STAT 154 - Spring 2022

Group 3 Final Project Report

Yunjae Cho
Han-Yuan Hsu
Arlina Shen
Wei-Hsiang Sun

Contents

1 Description 3

2 Feature Creation 4

3 Unsupervised Feature Filtering 5

4 Power Feature Creation 5

5 Combined Feature Matrix 5

6 Classification on the Feature Matrix 5

7 Verification and Classification Results 8

8 Unsupervised Clustering of Feature Matrix 9

9 Improvements 10

10 Final Model and Results 12

1 Description

This project report concerns the detection of fraudulent job listings using available data, including fraudulent labels, a company's profile, job description, and many other characteristics. The report consists of several components: exploratory data analysis, feature engineering, unsupervised and supervised learning, model building & evaluation, and a discussion for potential improvement.

The data is expressed in a matrix form with 5362 rows corresponding to different job listings and 18 columns for the associate features of a job, where the last column is a 0-1 indicator with 1 indicating "fraudulent" and 0 indicating otherwise and the other 17 columns contain both textual, numerical, and categorical features. For example, features "company_profile" and "description" are textual, with some sentences in English describing a company's profile and job description. On the other hand, features "has_company_logo" (1/0) and "employment type" (Full-time, Part-time, Contract, etc.) are categorical, while the feature "salary" is numerical. Moreover, some entries of the data are labeled as "NaN", indicating that the values are missing. Unfortunately, the information concerning the cause of these missing values is unavailable. Next, we display the histograms of two categorical variables such as "required_experience" and "employment_type" to reveal the data's characteristics.

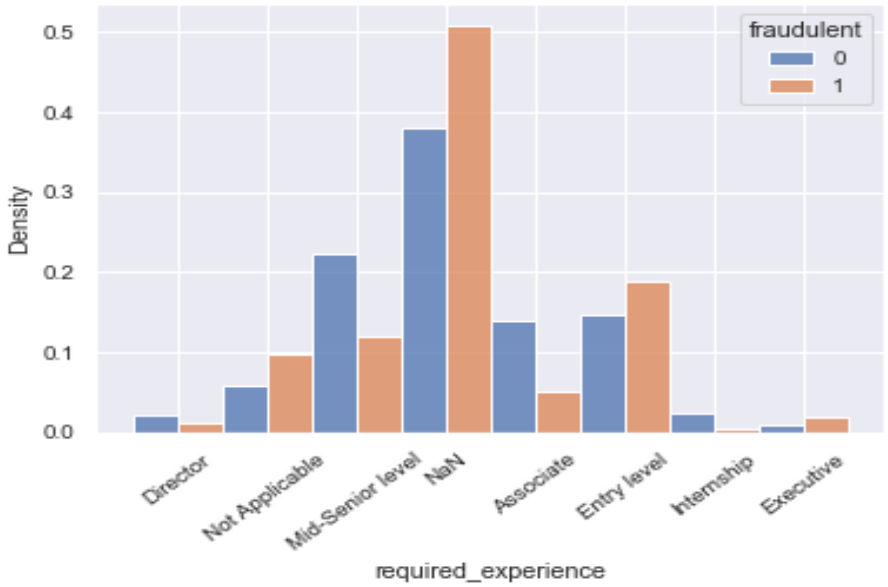

Figure 1: The Distribution of Fraudulent and Non-Fraudulent Job Listings for the "required_experience" Feature

