Vasudha Jasthi, Gaurav Shinde
4/28/21

Fraudulent Job Posting Classifier and Job Posting Clustering

Introduction:

With the pandemic and the advent and ease of technology, many more companies today than ever are posting their job vacancies online on various websites. Even though this is very convenient for companies, this poses a risk of job scams and fraudulent job postings known as Online Recruitment Fraud (ORF). This is a new challenge to both companies and job-seekers because it leads to stolen money, and job-seekers having false hopes in getting employed. According to the Better Business Bureau, "Employment scams were the #1 riskiest scam in both 2018 and 2019… An estimated 14 million people are exposed to employment scams with more than $2 billion lost per year, not counting time or emotional losses. The risk of this scam continues to rise…".

For every one of us, finding a job is an unavoidable task. According to TopResume, the average job-hunting time in the United States takes about 5 months. 5 months is a long time when thinking in terms of finances and daily life for instance. The general steps of job finding includes sifting through numerous websites with job ads, reading through each job description and features, applying to the jobs, etc. Overall, the entire process is a tiring and a laborious one, especially to read every description in order to figure out if it matches the job seekers skills and wants.

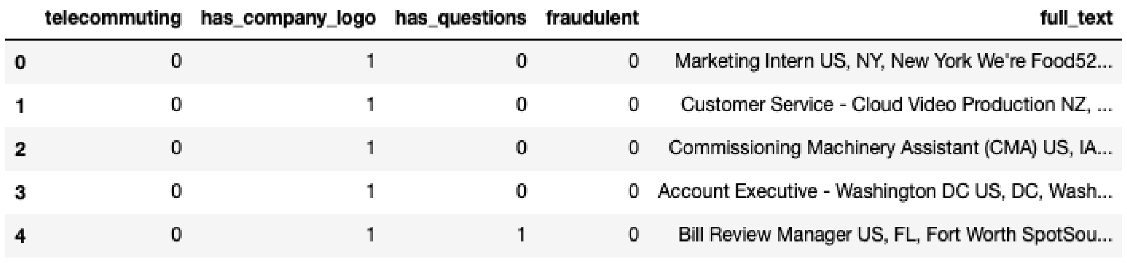
This project seeks to help these problems by building a model to determine whether a job ad posting is fraudulent, based on recent job ad postings, and creating a model to identify the most similar job descriptions. The "Fraudulent Job Posting Classifier" determines whether a particular job ad is fake or not using features of a job posting such as the description of the ad. In addition, we at first wanted to create and compare search-engine for finding the best job for a candidate based on resume and job description match using contextual embedding models of BERT and ELMO. However, such proved much difficult to build because of library errors (gensim library methods and amazon s3 bucket links in particular) and model importing in google colab and local jupyter notebook. Instead, we chose to do clustering analysis of job descriptions for categorization. We used Kmeans, Agglomerative, Nearest Neighbor, and Fully Connected unsupervised learning model to visualize clustering of description word vectors, evaluated their performance, and gained information from cluster groups about job categorization.

Approach:

Before creating our models, we will perform various Exploratory Data Analysis methods and techniques so we can understand the data better. We will explore what are the most common words used in the postings as well as any high frequency groups among features fraudulent vs non fraudulent.

We will be using the Kaggle dataset Fake JobPostings which has information about various jobs including descriptions and whether they are fraudulent or not. This data set had 17,880 number of job posts. We will use this dataset for all of our analyses and models in the project. Before we start with building models, since the dataset is unbalanced, we will need to use imblearn for oversampling techniques. We will also then use data preprocessing methods for any missing values, stopword and unnecessary characters removal, and feature vectorizing (ex. Tf_idf vectorization). For the classification of fake job postings, we will be splitting the dataset into training, validation, testing sets to create the model. We plan to use an ensemble model of logistic regression, k-nearest neighbor, random forest classifier, and multinomial naive bayes models. GridSearchCV will also be used to find the best hyperparameters and we will use f1 score, accuracy, precision, and recall metrics to compare models and choose the best ones. We will finally choose the top 3 models from the models created and construct a Voting Classifier model which is an ensemble model technique. Included below is a step by step road map to create the classification model:

1. Preprocessing:
    a. Missing Values: removed any features that contained 50% or more NaNs.
    b. Any other missing text feature rows, replace NaN with ‘’ (empty character).
    c. Combined all the Text columns into a new feature.
    d. Dropped the combined features.



	telecommuting	has_company_logo	has_questions	fraudulent	full_text
0	0	1	0	0	Marketing Intern US, NY, New York We're Food52...
1	0	1	0	0	Customer Service - Cloud Video Production NZ, ...
2	0	1	0	0	Commissioning Machinery Assistant (CMA) US, IA...
3	0	1	0	0	Account Executive - Washington DC US, DC, Wash...
4	0	1	1	0	Bill Review Manager US, FL, Fort Worth SpotSou...

