Binghamton University, Watson School of Engineering

Termination Project:

CV Selection System

Name: Rahul Verma

Email: rverma4@binghamton.edu

B-Number: B00892091

Termination Project (CS-595-10)

Prof. Leslie Lander

April 17, 2023

CONTENTS

1) ABSTRACT

One of the most crucial items needed for someone to find a job is a resume. It is essentially a depiction of all the person has accomplished, including all of their achievements, qualifications, and skill sets, in addition to a summary of all of their extracurricular activities throughout their academic careers.

A perfect resume is required for any one of us who wishes to get a job quickly, a good resume makes the process of selection so much faster. It is a thing that differentiates us from all the competition. Thus, a good-quality resume makes a huge difference in a person's career.

This project focuses on making a system based on machine learning which will analyze the resume dataset and come up with what keywords are the most used, and what makes a resume stand out. Also, it will make the employers' job much easier by using this system by integrating it into their system. Thus, making screening of multiple candidates faster. This in turn saves a lot of resources, time, and effort.

2) INSTALLING, IMPORTING LIBRARIES AND LOADING DATASET

This project uses the following 8 libraries in Python for the project, they are as follows along with their brief description:

1. WordCloud: It is basically a representation of text data where the size of the word directly corresponds to its frequency.
2. Gensim: Gensim is used for all kinds of natural language processing work in Python.
3. NLTK: It is also known as Natural Language Tool Kit, and we have used it in the process of text pre-processing.
4. seaborn: a Matplotlib-based visualization package that offers more features for making statistical visualizations.
5. sklearn: It is also known as scikit-learn, it is used for data pre-processing, we have also used it in model selection and evaluation of our model.
6. NumPy: This library is required for mathematical computation, also the part where we have created Matrix, NumPy is used.
7. Matplotlib: This library helps in the visualization part of the project, it can be used in creating interactive, animated, and static visualizations.
8. Pandas: This library helps in the creation of tabular data as well as time series data.

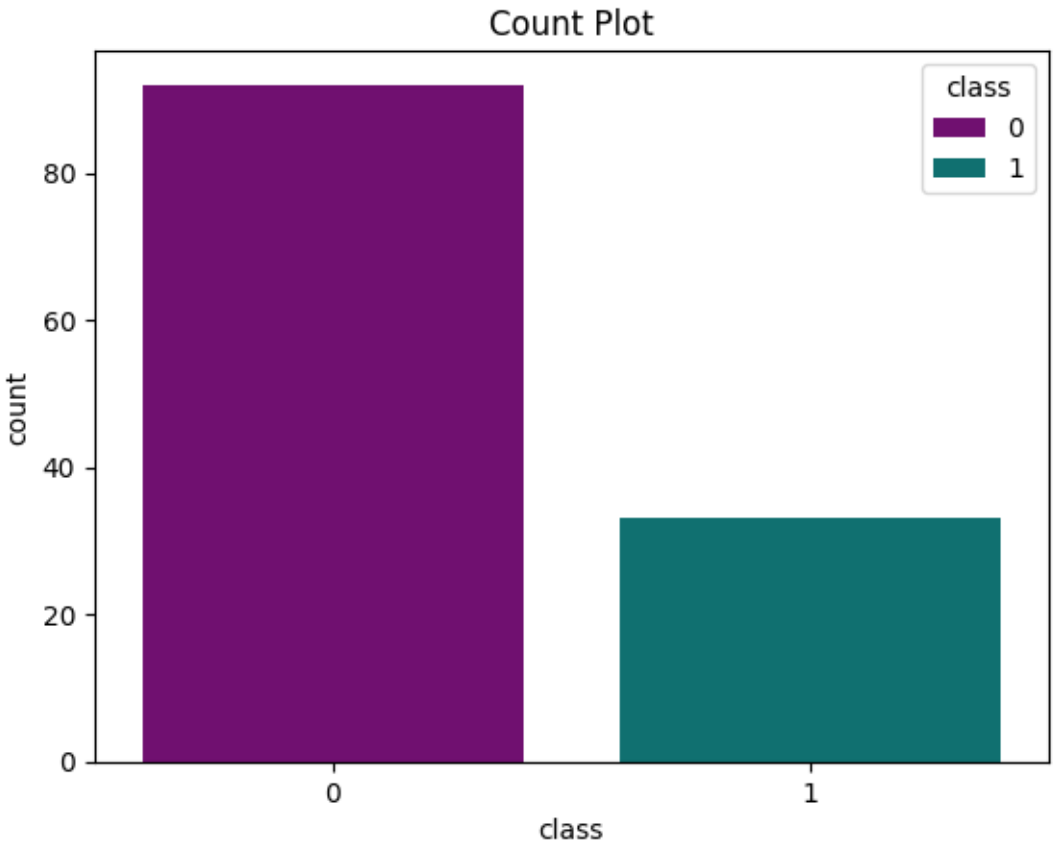
The Dataset used in this project is just one file, which is “resume_data.csv” which consists of 125 rows and 3 columns. The columns are “resume_id”, “class” and “resume_text”. It consists of resumes of several candidates, which are classified as flagged or not_flagged.