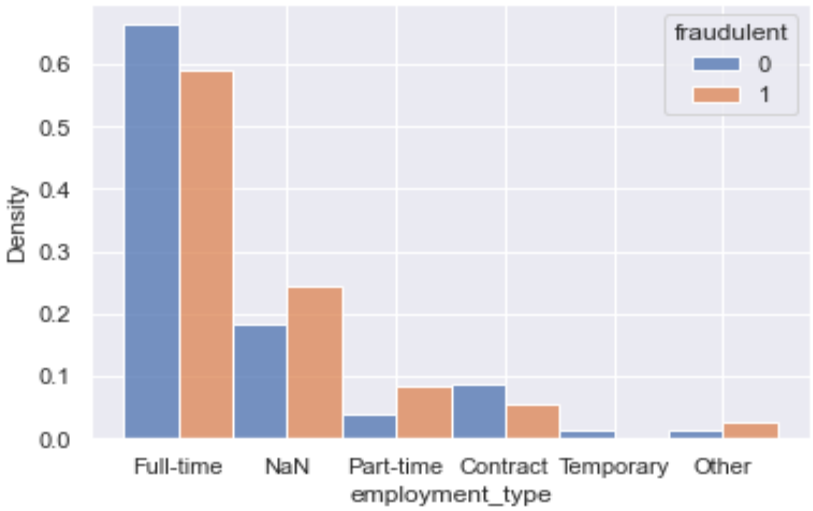

Figure 2: The Distribution of Fraudulent and Non-Fraudulent Job Listings for the "employment_type" Feature

As indicated in Figures 1 and 2, a considerably higher proportion of the "NaN" value occurs in fraudulent than non-fraudulent cases for variables "required_experience" and "employment_type". This observation suggests that the value "NaN" is important in distinguishing fraudulent and non-fraudulent job listings when we later construct a feature by converting this categorical variable to a dummy variable through one-hot encoding.

Another interesting observation about job descriptions is that a poorly written job description tends to be associated with fraudulence. For instance, many sentences ending with question marks, excessively mentioning "money", and a job description containing stylistic errors are signs of fraudulence. Therefore, this suggests that we consider the number of question marks or word frequency of "money" as informative features for identifying fraudulence.

2 Feature Creation

In addition to categorical and numerical variables, we need to construct features for unstructured data for the textual columns "company_profile," "description," "requirements," and "benefits." To proceed, we pre-process these textual contents by first removing irrelevant words such as "is," "are," and "it," in addition to white spaces, semicolons, and periods while treating two similar words with a slight difference as the same word. Then, we merge the parsed texts into one large text corpus to generate a word dictionary, of which we compute word frequency for each of the four text columns.

During this process, we faced a challenge in that for each text column, the text frequency from its merged text data contains roughly 120,629 unique words. This includes some typos and URL's that are very limited to one specific observation. If all 120,629 unique words are included in the feature matrix, the model would suffer from high variance. To address this issue, we set a threshold value of k to drop any words appearing less than k times from the word dictionary. In this fashion, we omit the low-frequency words from the feature matrix. By setting the threshold to $k = 5$, we were able to reduce the dimension of the feature matrix from 120,629 to 21,198. In the end, the dimensions of the word feature matrix is 5362 x 21,198.

3 Unsupervised Feature Filtering

Given a large number of predictors of the original word feature matrix (120,629), we use Principal Component Analysis (PCA) to filter out some features for dimension reduction. After normalization (re-scaling and centering the features), we apply PCA to the normalized text features. Then, we retain the top 2794 principal components, which account for 90% of the total data variation. However, the Random Forest model using these 2794 principal components as features tends to overfit the training data with poor test accuracy. As suggested in Table 1, the classification accuracy rate is 96% but the recall rate is especially low with 28% based on sample splitting of the original data into training and test sets with 67% observations for training while remaining 33% for testing. Therefore, we decide to seek alternatives rather than using PCA for feature generation.

		Predicted		
		0	1	Total
True	0	1543	0	1543
	1	47	19	66
	Total	1590	9	1599

Table 1: Confusion matrix based on PCA features, the recall (TP) rate is too low, Note: 1 indicates fraud, 0 indicates otherwise

4 Power Feature Creation

We now manually create features based on summary statistics obtained from each column, guided by our exploratory data analysis in Section 1 and informative analysis in Section 9. We refer to these features as power features, which are not generated directly from a text mining algorithm based on the original data. Specifically, for the four descriptive columns, "company_profile", "description", "requirements", "benefits", we create the number of characters in the descriptive texts as features. Moreover, we also create two categorical features respectively based on the minimum and maximum of the salary range. These features provide more discriminant power than those from the principal components of text features. In total, we include six power features, four of which are based on several characters in each of the descriptive columns while the other two come from the salary range.