*This is how the data looks after combining the text column

    e. Removed punctuation marks, brackets html, urls, and any other unnecessary characters in the full_text column.
    f. Tokenize the full_text column.
    g. Apply stemming to each word in the column.
    h. Remove stop words .
2. Apply TFIDF Vectorizer and Countvectorizer (Bag of Words) and combine the vectors into 2 different datasets (df_tfidf, df_cv).
3. Balance both the datasets with oversampling method SMOTE.
4. Train and split the datasets using the 80:20 rule.
5. Create the various models and computer confusion matrix with target = fraudulent feature and the x = all other columns:
    a. Logistic Regression - tfidf
    b. Logistics Regression - BOW
    c. KNN - tfidf
    d. KNN - BOW
    e. MLP - tfidf
    f. MLP - BOW
    g. Random Forest Classifier - tfidf

        h. Random Forest Classifier - BOW

6. Choose the Top3 performing models from the 8 created above and run a Voting Classifier model (both hard and soft and for each tfidf and bow to compare)

This same preprocessing of data method was used for the Clustering models comparison as well as for generating text similarity using cosine similarity. One subtle difference was that data imbalance with SMOTE oversampling method was unnecessary since the target values were not binary and rather categorical in nature. The train-test-split was also not necessary because we wanted to analyze the categorization patterns of models rather than evaluate their predictive performance here. Also, the function feature was label encoded and conserved to use as target value for evaluation.

To find the best number of k clusters for tfidf was by using kElbowVisualizer. This uses Within-Cluster-Sum-of-Square errors in order to show its diminishing value over the course of increasing k value. Although the elbow curve is meant to look like an exponential dip followed by a flat line, in this case since we used upto 200 features that represented various words from the vectorization of job descriptions, so it instead shows an increasing and then decreasing trend. However, this is perhaps due to the number of distortions in clustering throwing the measure off course due to randomness of data aggregation. But fortunately, since we can still see a trend for the elbow curve and the trend for time taken, we can use their points of intersections to have the perfect diversity of representing the data values in the features as k. The ideal k for tfidf vectorized dataframe was 38, similar to the number of unique values in the function data feature. When tried with count vectorizer, the k value was predicted at a much lower value of 25. However since this was nearly half of the number of unique values (45) in the target function attribute, we chose not to evaluate it since it would not have been representative of the data. In the future however, we might plan to do this to perhaps see if evaluation scores differ. Coming closer to the clustering step, normalization of features and Principal Component Analysis dimensionality reduction to 2 was also later performed.

We will use K means clustering, Agglomerative clustering, Spectral Clustering with Nearest Neighbor, and Spectral Clustering with Fully-connected. These will then be evaluated on external evaluation and internal performance measure and giving remarks on observed differences between clustering.

Results:

We created many graphs during the exploring the data part. In Figure 1. we can see that the data is imbalanced as there are significantly more non-fraudulent cases than fraudulent cases. This is also most likely just a natural phenomenon. According to Figure 2, the largest % of fraud jobs fall in the full-time category. Figure 3 shows that the most amount of fraud jobs called for a certification. Figure 4 shows the % of fraud jobs by telecommuting. We hypothesize that telecommuting convenience may make employees overlook other aspects about the job posting, which may be a technique used by fraud job postings. We hypothesize that function may make employees overlook other aspects about the job posting, which may be a technique used by fraud job postings. We constructed a few more graphs, but we also created a correlation heat map matrix, as shown in Figure 5, between all of the features with the target feature which is the 'fraudulent' column. Even though not many of the features are highly correlated with the target, we will still include it in our models to see how they will affect them. We also created word clouds to see what are the most common words among fraudulent job postings and non-fraudulent job postings, as shown in Figure 6 and 7. In both, the phrase full time appeared frequently.

