|  |  |
| --- | --- |
| Name: Abdalrhman Alaa Eldeen Elsyed | Industry classification task |

1. Explain my solution:

* First I read data and try to get insights about it
* There was no missing data but I noticed there is a lot of duplicated data so I remove it
* I tried to see if the length of job title is important to distinguish between class but it didn’t
* I removed the unnecessary words from the text (it, was, etc.) as well as the punctuations because it doesn’t have an impact to the output
* I created bag of word model or document represented as vectors of words or sparse matrix
* I enchained it by adding term frequency-inverse document frequency(TFIDF) to represent the importance of the word in the corpus
* Then I split the data into training set and test set with ratio of 0.33% for test set
* I fit the training set into classifier to train on then test it on test set
* To see my result, I used F1 score on confusion matrix
* I save my classifier as python object to use it again when I want
* I create new python script to implement Http
* I loaded my classifier then created a route to send request
* The user passes the request with argument job title = and give his job title
* The server uses the classifier to predict the industry
* Server send back the industry to the user

1. Notes on my solution:

* I used two classifiers because naïve bayes is recommended for text classification because it works well with great feature space like sparse matrix (sparse matrix have lots of features with zero in it) but I also tried ensemble learning because it not only one classifier but you can put many and that enhance the output and also from my experience it was always the best classification algorithms in output
* I used confusion matrix because data is imbalance so the if there is classifier can’t ignore the low data class in normal accuracy way you can’t tell if your classifier is doing good or not (your accuracy may be high while your classifier miss entire class) but with confusion matrix and precision and recall (True positive, True negative, type one error and type two error) you can see easily what your classifier do in each class
* I choose the Random Forest classifier over the naïve classifier even the both close in accuracy because the random forest classifier get more F1 score on Accountancy label (80%) and the for all classes the F1 was close but naïve did great on the classes that have higher data but did bad on Accountancy class (55%)
* The user must specify the argument with request (?job title = ) otherwise the server will return hint on how to use it
* The libraries that I have used (flask, scikit-learn, numpy, matplotlib, seaborn, pandas, string, nltk and pickle)

1. Answer your questions:

* The data had lots of duplicated data so I remove it using pandas built in function
* I choose naïve bayes because it is simple to implement and it works well with great feature space like sparce matrix and random forest because it has inside many classifiers that vote for the output so it powerful and binary tree the classifier inside the random forest is very fast at getting the output -> O(depth of the tree)
* Dealing with imbalanced data: used confusion matrix to evaluate accuracy instead of normal evaluation of accuracy
* To have better performance: we can try to get more data if it is possible and may be try to get more features like the what is the B.A. because some B.A.s can tell what will the person will on
* I evaluated my model with F1 score on confusion matrix
* The limitation of naïve bayes is it can break. It breaks in funny ways like (the is team called Chicago Bulls) so if you try this with naïve it will not understand it is sport team but it will get it literally as bulls in Chicago
* Random forest limitation is it is slow because there are many classifiers getting the output so you have to wait until everyone is done then collected the votes but because the random forest is using binary tree and binary tree is fast on getting the output it reduces the time somehow