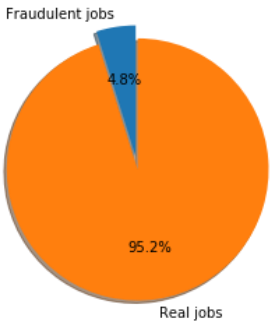
COMP5310 Principles of Data Science
Assignment 1 for project "Fake Job Posting Prediction"

Student name: Lupita Sahu
Student ID: 500426353
Unikey: lsah8006

Introduction:
There has been a significant rise in fake job postings where these postings seem fairly reasonable and it's not very difficult for job-seekers to fall prey to such scammers. At times such companies could have their own websites and even conduct fake interviews, during which the desperate job-seekers may end up sharing personal information. The goal of this project is to analyse the patterns using real data and to predict whether a job posting is fraudulent or not through machine learning, which could be used by Job portals to filter out potential fraudulent jobs.

Research Problem and Evaluation setup
The objective of this project is to create a classification model that predicts whether a posted job is fraudulent or not. To test the significance of the prediction model, we can perform hypothesis as follows.
Null hypothesis H_0: There is no significant relationship between any of the attributes of a job posting and the authenticity of the job post and we cannot predict the authenticity of a job using independent variables
Alternate hypothesis H_1: There is a significant relationship between one or more attributes and authenticity of the job post and this can be used to build the classification model.
Due to the highly imbalanced nature of the dataset with a ratio of 95:5 for real and fraudulent job posts, we cannot use accuracy as a factor to measure the accuracy of our classification models; so we will rely on f1-score.



Approach
The dataset was downloaded from Kaggle via the following url:
https://www.kaggle.com/shivamb/real-or-fake-fake-jobposting-prediction

The data set contained 17880 observations with each observation corresponding to a job post along with a label depicting whether it's fraudulent or not. One of the data which was labeled as fraudulent had most of the fields set to NA. Hence it was important to preserve NA values as well. So all NA values were replaced with "NoValue", so that there is no loss of information.

The total number of variables is 18 including the target variable. Other than a few categorical ones, most of the variables consisted of unstructured data such as company profile, job description, requirements, benefits, etc. Such data could not be directly used for any analysis or any machine learning purpose; hence it needed to be transformed first. This was done first by combining all such attributes into one. By using "nltk" library, the text was then cleaned by removing stop words, numbers, special characters, and single letters and was converted to lower case to remove case sensitivity. The individual words were then separated and converted into tokens.

We performed some initial exploratory analysis using various independent variables using graphs and observed some interesting findings from this dataset:

- Most fraudulent jobs were targeted towards “Entry-level” jobs. We can even observe the word “entry” in the word cloud for fake jobs [Appendix fig-4]
- For fraudulent jobs they are slightly more inclined towards High school graduates. Also the percentage of NA values is more for fake as compared to real jobs. [Appendix fig-5]
- We see an increased amount of job posts where required experience was missing for fraudulent jobs. Also we may notice that percentage of "Entry level" roles are more for fake jobs as compared to the real jobs [Appendix fig-6]
- More than 2/3rd fraudulent jobs didn't have screening questions. [Appendix fig-7]
- Similarly 2/3rd of the fraudulent jobs didn’t have a company logo. [Appendix fig-8]
- Fraudulent jobs had less details in Company description as compared to their real counterparts. [Appendix fig-9]
- Another common characteristic of fraudulent jobs was the absence of values in many of their attributes, which can be observed in the word cloud [figure-4 from Appendix]. The word “NoValue” in the word cloud signifies the presence of NA values.

An initial logistic model was applied without using the unstructured data and the result was poor with only 0.19 as f1-score. This was expected as we had not yet considered the text data. So text mining was resumed where now each token was tagged with a part of speech. The next step was to remove grammar from these tokens i.e. lemmatization, after which tokens were converted into their most basic forms so that we get a better count of words. Afterward using scikit-learn's CountVectorizer a sparse document term matrix was prepared which included both unigrams and bigrams, which was later converted into a dense matrix and added to the main dataset. Only tokens above a certain threshold were retained to keep the dataset size optimal.

