**Restaurant Rating Application**

**Technical Appendix**

Contents

[Technical Appendix 2](#_Toc37333261)

[Python Libraries Used 3](#_Toc37333262)

[Data Cleaning 3](#_Toc37333263)

[Data Transformation 4](#_Toc37333264)

[**Algorithm Used** 5](#_Toc37333265)

[NLTK 6](#_Toc37333266)

# 

# Python Libraries Used

1. **Numpy:** Numpy is used for math calculation and scientific computing package.
2. **Pandas:** Pandas is used to manipulate and analyze data. It was used for:

* Inserting and deleting columns in data structures.
* Merging and joining data sets.
* Reshaping and pivoting data sets.
* Aligning data and dealing with missing data.
* Reading from files with CSV, XLSX, TXT, among other formats.
* Arranging data in an order ascending or descending.

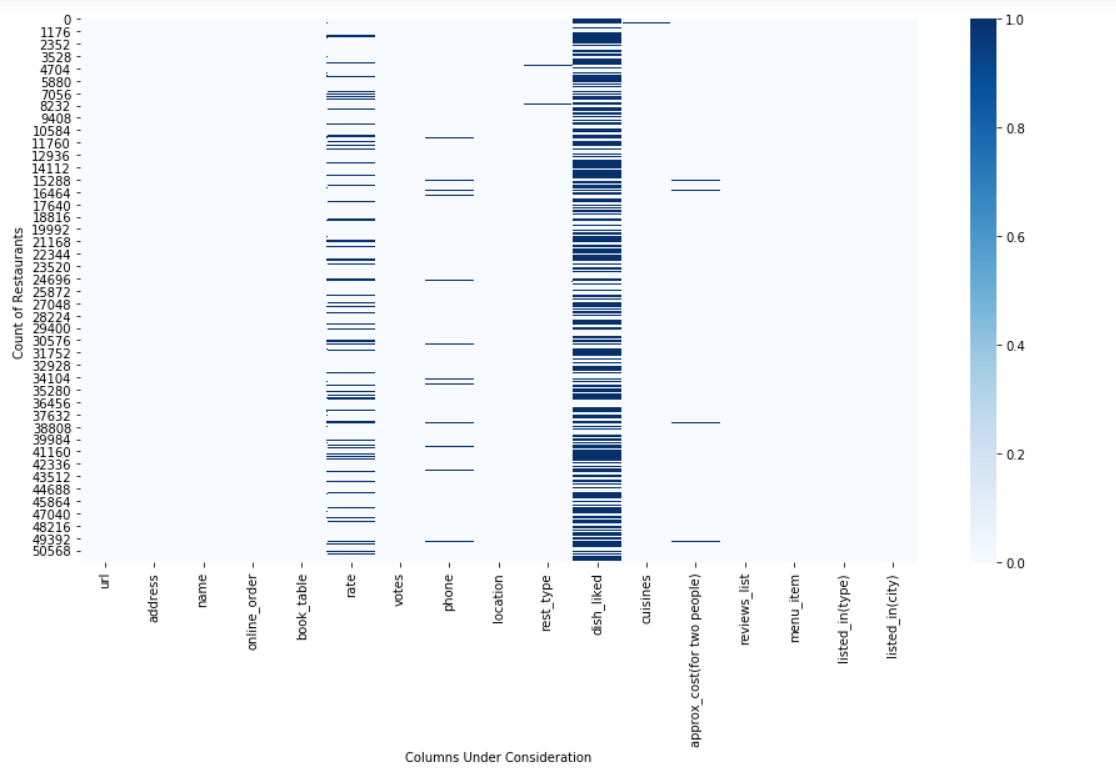
1. **Seaborn:** Seaborn was used for data visualization and creation of informative statistical graphics.
2. **Matplotlib:** Matplotlib was also used for plotting visualizations.
3. **Scikit-learn:** Scikit-learn was used for performing many unsupervised and supervised machine learning algorithms. We used the below functions of this library.

* **Preprocessing:** We did normalization by importing scale from Preprocessing functionality.
* **Linear\_Model :** We did Linear and Logistic Regression by importing Linear Regression and Logistic Regression form Linear\_Model functionality.
* **Model\_Selection :** The data was split into train and test by importing train\_test\_split from Model\_Selection functionality.
* **Metrics:** Various metrics like Classification Report, Confusion Matrix, Mean Squared Error, Silhouette Score and R2 Score were imported from Metrics functionality.
* **Tree:** Decision Tree Regression was performed by importing DecisonTreeRegressor from scikit-learn Tree functionality.
* **Ensemble:** We performed Random Forest Regression by importing RandomForestRegressor from scikit-learn Ensemble functionality.

1. **NLTK:** NLTK was used for Natural Language Processing. The below functions were used from this library. We imported FreqDist, bigrams, trigrams from NLTK and Stopwords and RegexpTokenizer from NLTK\_Corpus.
2. **WordCloud:** WordCloud was used to generate WordCloud image visualizations of most popular words.
3. **OS:** OS library was used to import path required to get the WordCloud images.
4. **PIL:** Python Imaging Library (PIL) was used to import Image to save the image in different file formats.

# Data Cleaning

* Heatmap was created of null data. Null values are indicated by dark blue colour.