5 Combined Feature Matrix

Finally, we construct 231 features for our classification model, including 4 features for four descriptive text columns, 2 categorical features based on the min and max of the salary range, and 225 features for the original numerical columns and one-hot encoded categorical columns. Altogether, we generate a combined feature matrix of 5362×231 to be used in our random forest model. In the next section, we will compare the results of our models for the combined feature matrix (231 features) with the word feature matrix (21,198 features).

6 Classification on the Feature Matrix

To train classifiers on training data, we consider two classification models: Random Forest and SVM. For each classification model, we compare classification results using the word feature matrix with $21,198$ features and the combined feature matrix with 231 features to demonstrate that the combined feature matrix outperforms the original word feature matrix. For training and

testing, we randomly partition the original data into training and test sets with 67% observations for training while remaining 33% for testing.

(A) Random Forest Model on word feature matrix with 21,198 features

We tune the Random Forest model using 10-fold cross validation to yield the optimal number of 180 trees. Then, we apply the tuned Random Forest model to classify the training data. In this case, we have achieved an overall accuracy rate of 97% with a recall rate (sensitivity) 36% and the true negative rate of 99%. The confusion matrix is displayed in Table 2, and the ROC curve is shown in Figure 3.

		Predicted		
		0	1	Total
True	0	1541	2	1543
	1	42	24	66
	Total	1579	30	1599

Table 2: Confusion Matrix of Word Feature Matrix for Random Forest Model

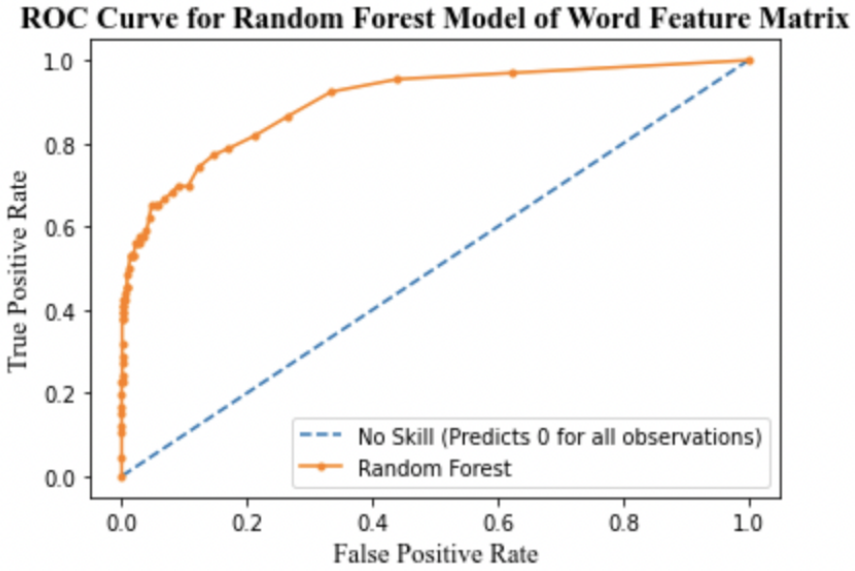

Figure 3: ROC Curve for Word Feature Matrix of Random Forest Model

(B) Random Forest on the combined feature matrix with 231 features

Again, we apply 10-fold cross validation to tune the number of trees to yield the optimal number of 200 trees and apply the tuned Random Forest model to classify the training data using the combined feature matrix. Now, we have achieved an overall accuracy rate of 97% with a recall rate (sensitivity) 45% and the true negative rate of 99%; see the confusion matrix in Table 3 and the ROC plot in Figure 4 for more details.