3) DATA ANALYSIS AND DATA CLEANING

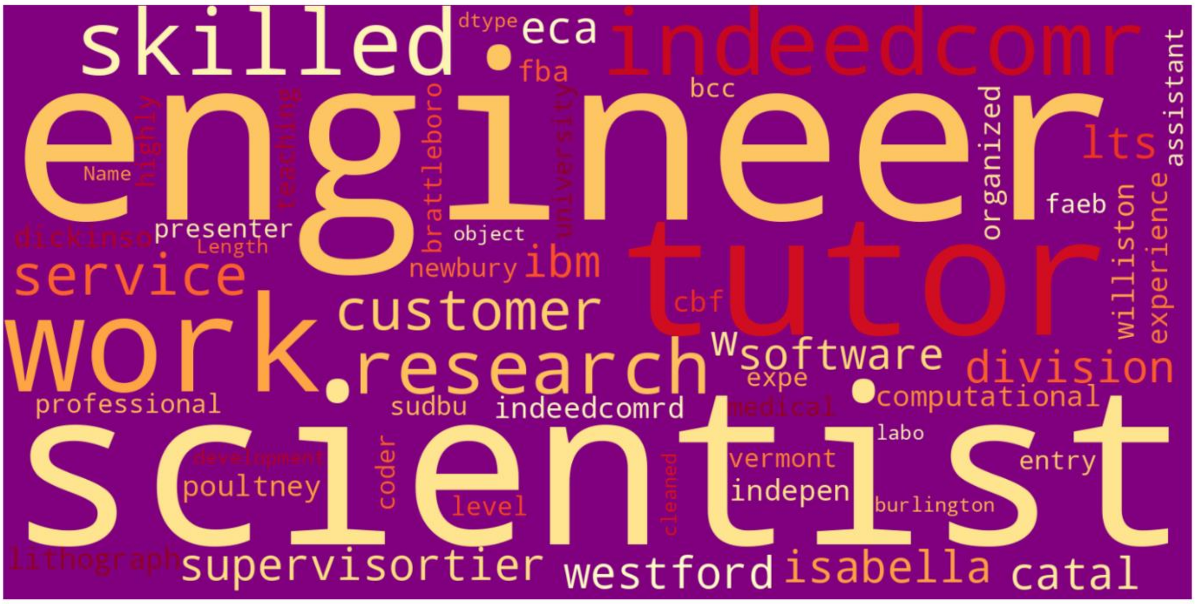
In Data Analysis, we use the info() method to get the complete overview of the DataFrame which consists of data type, memory usage, non-null count, and the number of rows and columns in the DataFrame. We also count the number of occurrences of all unique values in the “class” column. Then we convert the existing “class” column from string to int, for flagged we assign 1 and for not_flagged we assign 0.

In Data Cleaning, firstly we use NLTK library where we use Punkt tokenizer used for sentence segmentation. Stop Words, which consists of words that are not useful while analyzing text data, we also add some additional words in this. The preprocess function returns a cleaned version of the text we previously provided. We also remove carriage returns, all special characters, and words with lengths of less than 3 characters. We use Gensim to tokenize text into individual words. In the end, we join cleaned and pre-processed words into a string.