* Dropping of unnecessary columns. i.e. 'url','address','dish\_liked','phone'
* Duplicate rows were dropped. We had 63 duplicate rows.
* NaN values were dropped from the dataset.
* Renaming of column names.

approx\_cost(for two people) 🡪 cost

listed\_in(type) 🡪 type

listed\_in(city) 🡪 city

# Data Transformation

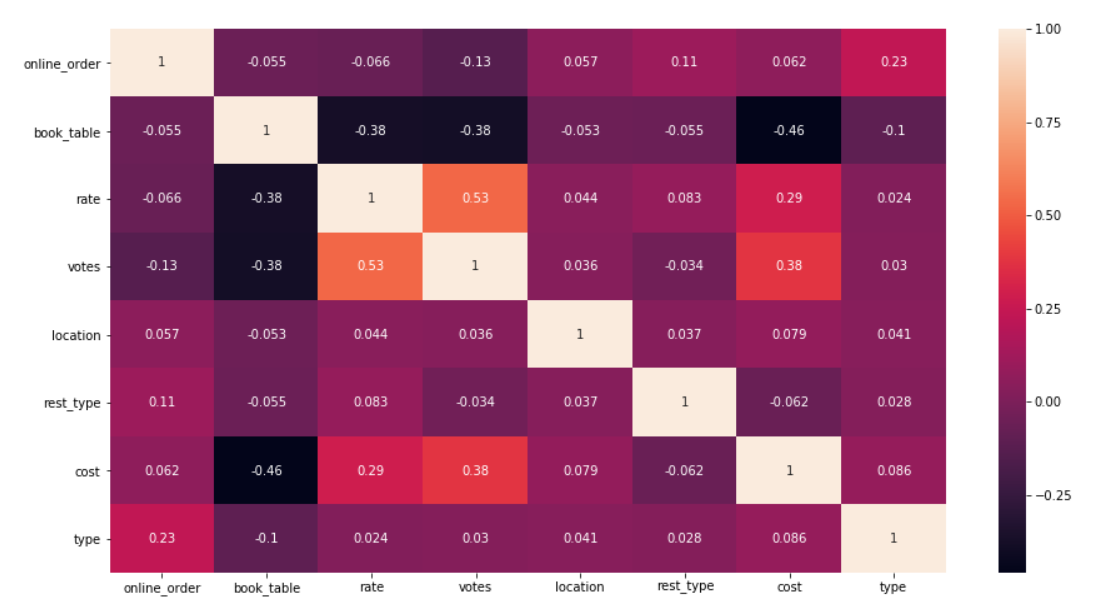
* Cost column was converted to string to remove the comma (,) by replacing it. Then cost was changed to float as original. For example, 1,456 was changed to 1456.
* Rate column had rating with a slash 5 i.e. 4/5, 3/5. Slash 5 was removed using lambda expression.
* Online order and book table column values were converted from Yes/No to Boolean True/False
* Name column values were changed to title case using lambda expression.

**Encoding Variables**

|  |  |  |
| --- | --- | --- |
| **Column** | **Original Value** | **Encoded Value** |
| Online Order | True, False | 1,0 |
| Book Table | True, False | 1,0 |
| Type | Buffet, Cafes, Delivery, Desserts, Dine-out, Drinks & nightlife, Pubs and bars | 0,1,2,3,4,5,6 |
| City | Banashankari, Bannerghatta Road, Basavanagudi, Bellandur, Brigade Road,Residency Road, Sarjapur Road | 0,1,2,3, 4…….,27,28 |

**Correlation between variables**

Heatmap was generated to find correlation between different variables.



From the above plot, we can see that there is positive correlation between rate and votes (0.53) meaning that with the increase of votes, the rating of the restaurant increases.

# Algorithm Used

Our Restaurant Rating App (RRA) uses **Random Forest Algorithm** to predict the rating of the restaurant.

**Random Forest Python Code:**

from sklearn.ensemble import RandomForestRegressor

RForest=RandomForestRegressor(n\_estimators=500, random\_state=329,min\_samples\_leaf=.0001)

RForest.fit(x\_train,y\_train)

y\_predict=RForest.predict(x\_test)

print ('R-squared score of the Random Forest model: ', r2\_score (y\_test, y\_predict))

print ('MSE of the Random Forest model: ', mean\_squared\_error(y\_test, y\_predict))

**Output:**

R-squared score of the Random Forest model: 0.8061121020287574

MSE of the Random Forest model: 0.03828193179600166

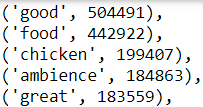
**Interpretation:**

The above Random Forest output depicts that the accuracy of the algorithm is 0.806 or 80.6%. Also, the Mean Square Error of the model is 0.03828 which is close to zero making it a good fit model.

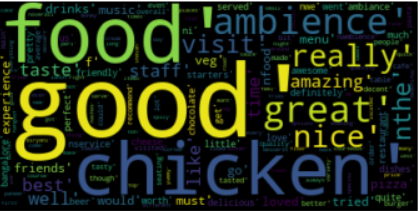
# NLTK

The below steps were done to the data using NLTK.

* Converting all the text in ReviewList column to lowercase.
* Creating a regular expression tokenizer that matches only alphabets.
* Applying the tokenizer to each row of the reviews.
* Removing stopwords i.e. unwanted words like is, and, the etc. from the tokens.
* Getting the frequency distribution of individual words in the review list column using FreqDist() function.
* Examining the top 5 most frequent words. The output below indicates that the word ‘good’ was the most frequent and it came 504491 times followed by food and chicken.



* Finding Nouns, Adjectives, Verbs and Pronouns using Parts of Speech (POS) Tagging.
* Create and generate a word cloud image. The below image contains all the frequently used words in the review list column. The most common words are big in size. i.e. good, food, chicken etc.



* Creation of wordcloud wine image. The most common words in review list are placed in a wine image below.