		Predicted		
		0	1	Total
True	0	1537	6	1543
	1	37	29	66
	Total	1574	35	1609

Table 3: Confusion Matrix based on Combined Feature Matrix for Random Forest Model

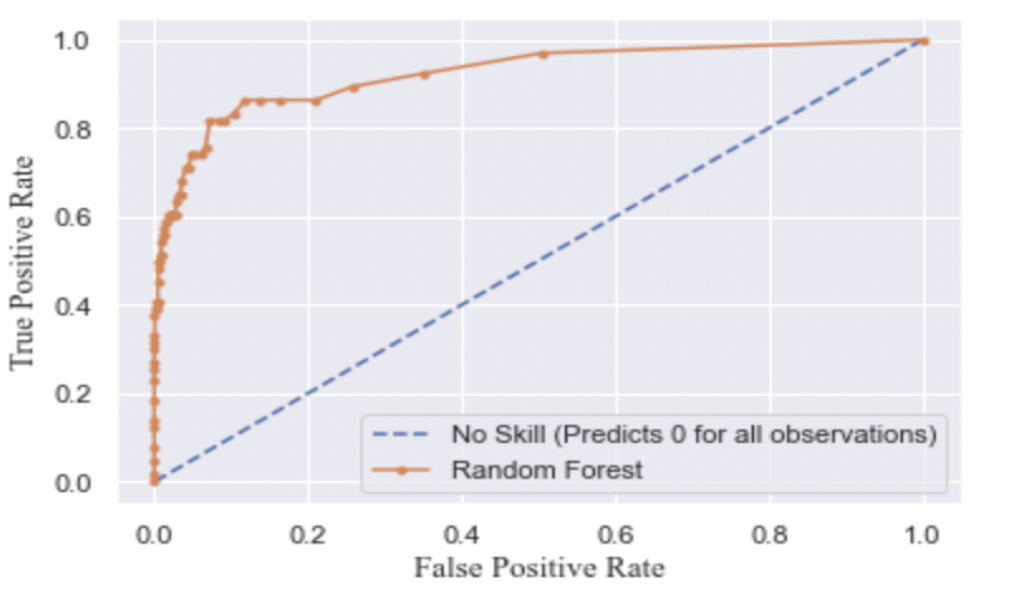

Figure 4: ROC Curve for Combined Feature Matrix of Random Forest Model

As illustrated in Figures 1 and 2, the Random Forest model achieves a higher accuracy of detecting fraudulent job listings for the 231 combined feature matrix compared to the $21,198$ word feature matrix. This is measured by the true positive rate, as the ROC curve for the combined feature matrix is closer to the upper left corner.

(C) Linear SVM on the word feature matrix with $21,198$ features

Similarly, we apply a trained linear SVM model to classify the training data using the $21,198$ word features. In this case, we have achieved an overall accuracy rate of 94% with a recall rate (sensitivity) of 20% and the true negative rate of 75%; see the ROC plot in Figure 5.

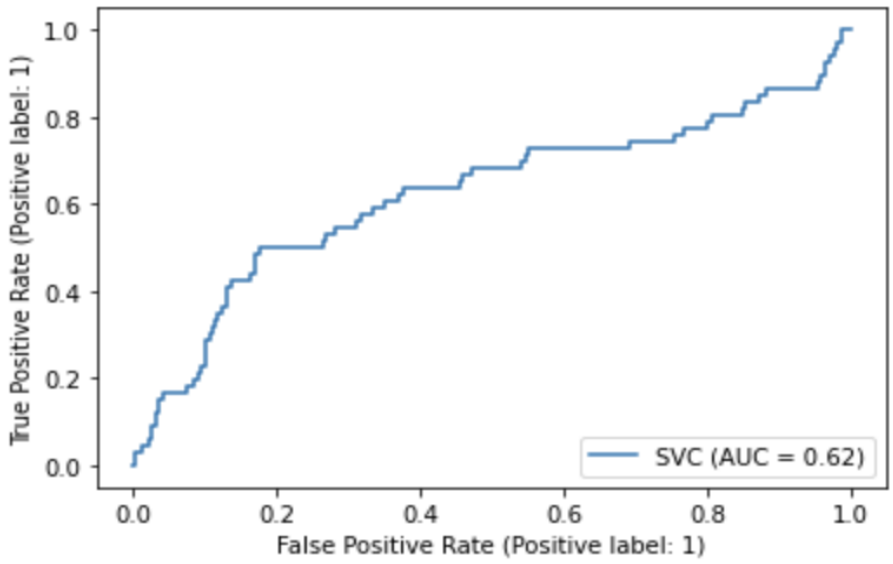

Figure 5: ROC Curve for Word Feature Matrix of SVM Model

(D) Linear SVM on the combined feature matrix with 231 features.