Once the dataset was ready, a logistic regression model was applied again with parameter tuning and this time it rendered an f1-score of 0.73. As we can see there is still room for improvement, a couple of other models were also applied such as Support Vector Classifier, Neural Network MLP Classifier, Naïve Bayes, and Random Forest classifier. Since this is a highly imbalanced dataset, the models were trained keeping this in mind. 70% of the data was allocated to training the model to include more of the fraudulent data in the training dataset. Also we decided to analyse the Precision-Recall curve for the very same reason. The curve for the Logistic regression on the validation dataset looked fine with good area under the curve and precision value looked fine for the majority of recall values [Refer to fig-2 in Appendix].

The next model was the Support Vector Classifier. We made use of the parameter class_weight = ‘balance’ to give more weight to less frequently appearing class. After tuning parameters, the different accuracy scores were observed to be quite similar to the Logistic model with an f1-score of 0.73. Refer to fig-3 in Appendix for confusion matrix and precision-recall curve.

Neural Network MLP classifier was then applied with a range of parameters for tuning and this resulted in better accuracy scores, with 0.78 as f1-score on the validation dataset. We tried ‘adam’ and ‘sgd’ solver and ‘adam’ gave better accuracy. The precision-recall curve looked pretty good with more area under the curve giving good output for precision and recall values. [Appendix fig-1]

Other models such as Naïve Bayes and Random Forest were also tried. Surprisingly Random Forest could only give an f1 score of less than 0.6 and after parameter tuning it was

shockingly reduced to less than 0.5. Naïve Bayes also could not produce an accuracy even close to any of the 3 models we have discussed above. So the detailed results of these models are not included in this report.

Since the model is highly imbalanced with only 866 observations containing fake posts, I also tried down-sampling the other class to reduce the ratio from 95:5 to 60:40. However the results were not very reliable as the number of observations was drastically reduced from 18k to only 2215. The models generated good accuracy on train data and validation data, but failed to provide similar results on the held-out test data. When a paired t-test was performed using the results obtained from our final model and the results obtained using the model obtained from down-sampled data, the results turned out to be better for the Neural Network MLP classifier on the larger dataset. The reason could be that classifier models preferred more data over less data to render a reliable prediction.

Analysis and Results

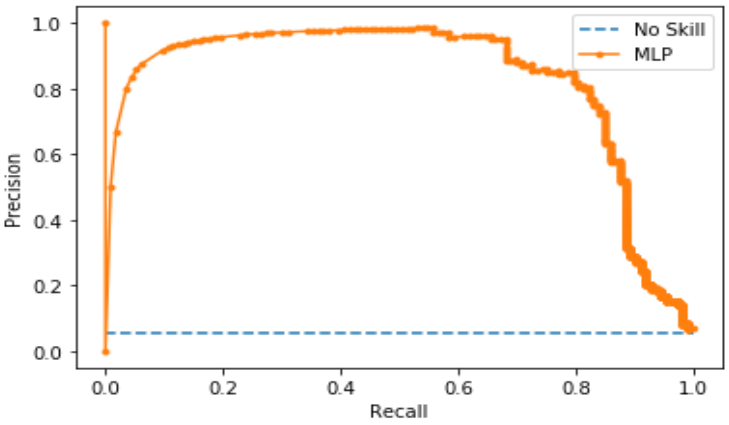
To evaluate each classifier, the held-out test data was used to calculate predicted values using each classifier separately. 10-fold cross-validation was used to calculate f1-scores on test data folds using each classifier. The F1-scores from these models were then compared using one-sided paired t-test and the results were as follows:

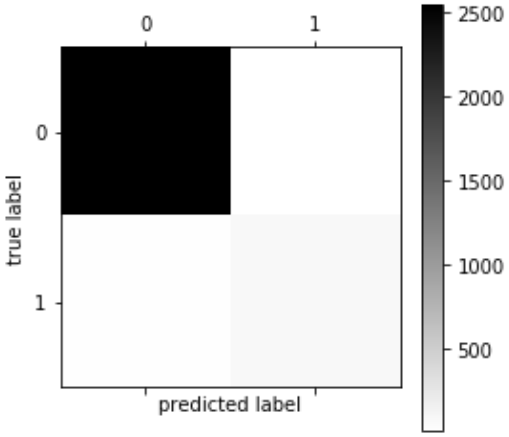
1. Between Logistic regression and SVC, the p-value was 0.1, meaning there wasn't much difference in the f1 scores.
2. Between SVC and MLP, the p-value was 0.0008, meaning there was a significant difference in the f1 scores. And the negative t-statistic value indicated that the MLP predicted better than SVC.
3. Similarly, a low p-value of 0.0009 and negative t-statistic values suggested that MLP was a better model than Logistic regression as well.

Below are images of the confusion matrix and precision-recall curve we obtained by using the final. Due to the imbalanced nature of the dataset, the fraudulent jobs are in a lighter shade. However from the confusion matrix table below, we can see that we got a fair prediction using the Neural Network MLP Classifier model. The precision-recall curve for the MLP model looked better than that of SVC and Logistic models, with very good precision for the majority of recall values.

Predicted	0	1
Actual		
0	2555	14
1	29	84

	precision	recall	f1-score	support
0	0.99	0.99	0.99	2569
1	0.86	0.74	0.80	113
accuracy			0.98	2682
macro avg	0.92	0.87	0.89	2682
weighted avg	0.98	0.98	0.98	2682





To answer our research question, whether there is a significant relationship between the dependent and independent variables in our dataset, we collected predicted variables from the folds of held-out test data. Compared them with the actual values from the folds of test data using a 2-sided paired t-test. Our null hypothesis for this t-test was that both values were fairly similar. We then looked at the average of all the p-values, which was 0.3; hence we failed to reject the null hypothesis. This proved that our final classifier was able to predict with good accuracy.

Conclusion

Our research question was if there was a relationship between attributes of the dataset and the target variable 'fraudulent' and if we could use the variables to predict the fraudulent nature of a job post. This was the null hypothesis of the 2-sided paired t-test we performed using actual and predicted outputs. Since we have already failed to reject the null hypothesis we can conclude that we can predict the fraudulent nature of a job post by looking at the features in the given dataset.

There is a lot of scope for improvement in this project. Accuracy scores could be much better. For example, from the precision-recall curve, we obtained from the final held-out test data, the lower values of recall rendered poor precision values. By trying out other models such as XGBoost or deep learning, accuracy could be improved to some extent. However the best way to improve accuracy would be to perform better feature engineering. We could also try giving priority to bigrams over unigrams or try trigrams. Our model will fail to predict if there is any pattern containing alphanumeric characters, mixed case words (ex: s@l@ry, LucraTiVe) etc on fraudulent jobs.

Thank you for reviewing the report!

References

1. https://luminate.prospects.ac.uk/the-rise-of-fake-job-adverts-and-recruitment-fraud-
2. Real or fake job posting prediction, viewed 20/05/2020 https://www.kaggle.com/shivamb/real-or-fake-fake-jobposting-prediction

