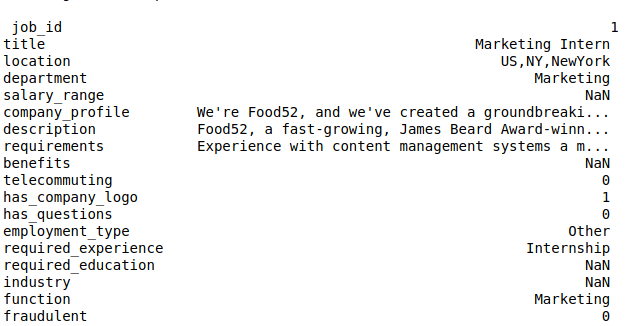
COMS3007: Machine Learning Assignment 2020

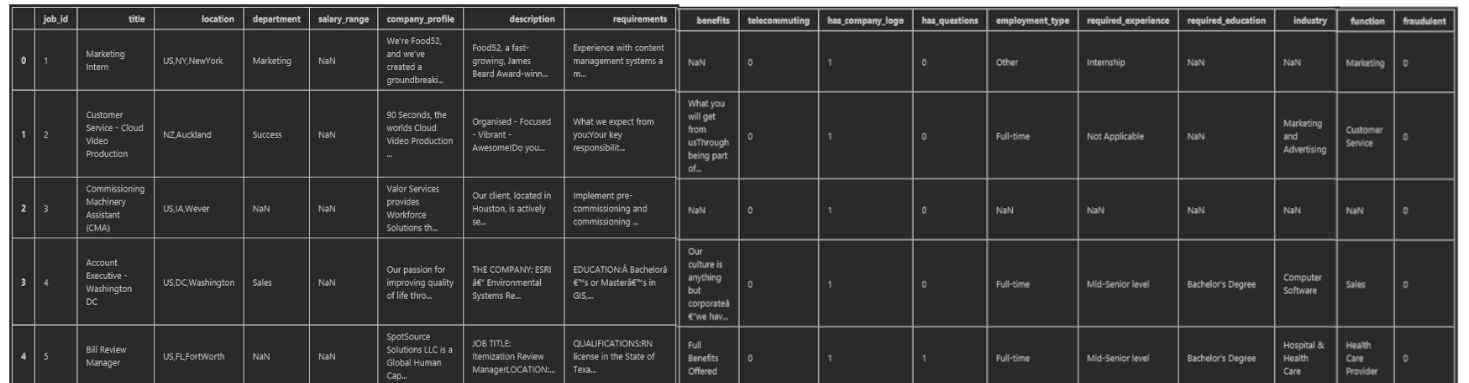
Group Members

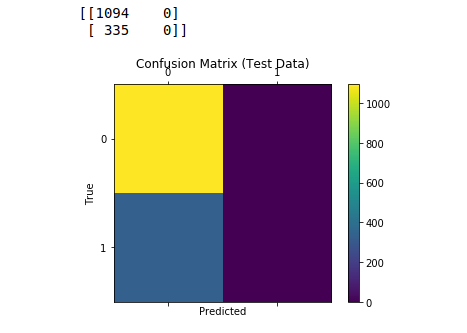
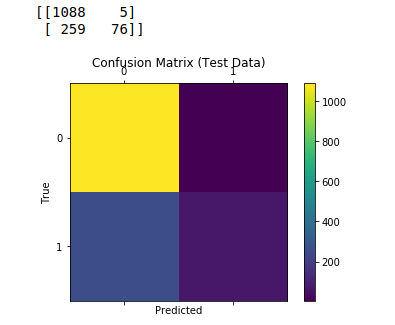
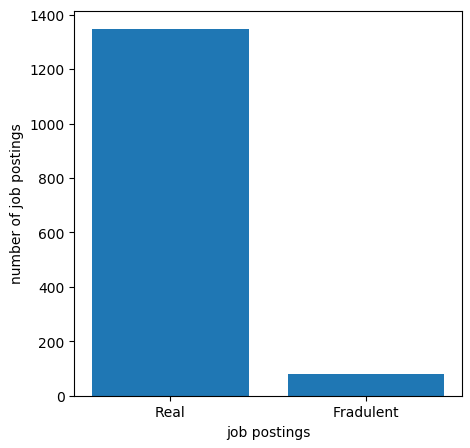
|  |  |
| --- | --- |
| Bakang Majova | 1733164 |
| Maccauley Marota | 1637447 |
| Thabo Rachidi | 1632496 |

1. Our dataset has just under 18 000 job descriptions with around 800 of them are being fraudulent. The data set contains 13 string/text columns, 4 Boolean (0/1) columns and an ID column. We are using this dataset to create classification models that can learn whether a job post is real or fraudulent.

|  |  |  |
| --- | --- | --- |
| Attribute | Description | Type |
| Title | The title of the job posting | String/Text |
| Location | The Geographical location of the job posting. | String/Text |
| Department | A corporate department within the company (e.g. Sales). | String/Text |
| Salary range | An Indicative salary range. | String/Text |
| Company profile | A brief company description. | String/Text |
| Description | A detailed description of the job post. | String/Text |
| Requirements | Enlisted requirements for the job opening | String/Text |
| Benefits | The benefits offered by the employer | String/Text |
| Telecommuting | It is true if it is a telecommuting position | Boolean |
| Company logo | It is true if the company logo is present | Boolean |
| Questions | It is true if screening questions are present | Boolean |
| Employment type | The type of employment for the job posting (Full-type, Part-time, Contract, etc.) | String/Text |
| Required Experience | The experience required for the job posting (Executive, Entry level, Intern, etc.) | String/Text |
| Required Education | The education required for the job posting (Doctorate, Master’s Degree, Bachelor, etc.) | String/Text |
| Industry | The industry that the job posting belongs to (Automotive, IT, Health care, Real estate, etc.) | String/Text |
| Function | The function the job posting belongs to (Consulting, Engineering, Research, Sales etc.) | String/Text |
| Fraudulent | Classification attribute | Boolean |

**Sample Datapoint**

****

1. The data set we used is in an excel spreadsheet. The inputs are structured in such a way that each data point has all its features and the corresponding class it belongs to, which can either be 0 or 1 to represent whether it is a real or fraudulent job post, respectively.  
     