Words such as bachelor's degree, and others relating to having more experience appeared more in non-fraudulent; words like highschool and product/customer shows up more in fraudulent postings that could be designed for entry level job luring.

For the Classification Model, we created 8 models plus 4 Voting classifier models with the top 3 models/the 8 created. Here is a table summarizing the metrics:

Model	Accuracy	Precision	Recall	F1 Score
Log-Reg TFIDF	92%	90%	93%	92%
Log-Reg BOW	95%	94%	96%	95%
KNN (k = 2) TFIDF	87%	88%	85%	87%
KNN (k = 2) BOW	91%	88%	96%	92%
MLP TFIDF	51%	50%	100%	67%
MLP BOW	68%	68%	68%	68%
Random Forest Classifier - TFIDF	99%	99%	99%	99%
Random Forest Classifier - BOW	79%	100%	60%	75%

From this table, we can see that the Random Forest Classifier with tfidf performed the best with accuracy 99% out of all the eight models created and the least performing is the MLP tfidf model with accuracy about 51%.

In this table below, we display the metrics for the Voting Classifiers that we ran when combining Logistic Regression, K-Nearest Neighbors, and Random Forest Classifier Models.

Model	Accuracy	Precision	Recall	F1 Score
Voting Classifier - TFIDF (hard)	98%	98%	98%	98%
Voting Classifier - TFIDF (soft)	97%	97%	98%	97%
Voting Classifier - BOW (hard)	98%	98%	99%	99%
Voting Classifier - BOW (soft)	98%	97%	100%	98%

Overall, all four of the classification models did really well with each having at least a 97% accuracy.

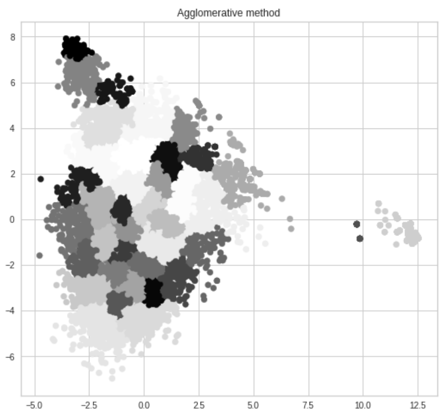
For our Clustering models, we compared 7 clustering methods through 3 external and 3 internal evaluation metrics in the table below:

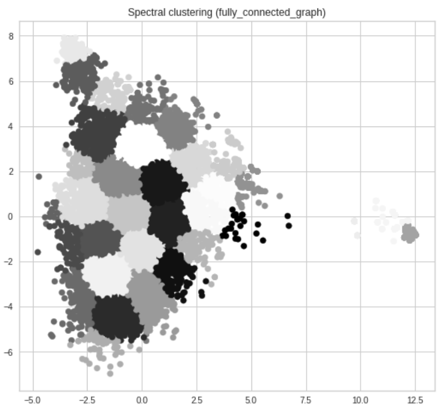
Clustering Model	NMI	Accuracy	Random Score	Dunn's index	Silhouette's Index	Davie Bouldin's Index
K Means	0.2286	0.1069	0.0171	0.0006	0.3526	0.7971
Agglomerative (ward link)	0.2303	0.1058	0.0164	0.0063	0.2951	0.8895
Agglomerative (single link)	0.0227	0.3406	-0.0165	0.0249	-0.1703	0.4597
Agglomerative (average link)	0.2182	0.1512	0.0158	0.0084	0.2562	0.7267
Agglomerative (complete link)	0.2212	0.1375	0.0165	0.0061	0.2656	0.8643
Spectral Clustering (nearest neighbor)	0.2207	0.1112	0.017	1.11	0.291	1.3916
Spectral Clustering (fully connected)	0.2173	0.1432	0.016	0.0007	0.3111	0.7379

Unsupervised learning and prediction model quality is judged by external validation that is not exactly the best for clustering, but here an NMI close to 1, accuracy close to 1, and random score that is positive and closer to 1 mean that they are the most representative clusters. Internal performance measure is best for judging the quality of clusters in terms of many factors like connectivity, member variance, compactness, grouping accuracy, etc. Here a dunn's index closer to 1, silloheutte's index positive and closter to 1, and david bouldin's index closer to 0 indicate best clustering.