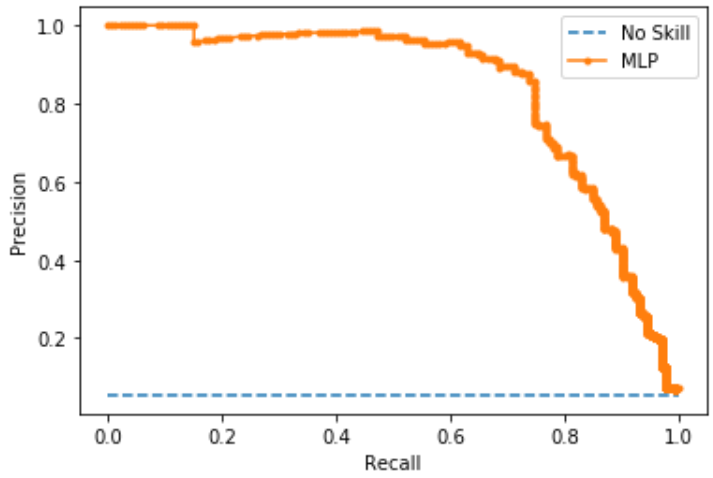
3. https://towardsdatascience.com/text-mining-for-dummies-text-classification-with-python-98e47c3a9deb - for ideas of text mining
4. https://www.datacamp.com/community/tutorials/text-analytics-beginners-nltk
5. https://machinelearningmastery.com/roc-curves-and-precision-recall-curves-for-classification-in-python/
6. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4349800/

Appendices

Figure 1: Confusion matrix and precision-recall curve for Neural Network MLP on validation data using 'adam' solver, max_iter = 500 and hidden_layer_size=200

	precision	recall	f1-score	support
0	0.98	1.00	0.99	2536
1	0.89	0.69	0.78	146
accuracy			0.98	2682
macro avg	0.94	0.84	0.88	2682
weighted avg	0.98	0.98	0.98	2682

Predicted	0	1
Actual		
0	2524	12
1	45	101



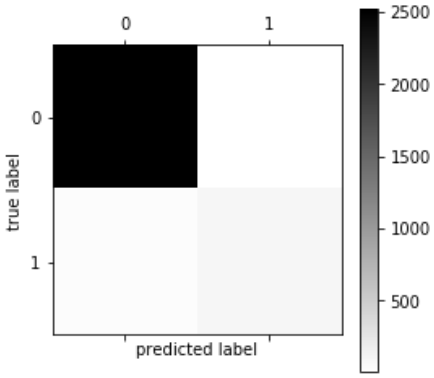


Figure 2: Confusion matrix for Logistic regression and precision-recall curve on validation data using C = 1 and penalty = l2

	precision	recall	f1-score	support
0	0.98	0.99	0.99	2536
1	0.81	0.66	0.73	146
accuracy			0.97	2682
macro avg	0.90	0.82	0.86	2682
weighted avg	0.97	0.97	0.97	2682

```
[[2514   22]
 [  50   96]]
```

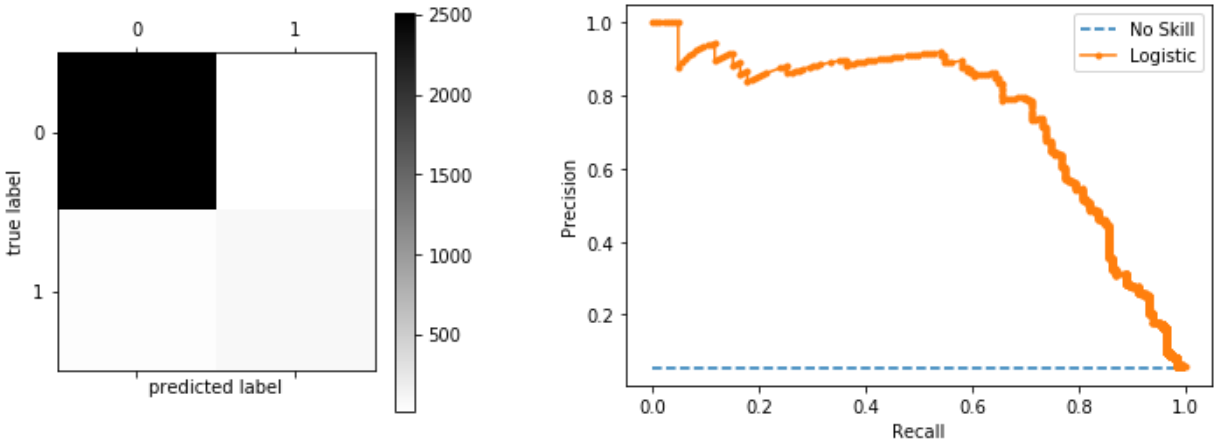


Figure 3: Confusion matrix and precision-recall curve for SVC using kernel = 'linear'

```
[[2499   37]
 [  40  106]]
```