Similarly, we apply linear SVM to classify the training data using the 231 combined features. The overall accuracy rate is 96% with a recall rate (sensitivity) of 30%, and a true negative rate

of 83%; see the ROC plot in Figure 6.

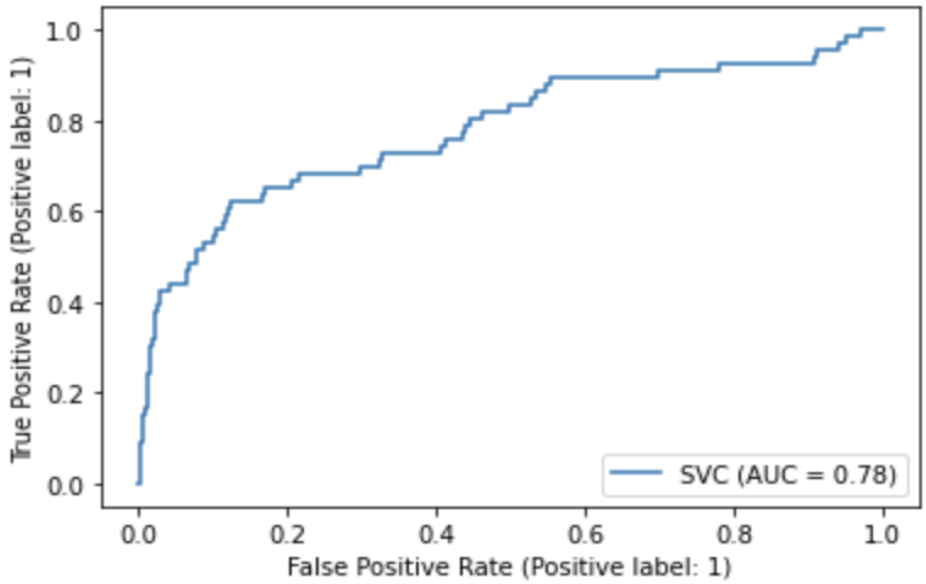

Figure 6: ROC Curve for Combined Feature Matrix of SVM Model

(E) Summarization

We now summarize our findings based on (A)-(D). First, the Random Forest model produces higher accuracy of prediction, as measured by the overall accuracy, than the SVM model for both sets of feature matrices. Second, the combined feature matrix appears to be more preferable in terms of the recall rate (sensitivity) for the Random Forest model and the true negative rate. Moreover, the combined feature matrix yields a more parsimonious representation for a classification model. It is worthy to note that the linear SVM model seems unstable in discriminating fraudulence from non-fraudulence due to its requirement of a linearly separable dataset.

7 Verification and Classification Results

Next, we apply the saved Random Forest model from the original training data (without sample splitting as before) on the combined 231 features to classify unseen new observations. To address the class imbalance with roughly 5% of the training observations being fraudulent, we use a scheme to down-weight job fraudulent instances, which is determined by a threshold value k between 0 and 1 for class prediction. To determine the threshold value, we perform 5-fold cross validation. The results of our prediction is displayed in Table 4.

		Predicted		
		0	1	Total
True	0	733	217	950
	1	1	49	50
	Total	734	266	1000

Table 4: Confusion Matrix of Random Forest Model on Unseen Test Set

As illustrated in Table 4, the overall accuracy rate on the unseen test dataset reaches 78.2% with a recall rate (sensitivity) of 98% and a true negative rate of 77%.