   We found that some values in certain features were the same but had upper-case letters while others had lower-case letters. To fix this in code, we converted all entries to lower-case letters so that when we count unique entries for each of the features, we don't count the same entries more than once.  
     
   We pre-processed the dataset by replacing all blank values and values marked as null to ‘missing’ as this was easier to use.   
   We also removed punctuation marks from some of the feature values.  
     
   We split the dataset into training, validation and testing data. We allocated approximately 60% of the dataset to be training data, 30% to be validation data, and the remaining 10% to be testing data.
2. The classification algorithms we used are Naive Bayes Classifier and Logistic Regression:
   * **Naive Bayes Classifier:**
     + We used the Naive Bayes formula: P( y | x ) = ( P( x | y ) \* P( y) ) / P( x ),
     + We then used the training dataset to compute the prior probabilities, which in the formula would be represented by P(y). The prior probability is a probability of any job posting being fraudulent and the probability of it being non-fraudulent (real) before observing any more data.
     + We then create a class conditional model which includes calculating the likelihood, which in the formula is given by P(x | y). This probability asks, given that the job posting is fraudulent / non-fraudulent, what is the probability of seeing this particular job posting?   
       Calculating likelihood includes:
       1. We find all the unique values in each feature of the dataset.
       2. For each of the unique values in each feature, we compute the probability of it belonging in a job posting that is fraudulent and a job posting that is non-fraudulent.
     + Now that we have fraudulent and real probability for each feature, we can use the independence rule of Naive Bayes, which assumes that each of the features is independent of one another. This means each feature is equally important and this means given a job posting with certain features we can calculate the probability of that job posting being fraudulent and the probability of it being non-fraudulent by multiplying the probabilities of those feature values given a specific class target.
     + As an example, given a job posting with the title ‘Marketing intern’, location ‘us’, salary range ‘5000-10000’, etc. We can get the probability of the title given that the job posting is fraudulent, probability of location given that job posting is fraudulent, etc. Which we then multiply together to get the likelihood.
     + After obtaining the likelihoods we multiply them by the corresponding prior probability we computed to get the numerator according to the Naive Bayes formula. Since the denominator is going to be the same for both classes (fraudulent & non-fraudulent), we just calculate the numerator and using the MAP (maximum a posteriori probability) which simply says we take the maximum probability between the two classes and that becomes the class label for that data point.
     + Note: We had to implement sentiment analysis on the features company profile, description, requirements and benefits because they are heavily text-based.
       - We now have the likelihood of each unique word belonging to a fraudulent or real job post.
       - When we get a new job post we extract the features listed above and encode them. We then use the encoded array to help us find the likelihood of an individual word in a specific feature of the data post belongs to a fraudulent or real job post.
       - We use the likelihoods of each word to give use the overall likelihood that the feature is fraudulent or real.
     + Errors of the model in the form of a confusion matrix:
   * **Logistic Regression:**
     + We used logistic regression to classify our data. We chose logistic regression because it allows us to model the probability that each input x we get belongs to a particular category.
     + We have the data X= {x 0, ..., x (n) } and labels Y = {y 0 , ... , y (n) } and essentially we wanted to learn the function y = f(x, θ) to predict y for a new x.
     + We used Binomial or binary logistic regression since we only have two possible outcomes for y {0,1} which represent real or fraudulent outcomes for a job post.
     + To implement logistic regression we had to encode our categorical features. We were then able to use the sigmoid (logistic function) to generate probabilities which are between 0 and 1.
     + We need to find the best values for our weights. So we initialised our weights to zero and measured how well the algorithms perform on the initialised weights. We measured it using the loss/cost function
     + We need to minimise the loss function, so we do this by increasing/decreasing the weights. We do this by getting the derivative of the loss function for each of the weights. This process is known as gradient descent
     + We then update the weights by subtracting them by the learning rate times the learning rate. We repeat this process several times until we reach an optimal solution.
     + After training a few times we found that using a learning rate of 0.7 and 7000 iterations gave us the most optimal predictions.
3. Our algorithms were not very accurate at their predictions. There could be several reasons, for example when we examine the dataset there are many more real job posts than fraudulent, so it's possible the model did not have enough exposure to fraudulent job posts and became biased. Since Naïve Bayes is biased as it assumes that the features are conditionally independent to each other, it might also be that the dataset did not follow the bias of the Naïve Bayes thus making it to perform poorly/not good. Logistic regression worked out better for classifying the job posts than our Naive Bayes model. The best performance we can achieve on this dataset is to predict real job postings because our models interacted a lot with real job postings. Our recommendation to whoever who wants to work with this dataset would be to use logistic regression classifier instead of using Naïve Bayes classifier to predict if a job posting is fraudulent or not.

# Dataset link:

* https://www.kaggle.com/shivamb/real-or-fake-fake-jobposting-prediction