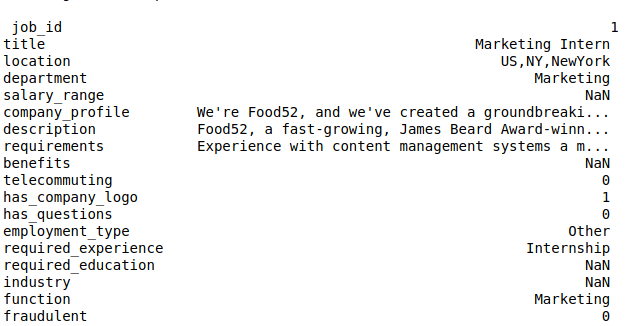
COMS3007: Machine Learning Assignment 2020

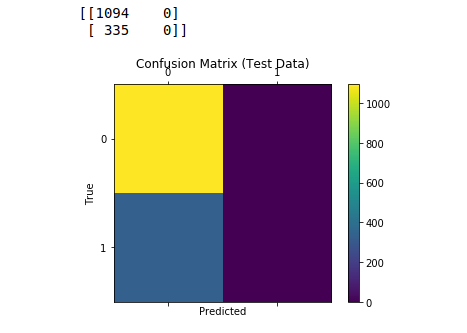
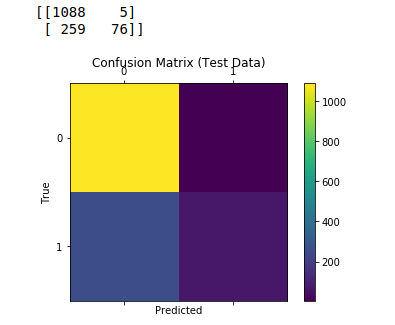
Group Members

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

1. Our dataset has just under 18 000 job descriptions and around 800 of them are 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 | Geographical location of the job posting. | String/Text |
| Department | Corporate department (e.g. Sales). | String/Text |
| Salary range | Indicative salary range. | String/Text |
| Company profile | A brief company description. | String/Text |
| Description | The detailed description of the job posting. | String/Text |
| Requirements | Enlisted requirements for the job opening | String/Text |
| Benefits | Enlisted offered benefits by the employer | String/Text |
| Telecommuting | True for telecommuting positions | Boolean |
| Company logo | True if company logo is present | Boolean |
| Questions | 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 Data point

1. The data set we used is in a 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 values to lower-case letters so that when we count unique values for each feature we don’t count the same values more than once.  
     
   We preprocessed the dataset by replacing all blank values and values marked as null to ‘missing’ as this was easier to work with.   
   We also removed punction marks from some of the feature values.  
     
   We split the dataset into training, validation and testing data, we allocated 60% of the dataset to be training data, approximately 30% to be validation data and approximately 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 proir probability is the 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 is simply asking, given that the job posting is fraudulent / non-fraudulent, what is the probability of seeing this particular job posting?   
       Calculating likelihood includes:
       1. Finding 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 this information we can use the independence rule of naive bayes which assumes that each of the features are 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 beng non-fraudulent by multiplying the probabilities of those feature values given a certain 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 probaility 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 these 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.
       - This allowed us to have the likelihood of each unique word belonging to a fraudelent 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 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 features which are catogorical. We were then able to use the sigmoid(logistic function) to generate probabilites which are between 0 and 1.
     + We need to find the best values for our weights. So we initiialsed our weights to zero and needed to measure how well the algorithms performs on the initialsed weights. It is measured using the loss/cost function.
     + We need to minimise the loss function and we do this by increasing/decreasing the weights. We do this by getting the derivative of the loss function with respect to each weight. 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 a number of reasons, for example looking at the dataset there are many more real job posts than fraudulent and so the model just may have not have had enough exposure to fraudelent job posts and became biased. Logistic regression worked out better for classifing the job posts than our naive bayes model.

# Dataset link:

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