	precision	recall	f1-score	support
0	0.98	0.99	0.98	2536
1	0.74	0.73	0.73	146
accuracy			0.97	2682
macro avg	0.86	0.86	0.86	2682
weighted avg	0.97	0.97	0.97	2682

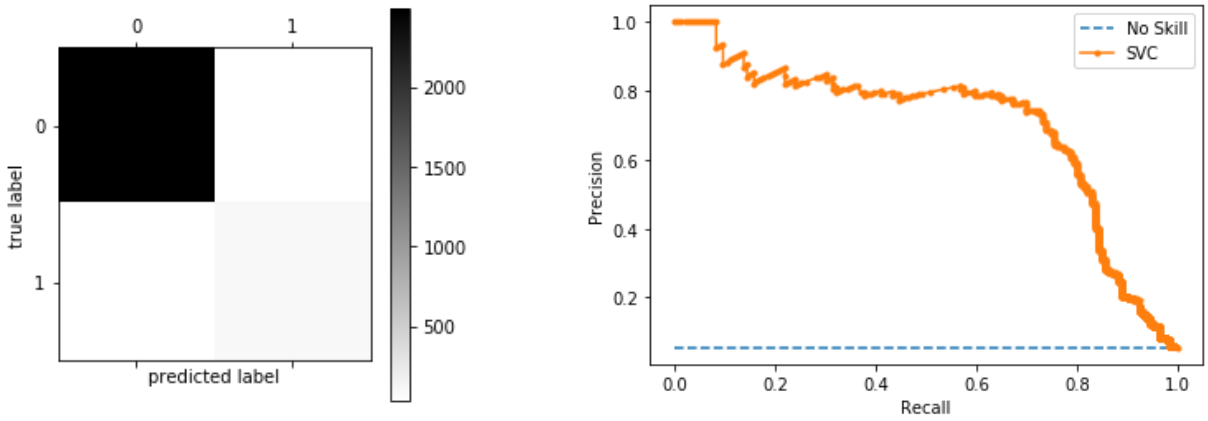


Figure 4: Word cloud for fake job posts

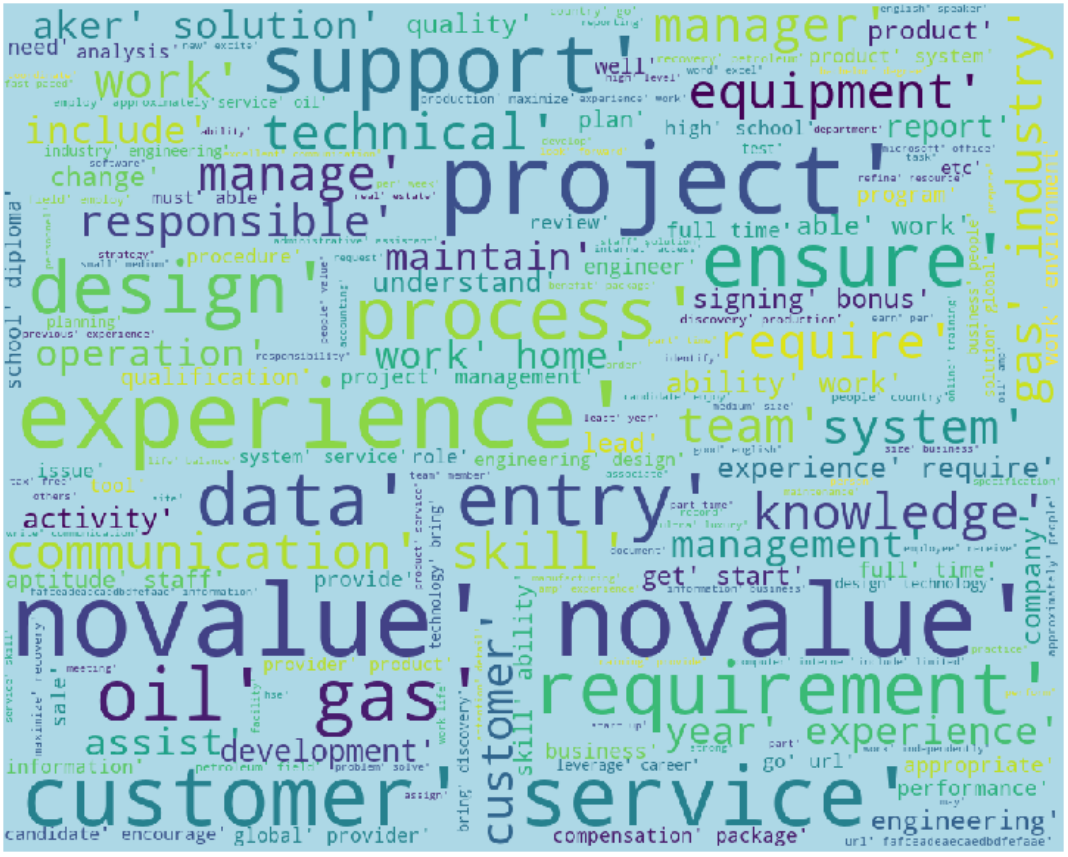
