vector machine, and the decision tree. Further, we construct two ensembled complex models using random forest and bidirectional long short-term memory respectively trying to improve the f1 scores of models.

Based on our results, we can see it quite clear that the SVM with linear kernel and TF-IDF representation shows the highest f1 score–0.855, comparing to the binary logistic regression and the decision tree models (the three single classifiers). It proves the robustness of SVM in classification problems with small detaset. In this project, the x number is even smaller than 18000. Besides, during the training process, the dataset has already been split into two parts with nearly 12000 x (each with 17 features) used for model training, making the dataset even smaller. Additionally, TF-IDF representation is better than the countVectorizer method in all models (except the BLSTM one using toknization).

Impressively, SVM even outperforms ensembeled models–the random forest model and the BLSTM model in terms of the f1 score (although the two complex models both show significant high accuracies). Since the f1 score which implies how less false positives and false negatives exist, it is a more suitable score for this imbalanced project when the accuracy actually cannot be trusted to select a well-performing model. But frankly speaking, ensembeled models will perform better when the models are well-constructed with better hyperparameters. One evidence is that random forest models all have higher f1 scores than these of decision tree models. Besides, BLSTM model should perform better since they are widely applied in natural language tasks.

We also print out the detailed classification report, such as shown in Figure 4 and 5, our machine learning models have a much higher f1-score on predicting real job posts than fake job posts. Low f1-scores of fake job posts prediction lower the overall f1-score of our machine learning models. One possible explanation for this is that our original dataset is highly imbalanced and we only have 866 fake job posts in the original dataset. Even though we use oversampling method to randomly increase the number of fake job posts in the training set, there are many same fake job posts in the training set and there is not a lot of variety among fake job posts. As a result, our machine learning models did not perform well on predicting fake job posts.

In this project, due to the unforeseen project memberâs health issues and the corresponding lack of time, more analysis is only considered without implementation. The application of gridsearch in SVM, or not linear kernel but Gaussian kernel can be used and compared. Further, a better constructed random forest, BLSTM, and Xgboost can be applied to increase model f1 score. Some other popular NLP machine learning algorithms, such as Bayesian Networks, should be considered in the future work. We also believe the dataset is relatively too small and too imbalanced for a robust model. The better dataset is necessary for further development.

In general, SVM can be seen as the best model for this fake job detection scheme. But due to its potential risk facing a large dataset. The complex ensembled algorithm can be better choices. Employment scams exist nowadays and will exist in the future continuously troubling job seekers. The SVM model introduced in this project with an f1 score of 0.855 can be a good tool for them to detect fake job postings.

Acknowledgments

We want to show great thanks to Dr. Ungar and TAs for their kind help and understandings during this project.

References

[1] ANDREW G. ROSEN. The dirty truth: Why employers post fake jobs. https://www.jobacle.com/blog/the-dirty-truth-why-employers-post-fake-jobs.html. Accessed: 2020-11-15.

[2] Sarah OâBrien. Employment scams are on the rise in tight labor marketemployment scams are on the rise in tight labor market. https://www.cnbc.com/2018/11/08/employment-scams-are-on-the-rise-in-tight-labor-market.html, November 2018. Accessed: 2020-11-27.

[3] Carmen Reinicke. Job scams have increased as covid-19 put millions of americans out of work. hereâs how to avoid one. https://www.cnbc.com/2020/10/06/job-scams-have-increased-during-the-covid-19-crisis-how-to-one.html, October 2020. Accessed: 2020-11-27.

[4] Ismaila Idris, Ali Selamat, Ngoc Thanh Nguyen, Sigeru Omatu, Ondrej Krejcar, Kamil Kuca, and Marek Penhaker. A combined negative selection algorithm–particle swarm optimization for an email spam detection system. *Engineering Applications of Artificial Intelligence*, 39:33–44, 2015.

[5] Ee-Peng Lim, Viet-An Nguyen, Nitin Jindal, Bing Liu, and Hady Wirawan Lauw. Detecting product review spammers using rating behaviors. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 939–948, 2010.

[6] Samir Bandyopadhyay and SHAWNI DUTTA. Analysis of fake news in social medias for four months during lockdown in covid-19. 2020.

[7] Emmanuel Gbenga Dada, Joseph Stephen Bassi, Haruna Chiroma, Adebayo Olusola Adetunmbi, Opeyemi Emmanuel Ajibuwa, et al. Machine learning for email spam filtering: review, approaches and open research problems. *Heliyon*, 5(6):e01802, 2019.

[8] Michael Crawford, Taghi M Khoshgoftaar, Joseph D Prusa, Aaron N Richter, and Hamzah Al Najada. Survey of review spam detection using machine learning techniques. *Journal of Big Data*, 2(1):23, 2015.

[9] Kai Shu, Amy Sliva, Suhang Wang, Jiliang Tang, and Huan Liu. Fake news detection on social media: A data mining perspective. *ACM SIGKDD explorations newsletter*, 19(1):22–36, 2017.

[10] Shawni Dutta and Samir Kumar Bandyopadhyay. Fake job recruitment detection using machine learning approach.

[11] Employment scam aegean dataset. http://emscad.samos.aegean.gr/. Accessed: 2020-11-15.

[12] Employment scam aegean dataset. http://emscad.samos.aegean.gr/, November 2018. Accessed: 2020-11-27.

[13] Peter McCullagh. *Generalized linear models*. Routledge, 2018.

[14] Vladimir Svetnik, Andy Liaw, Christopher Tong, J Christopher Culberson, Robert P Sheridan, and Bradley P Feuston. Random forest: a classification and regression tool for compound classification and qsar modeling. *Journal of chemical information and computer sciences*, 43(6):1947–1958, 2003.