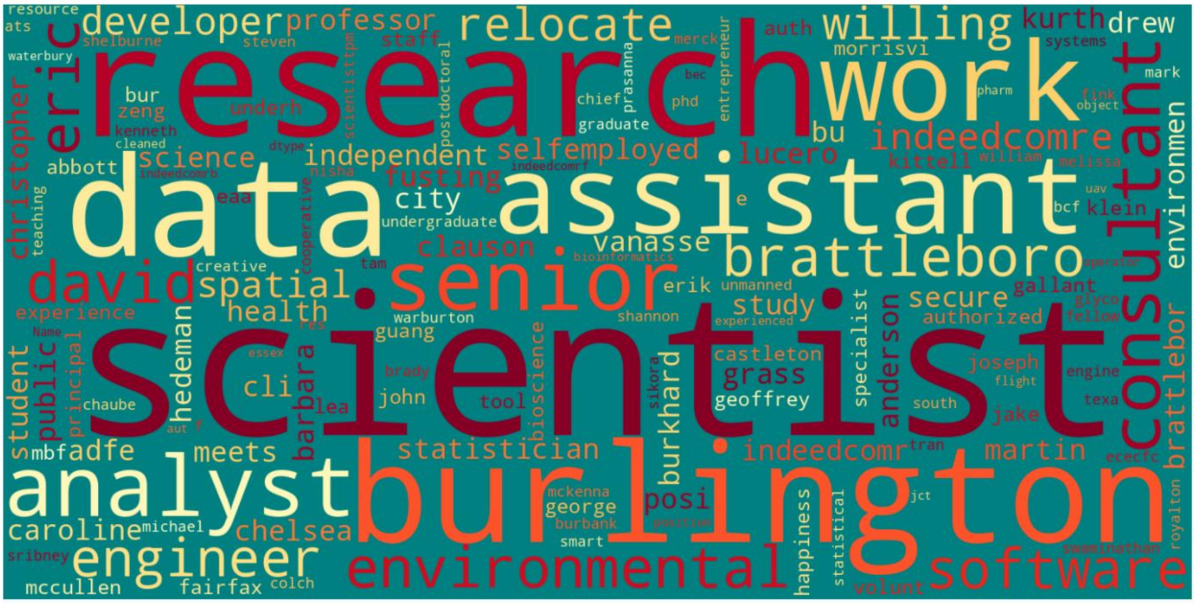
4) VISUALIZATION (GRAPH AND WORDCLOUD)



Here, we use seaborn to create a Count Plot which basically shows a comparison of 2 different groups in categorical variables. Here, it is Class 0 which is not_flagged and Class 1 which is flagged. We can clearly see in the visual representation above that Class 0 has 92 unique values and Class 1 has 33 unique values.



We also created WordCloud for Class 0 which is not_flagged and Class 1 which is flagged. The WordCloud above is for Class 0. Here, we can clearly see that the word size directly corresponds to the frequency of usage of the word. This gives us an idea of what word is widely used in most resumes.



The WordCloud above is for Class 1 which is flagged.

5) PREPARING THE DATA (APPLYING COUNT VECTORIZATION)

Here, we use sklearn's CountVectorizer to transform the text documents into a matrix that consists of token counts or words, fit_transform() is used to create a matrix where each row represents a resume and each column represents a word.

We then display the matrix generated. Each cell in the matrix represents the count of how many times the corresponding word appears in the corresponding resume. We also convert the sparse matrix generated by fit_transform() to a dense matrix by using toarray().

Confusion Matrix

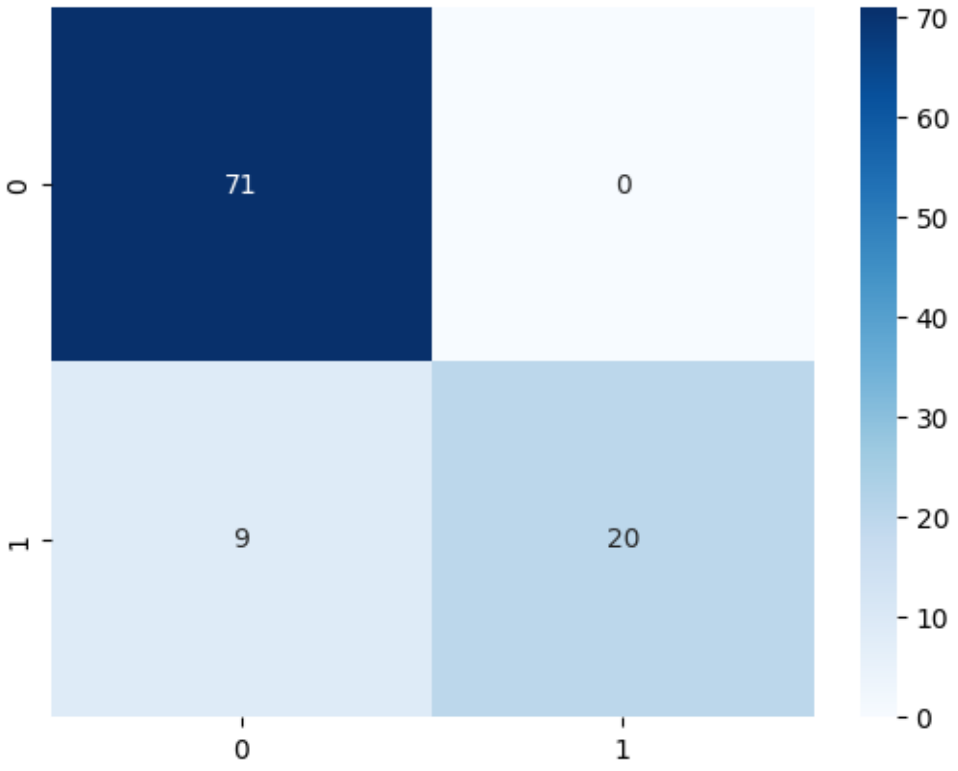
True Positive (TP)	False Positive (FP)
False Negative (FN)	True Negative (TN)