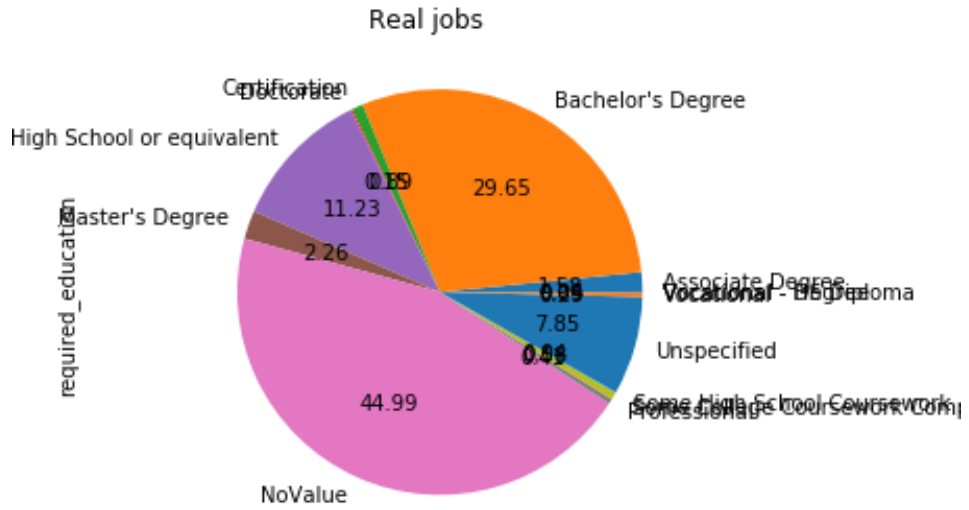


Figure-5: Required education "**High school"**: More portion of fraudulent jobs is inclined towards this group as compared to real jobs. Also the percentage of NA values is more for fake as compared to real jobs.