8 Unsupervised Clustering of Feature Matrix

First, we perform K-Means clustering ($K = 2$ or two clusters) using the combined feature matrix. By specifying two clusters, we were able to get labels for each observation. Since in unsupervised clustering, the labels can be flipped, we choose the label assignment that yields the the highest accuracy in terms of fraud classification.

To compare this clustering method for classification with the Random Forest results, we output the confusion matrix. As displayed in Table 5, this method achieves the accuracy rate of 66% with the recall rate of 41%.

		Predicted		
		0	1	Total
True	0	3437	1666	5103
	1	152	107	259
	Total	3589	1773	5362

Table 5: Confusion matrix based on Combined Feature Matrix for K-Means Clustering

Second, we perform K-Medoids clustering using the combined feature matrix in a similar fashion. The accuracy rate on the test set is 56% with the recall score of 44%, as displayed in Table 6.

		Predicted		
		0	1	Total
True	0	4179	924	5103
	1	171	88	259
	Total	4350	1012	5362

Table 6: Confusion matrix based on Combined Final Feature Matrix for K-Mediods Clustering

Finally, we perform Agglomerative Clustering using 4 types of linkage methods: Ward, Complete, Single, and Average. For "ward" linkage, the accuracy rate is 74% with a recall rate of 30%; see Table 7 for the confusion matrix.

		Predicted		
		0	1	Total
True	0	3907	1196	5103
	1	187	77	259
	Total	4094	1273	5362

Table 7: Confusion matrix based on Combined Final Feature Matrix for Hierarchical Clustering (Ward Linkage)

For "complete", "single", and "average" linkages, the accuracy rates are 95% with a recall rate of 0%; see Table 8 for the confusion matrix.

		Predicted		
		0	1	Total
True	0	5102	1	5103
	1	259	0	259
	Total	5361	1	5362

Table 8: Confusion matrix based on Combined Final Feature Matrix for Hierarchical Clustering (Complete, Single, and Average Linkage)

When comparing the results with the classification results from our Random Forest model, the true positive rate for the unsupervised clustering method is lower than that of the Random Forest model. At the same time, the overall accuracy for unsupervised clustering is lower than that of the Random Forest model. Thus, in terms of detecting fraudulent job listings, the random forest model performs better than the unsupervised clustering approach. This is anticipated as unsupervised clustering does not utilize the label information when partitioning the dataset.

9 Improvements

To improve the previous classification results from Section 7, we intend to enhance the feature generation process by comparing certain variables with the fraudulent job cases to detect any patterns. First, we visualize the text length distribution to see its power of detecting fraudulent cases by plotting the combined text length and its density for fraud and real job listings.