From the results, although we could mention many statistics, average link agglomerative clustering had the greatest mix of good external and internal evaluation scores, which makes it good to represent job categorization based on important word tokenized sentences through tfidf. No other clustering model did significantly worse, however if you check notebook, you will see single link agglomerative clustering had very inaccurate grouping and inaccurate cluster representation. Which explains why it's accuracy and dunn's index score was so high. We would not use single link agglomerative clustering in this case for future with this dataset, at least if such a high k number is used.

Additionally, we saw a few distinctions within the clustering itself between models:





Here the agglomerative clustering that uses ward linkage that minimizes variance between clusters being merged, results in clusters that are much smaller and concentrated. Spectral clustering with fully connected graph shows larger and more loosely connected clusters. This visualization tells us that using different clustering algorithms can help group important words from job descriptions in different ways to test and build an optimum search engine for job suggestion by category. For example, the tight clustering means unsupervised learning is more segmentation when compared to loose clustering that tries to put words into a bigger pool. So someone wanting a selective search can use an algorithm integrating agglomerative ward analysis over a fully connected spectral clustering based algorithm. But either graph shows clusters being closely connected, which is not very trustworthy for categorization with confidence.

In terms of generalized cluster visualization analysis, we noticed that the clusters were very compactly packed together which indicates that the words themselves were connected. Due to having so many words in one description, it was hard to separate categories distinctly without using contextual embedding for instance and just relying on word similarity. Perhaps this could be aided by stricter word preprocessing measures where only important and industry unique terms are kept from descriptions. In addition, individual clusters themselves varied in shape and size which indicates that there is diversity in job function categorization between clustering algorithms.

We believe that although clustering did not show as much promise with categorization as we hoped, by further improving sentence preprocessing for selecting industry distinct words only, testing with other word vectorizer methods like Count Vectorizer, and performing more evaluation metrics including performance time, there could be more insight through clustering that could be revealed. We will continue to learn about how best to implement predictive clustering in the future towards the goal of helping users find the right jobs for them.

Conclusion:

The main goal of this project was to identify key traits of job descriptions which are fraudulent in nature, to create a classification model that detects fraudulent job postings and to cluster jobs based on descriptions.
For the classification model, we preprocessed the data, balanced the data using SMOTE, ran Logistic regression, K-Nearest Neighbors, MultiLayer Perceptron, and Random Forest Classifier models for each tfidf vectorizer and countvectorizer. For each model, we performed GridSearchCV to choose the best

parameters possible for each model. We chose the top 3 models and then created a Voting Classifier model (the top 3 were logistic regression, K-Nearest Neighbors, and Random Forest Classifier). All the Voting Classifier models we constructed gave us an accuracy of at least 97%.
Although our second goal of providing users with a measure to find desired jobs based on attributes did not show expected results, our analysis of the descriptions through different clustering methods has helped us learn about how it could be better implemented in the future towards job recommendation or similarity ranking programs. We used 7 clustering models, 4 of which are model variations, and we computed 3 external and 3 internal evaluation statistics. Although clustering using tfidf vectorization of descriptions proved low efficacy through generalized results in providing accurate information about job categorization, it did help us learn about more innovative methods to evaluate and improve job categorization.

Acknowledgements:

- https://philpapers.org/archive/NASMLA-3.pdf
- https://dorsam7.medium.com/using-word-embedding-to-build-a-job-search-engine-a43a5d443009
- https://www.researchgate.net/publication/318814319_Skill2vec_Machine_Learning_Approaches_for_Determining_the_Relevant_Skill_from_Job_Description
- https://www.bbb.org/globalassets/local-bbbs/council-113/media/bbb-institute/employmentscams/2020-bbb-employmentscams-report.pdf
- https://www.kaggle.com/shivamb/real-or-fake-fake-jobposting-prediction/code
- sklearn.ensemble.VotingClassifier — scikit-learn 0.24.2 ...http://scikit-learn.org › stable › modules › generated › s…
- SMOTE for Imbalanced Classification with Pythonhttps://machinelearningmastery.com › Blog
- Python | Stemming words with NLTK - GeeksforGeekshttps://www.geeksforgeeks.org › python-stemming-wor…
- https://blog.paperspace.com/implementing-levenshtein-distance-word-autocomplete-autocorrect/#:~:text=The%20Levenshtein%20distance%20is%20a,transform%20one%20word%20into%20another.
- https://towardsdatascience.com/cheat-sheet-to-implementing-7-methods-for-selecting-optimal-number-of-clusters-in-python-898241e1d6ad
- https://smorbieu.gitlab.io/accuracy-from-classification-to-clustering-evaluation/#:~:text=Computing%20accuracy%20for%20clustering%20can,the%20accuracy%20for%20clustering%20results.
- https://link.springer.com/article/10.1007/s40595-016-0086-9
- https://validclust.readthedocs.io/en/latest/validclust.html
- https://dzone.com/articles/kmeans-silhouette-score-explained-with-python-exam
- https://people.csail.mit.edu/dsontag/courses/ml14/notes/Luxburg07_tutorial_spectral_clustering.pdf