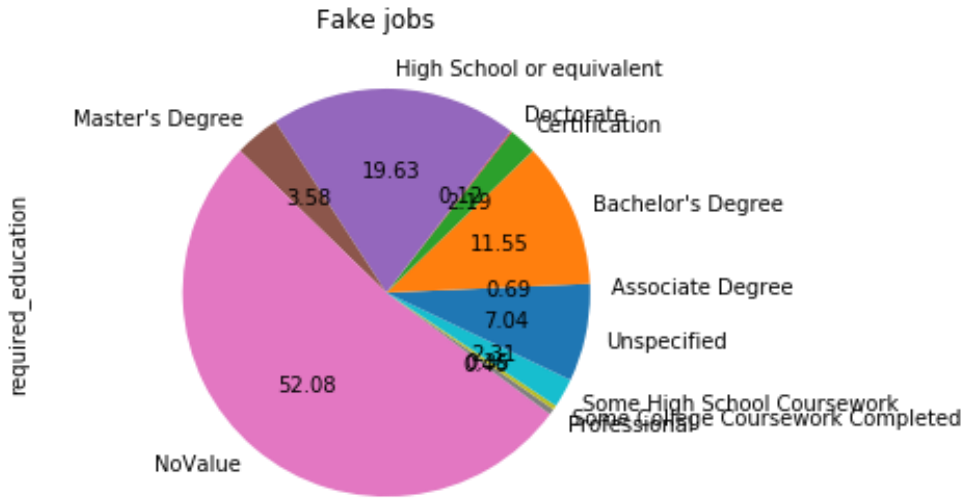


Figure 6: Comparison for Required experience: We see an increased amount of job posts where the required experience was missing for fraudulent jobs. Also we may notice that percentage of "Entry level" roles are more for fake jobs as compared to the real jobs.

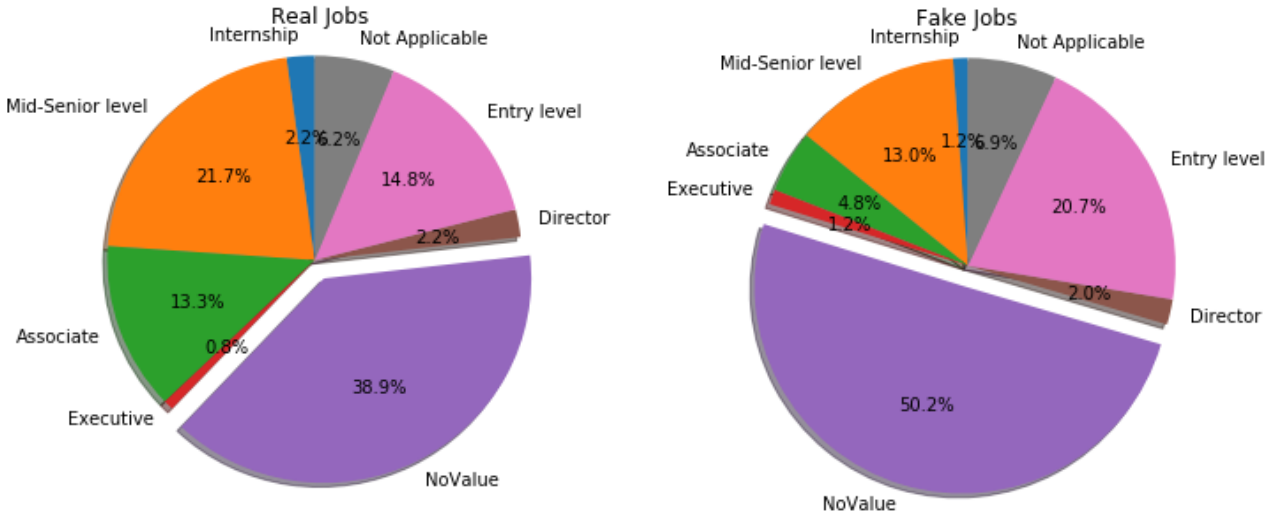


Figure 7: More than 2/3rd fraudulent jobs didn't have screening questions.