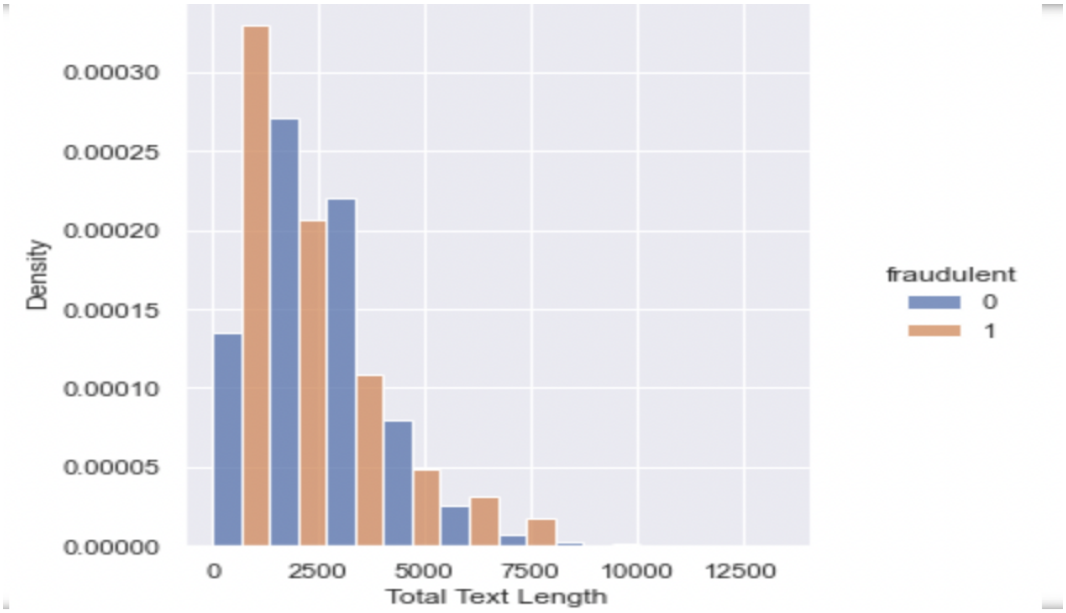

Figure 7: Distribution of Total Text Length for Fraudulent and Real Job Listings

As suggested in Figure 7, fraudulent job listings tend to have a small text length. In particular, between 0 and 1000 total text characters, there is a large gap between fraudulent and real job listings, suggesting that fraudulent job listings tend to have shorter text characters. This reasoning leads to the inclusion of text length as a feature in our final feature matrix.

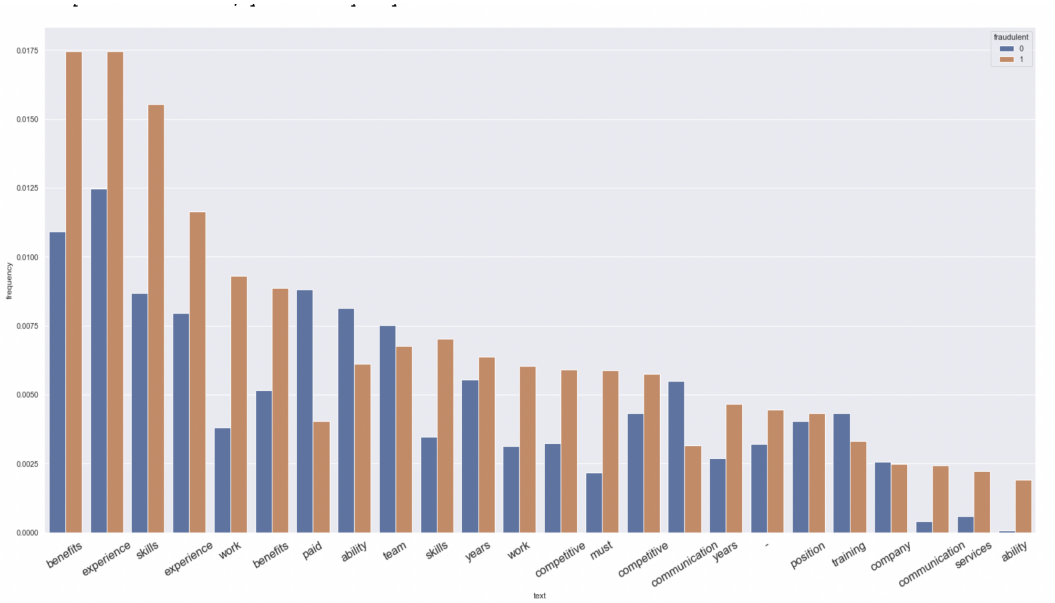

Figure 8: Distribution of Frequency for Certain Words in Text for Fraudulent and Real Job Listings

Second, we examine word frequencies for the most common words such as "benefits", "experience", and "skills" which are featured in the text for fraudulent and non-fraudulent cases, as suggested in Figure 8. However, there is a sizable gap between the frequency of some words for fraudulent and non-fraudulent job listings. As observed in our exploratory analysis in Section 1, the word "na" appears frequently in the textual contents of fraudulent job listings. Thus, we convert the "na" to "NaN" string so that it would become its category when converted by one-hot encoding.