6) TRAINING MODEL (NAÏVE BAYES CLASSIFIER) AND EVALUATING TRAINED NAÏVE BAYES CLASSIFIER

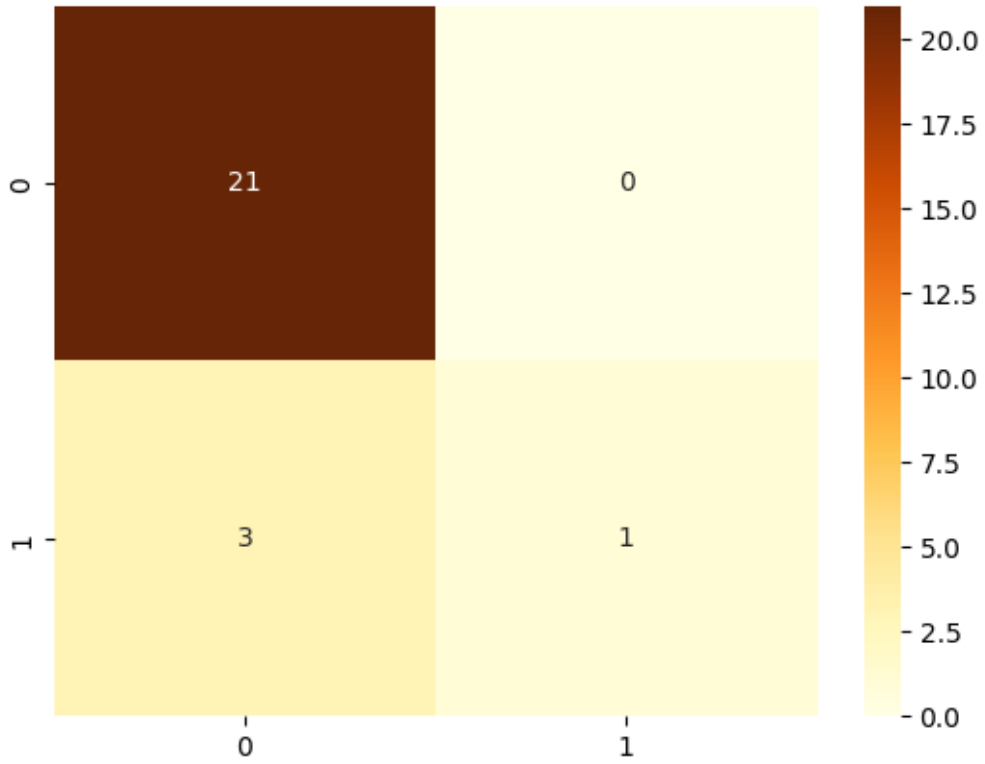
The data is split into training and testing sets in this code using the train_test_split() function of sklearn. The matrix in the previous step is assigned to feature matrix X and each resume's binary categorization labels, which denote whether or not it has been "flagged," are allocated to the target variable y.

The train_test_split() method divides the data into training and testing sets at random, using 80% of the training data and 20% of the testing data. The variables X_train, X_test, y_train, and y_test are allocated to the resultant training and testing sets, accordingly.

The classifier is trained on the training data using the fit() method of the classifier object.



Now we create a confusion matrix for the training set predictions of the Multinomial Naïve Bayes Classifier Model. Using Bayes_clf.predict(X_train) on training data, it predicts the target class. Then a confusion matrix is generated which consists of 4 different outcomes(True Positive, False Positive, True Negative, and False Negative) each corresponding to the 4 boxes in the heatmap shown above.



Here, we calculate the predicted class for the test set. Then a confusion matrix is generated which consists of 4 different outcomes(True Positive, False Positive, True Negative, and False Negative) each corresponding to the 4 boxes in the heatmap shown above.

In the end, we print the classification report, the accuracy score, and the 4 Performance metrics(Accuracy, F1 score, Precision, and Recall).

7) CONCLUSION

In conclusion, a Naive Bayes classifier-based resume selection system can be a useful tool for screening resumes for available positions. The system may be created to automatically choose resumes by gathering a dataset, preparing the data, training the classifier, and assessing its performance. This can save costs and inefficiency while increasing the effectiveness of the hiring process.

8) BIBLIOGRAPHY

- https://en.wikipedia.org/wiki/Naive_Bayes_classifier
- https://towardsdatascience.com/naive-bayes-classifier-81d512f50a7c
- https://www.geeksforgeeks.org/naive-bayes-classifiers/
- https://jakevdp.github.io/PythonDataScienceHandbook/05.05-naive-bayes.html
- https://www.youtube.com/
- https://stackoverflow.com/
- https://www.google.com/
- https://www.kaggle.com/