Figures from Exploratory Data Analysis explained earlier in Results:

Figure 1:

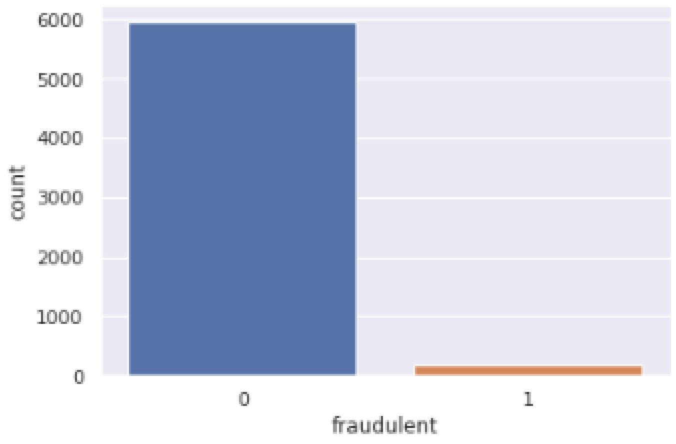


Figure 2:

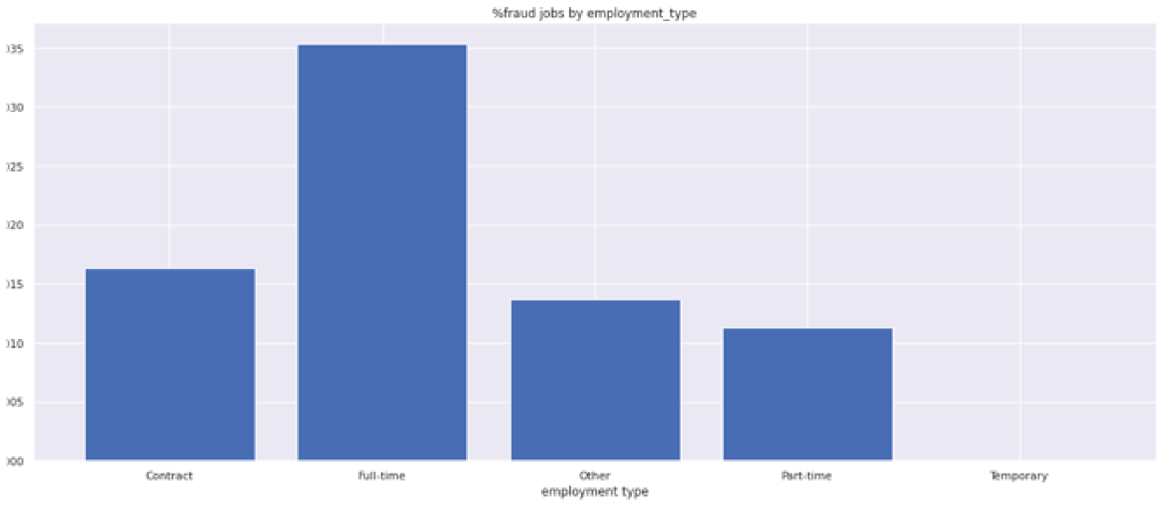
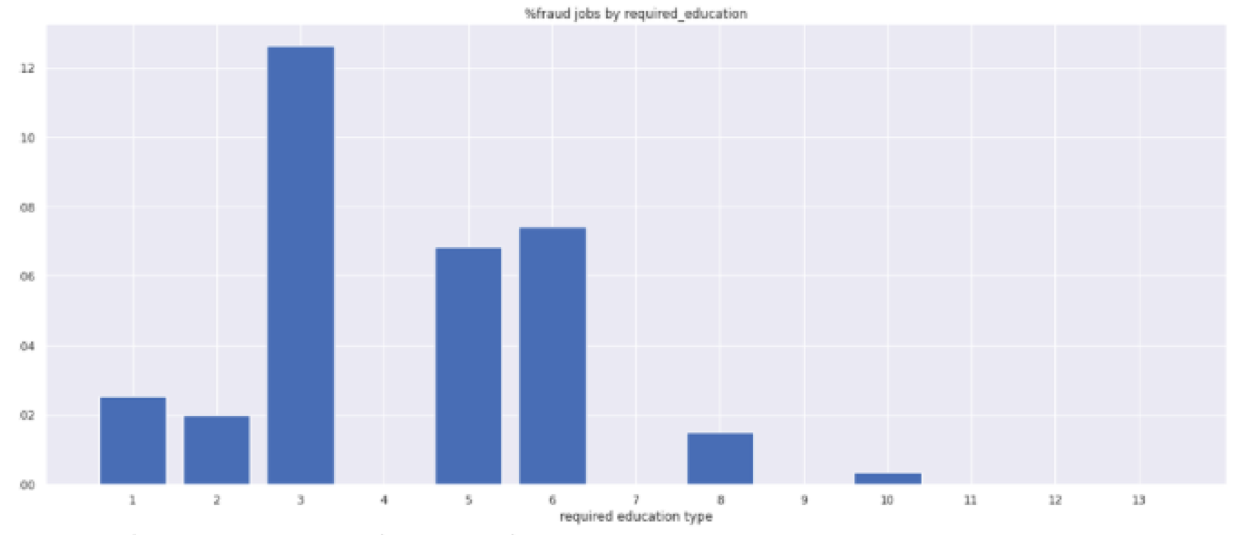