On the ground of the above analyses, we will include some word frequencies of the descriptive columns to further expand our manually constructed features. Towards this end, we select features from a large pool of candidates. First, we include, in addition to the 231 features described in Section 5, the word frequencies of the descriptive columns including “company_ profile”, “description”, “requirements”, and “benefits”, which amounts to around 70,000 features. To filter out irrelevant features, we conduct the chi-squared independence test between each feature and the outcome (fraudulent or not) and then use cross-validation to determine the number of features to retain after the chi-squared test. The null hypothesis for the chi-squared test is that no relationship exists between the feature and the outcome; they are independent, while the alternative hypothesis states that the feature is associated with the outcome. So the larger the test statistic, the more evidence we have against the null that two variables are independent. Thus the chi-squared statistics can be viewed as a scoring function for variable importance.

For the final classification model, we also consider gradient boosting in addition to Random Forest and SVM. For each method, we perform 5-fold cross-validation to determine the hyperparameter values as well as how many features to include via the chi-squared test. Next, we will adaptively select the best 300 features from the chi-squared test for the Random Forest and Gradient Boosting models.

For the Random Forest model, we use the best 300 features from the chi-squared test by considering all features in each split and 100 trees. The testing accuracy on the verification dataset is given in Table 9. The overall accuracy is 97.8% with a sensitivity rate of 64%.

		Predicted		
		0	1	Total
True	0	946	4	950
	1	18	32	50
	Total	964	36	1000

Table 9: Confusion matrix based on Combined Final Feature Matrix, random forest

For gradient boosting, we use the best 300 features from the chi-squared test with 200 trees and a learning rate of 0.15, and a tree's max-depth of 3. The testing result on the verification data set is given in Table 11. The overall accuracy is 96.8% with a sensitivity rate of 50%.

		Predicted		
		0	1	Total
True	0	944	6	950
	1	25	25	50
	Total	969	31	1000

Table 10: Confusion matrix based on Combined Final Feature Matrix, Boosting

Overall, the Random Forest model with feature selection outperforms Gradient Boosting with feature selection, which serves as our final model for classification. By comparison, the new Random Forest model (the overall accuracy of 97.8% and a sensitivity rate of 64%) outperforms the Random Forest model based on the 231 feature matrix from Section 7 (the overall accuracy of 97% and a sensitivity rate of 45%). As such, the dimensions of our final set of features on the training data set is 5362 x 300.

By doing 5 fold cross validation on the probability threshold, we choose the probability threshold to be 0.3. That is, if an observation has above 30% chance of being fraudulent, then it is identified as fraudulent by our probability threshold.

10 Final Model and Results

Finally, with our final saved Random Forest model from the original training data on the combined 300 features, we classify unseen new observations. Like before, to address the class imbalance with roughly 5% of the training observations being fraudulent, we use a scheme to down-weight job fraudulent instances, which is determined by a threshold value k between 0 and 1 for class prediction. To determine the threshold value, we perform 5-fold cross validation. The results of our prediction is displayed in Table 11.

		Predicted		
		0	1	Total
True	0	2783	45	2828
	1	34	110	144
	Total	2817	155	2972

Table 11: Confusion matrix based on Final Feature Matrix, Random Forest Model

As illustrated in Table 11, the overall accuracy rate on the unseen test dataset reaches 97.34% with a recall rate (sensitivity) of 76.39% and a true negative rate of 98.4%.

To compare with our previous random forest model from Section 7, our final random forest model produced a higher accuracy (97.34% compared to 78%) but a lower sensitivity (77% compared to 98%). Throughout this process, we learned that there is trade-off between overall

accuracy and sensitivity, in that it is harder to produce a significantly high accuracy and high sensitivity at the same time. Due to an imbalance in fraudulent cases, it is easier to sacrifice a little accuracy to get tremendous boost in sensitivity, but it's harder to do this the other way around.

Lastly, we noticed that supervised learning produces a better model than unsupervised learning due to the ability to use labeled input and output data. It helps us design effective power features (for example: total text length) and helps the pipeline weed out irrelevant features.