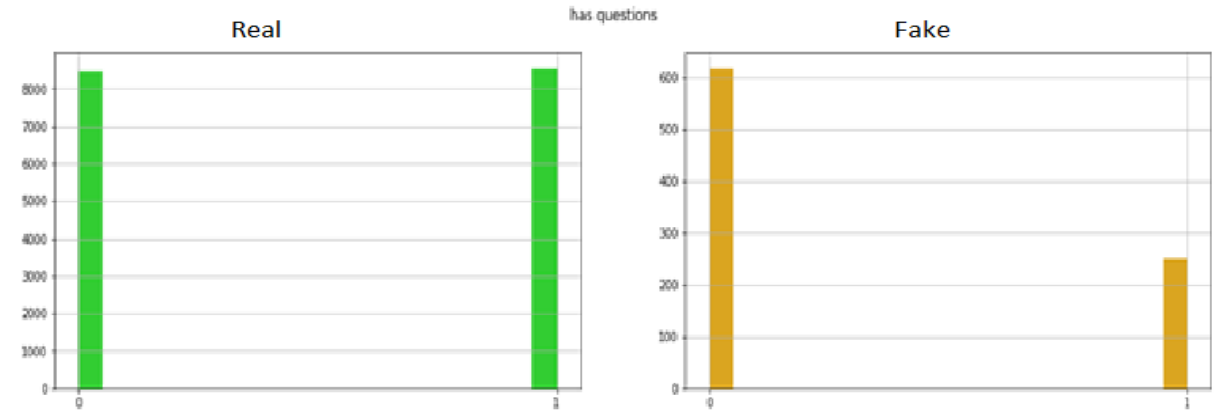


Figure 8: Similarly 2/3rd of the fraudulent jobs didn't have a company logo

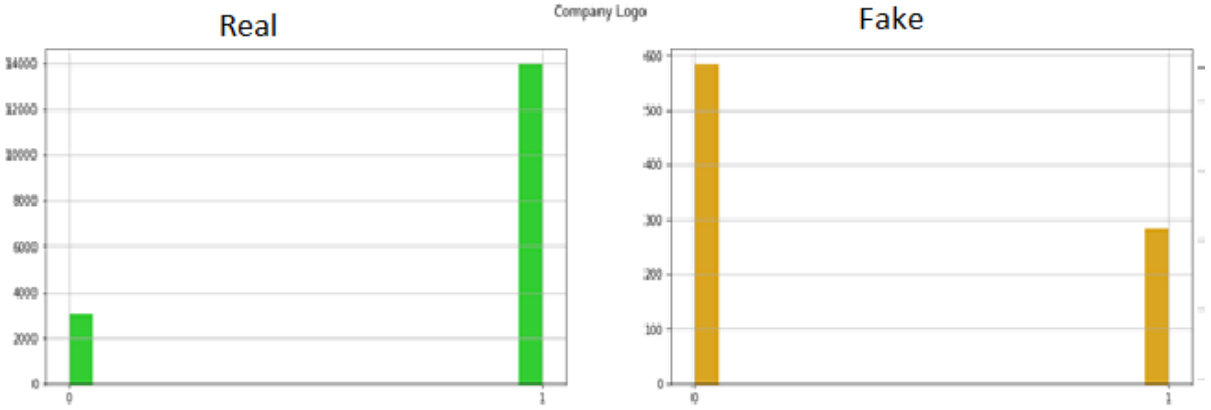


Figure 9: Fraudulent jobs had less details in Company description as compared to their real counter-part

Figure 10: f1-score with and without “location”:

Without location

	precision	recall	f1-score	support
0	0.98	0.99	0.99	4254
1	0.74	0.64	0.69	216
accuracy			0.97	4470
macro avg	0.86	0.81	0.84	4470
weighted avg	0.97	0.97	0.97	4470

```
[[4206   48]
 [  78  138]]
```

with location

	precision	recall	f1-score	support
0	0.98	0.99	0.99	4254
1	0.75	0.65	0.69	216
accuracy			0.97	4470
macro avg	0.87	0.82	0.84	4470
weighted avg	0.97	0.97	0.97	4470

```
[[4207   47]
 [  76  140]]
```

The difference is almost none, so “location” was removed from the final dataset to form a parsimonious model