Figure 3:



* index 1: Associate Degree, index 2: Bachelor's Degree, index 3: Certification, index 4: Doctorate, index 5: High School or equivalent, index 6: Master's Degree, index 7: Professional, index 8: Some College Coursework Completed, index 9: Some High School Coursework, index 10: Unspecified, index 11: Vocational, index 12: Vocational - Degree, index 13: Vocational - HS Diploma,

Figure 4:

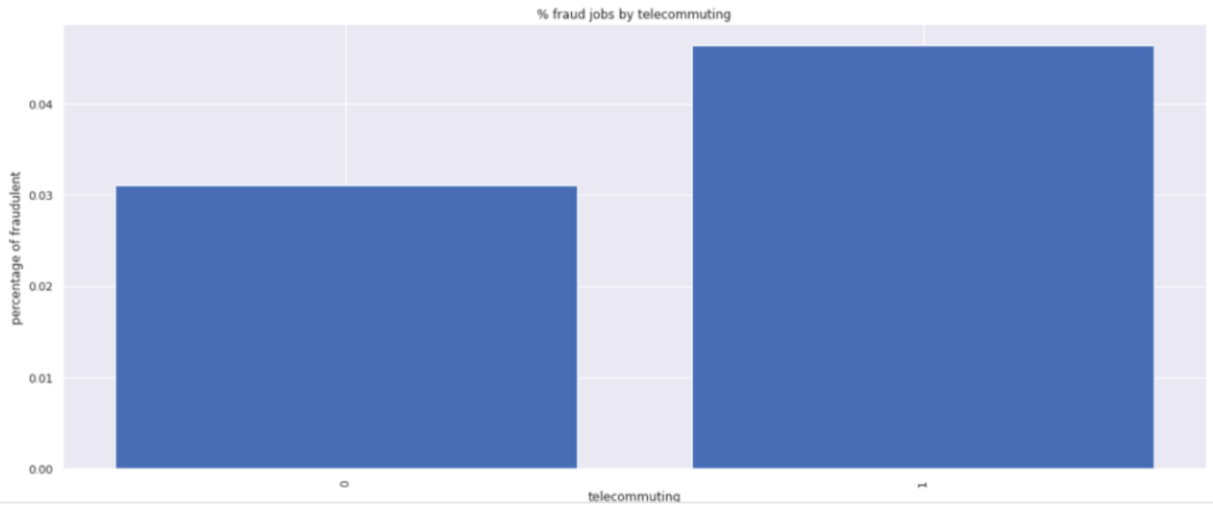


Figure 5:

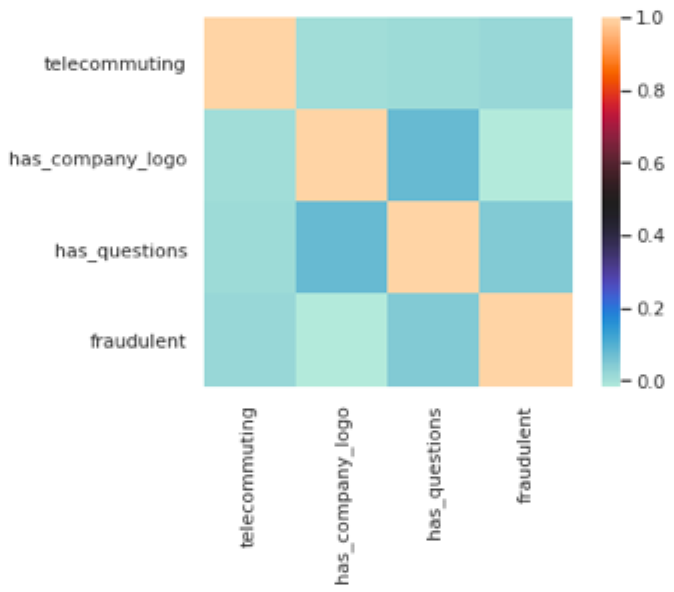


Figure 6. Non-Fraudulent Jobs

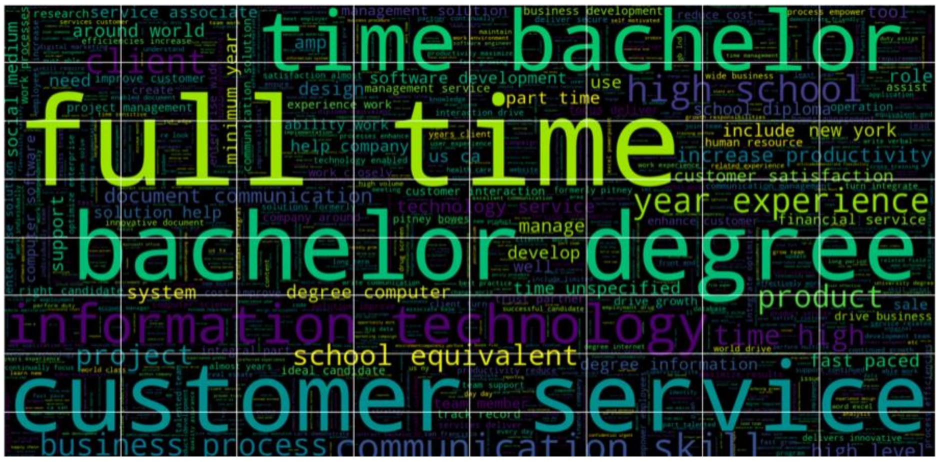
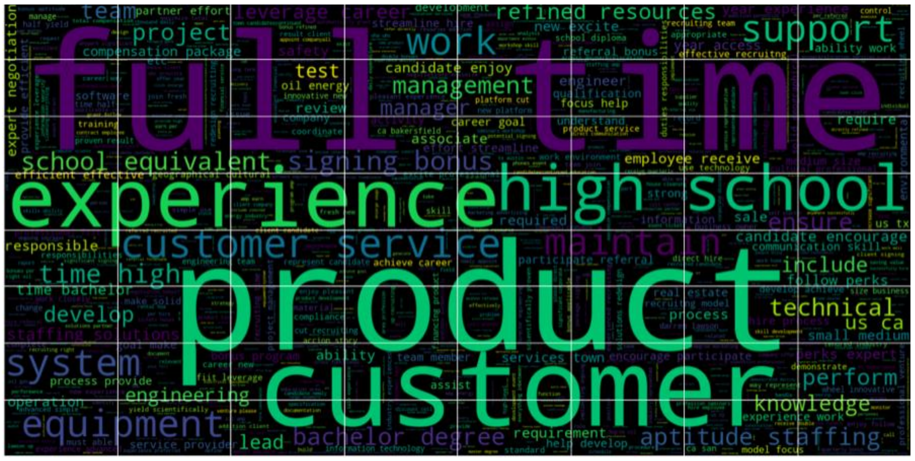


Figure 7. Fraudulent Jobs


full time
product
customer
experience
high school
customer service
support
work
management
equipment
system
bachelor degree
school equivalent