Predicting Rating for New Restaurants on Zomato

Machine Learning Capstone Project - Udacity

Shreyas Dhuliya

1. Definition

1.1 Project Overview

Zomato is a restaurant search and discovery website/application which operates in many major cities spread across 24 countries.

Zomato helps land customers over the restaurants through many search options based on restaurants location, restaurant type, cuisines they serve and helps discover better using filters and sort by options for advance searches.

Bangalore, India is continuously urbanization and stretching its boundaries, the city has more than 8,000 different restaurants, bar, pubs, cafes and has potential to have more successful running food restaurants.

Currently **51,000 restaurants** are spread across **30 listed location in Bengaluru**. Which has 2,000 new restaurants listed on Zomato. The Dataset is in csv file provided in Kaggle by Himanshu Poddar All copyrights for the data is owned by Zomato Media Pvt. Ltd. And the data is for education purpose only. (data repository).

1.2 Problem Statement

The overall rating on Zomato for a restaurant is not the average rating given by the customers. The rating is relative to other restaurants in the same city. Zomato has many algorithms to determine the final rating which can change with performance of other restaurants near by as well. **The final ratings all the restaurants of a city is placed on a normal curve. Which makes them relative to each other.** .(Zomato rating system)

Some known factors are:

- *Frequency of ratings/reviews by customers*
- *Rates/reviews of other nearest restaurants*

Other factors are:

- *Location of the restaurant*
- *Cuisines served*
- *Demography of the location*
- *Services of the restaurant – online service, cuisines*
- *theme of the restaurant - Café, restaurant, pub, bar*
- *category of the food served.*

The final rating for the restaurant is in between 1 – 5 with interval of .1. The ratings can be like 1,1.1,1.2,…,4.8,4.9 or 5 for the overall rating.

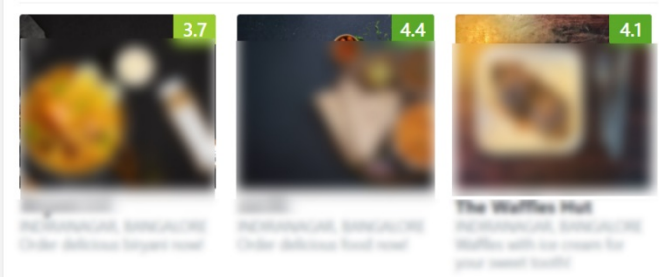


Figure 1: Three Restaurants and their Ratings on Zomato

Src: zomato.com/bangalore/

The rating is color coded between 1-5 with interval .5

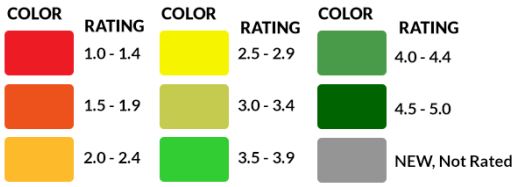


Fig. 2. Color codes for ratings – color code selected based on nearest color.

- *What is the Aim of the Project?*

For this project the ratings are classification problem with ratings falling in 8 different color bins as shown in figure 2. The 8 color bins are ratings with bin size of interval .5 between 1-5.

The Aim of this project is to make a Deep Learning Neural Network model which predicts the color bin of the

NEW restaurant it will fall in the future. There are 8 classes for prediction.

Figure 3: New Restaurant on Zomato

Src: zomato.com/bangalore/

- *What are some predicating models?*

A linear regression model can be used for predicting the rating. It is the simplest model which has been used to predict rating for wine quality and movie rating that assumes the problem is linearly defined.

1.3 Metrics

We will create a new target column which stores label for predicting the eight bins. The function target_rates_to_color(ratings) between 1-5 with interval .1 example "ratings = 3.4" and returns an integer label 4

Label	COLOR	RATING	Label	COLOR	RATING	Label	COLOR	RATING
0		1.0 - 1.4	3		2.5 - 2.9	6		4.0 - 4.4
1		1.5 - 1.9	4		3.0 - 3.4	7		4.5 - 5.0
2		2.0 - 2.4	5		3.5 - 3.9			To predict NEW, Not Rated

Figure 4 Color bins and their labels

1. Color category

Accuracy – calculating the accuracy of the model for predicting the color class the restaurant will fall into i.e categories defined with an interval of .5.

$$Accuracy = \frac{Correctly\ predicted\ Class}{Total\ Number\ of\ Predictions}$$

2. Analysis

2.1 Data Exploration

Dataset contains information of 51,717(8,792 unique) restaurants in Bengaluru city by 15th March 2019. It contains 16 features and one target column for each restaurant.

- *About the columns*

1. *url(object)* – This is simply a string with tells the URL which is almost common for all as it is starting with Zomato.com/xyz/abc.
2. *Address(object)* – Complete address of the restaurant. Contains the full address which is not required for the analysis, can be deleted.
3. *Name(object)* – Name of the restaurant, The name is unique and is different languages such as Gabar(Hindi Word), Asha(Hindi word), Twist (Englih) – it will be difficult to categories 51,717 values in the ANme column. For these reasons the column can be dropped.
4. *Online order(Object)* – Yes or No, useful for predicting by converting it to binary.
5. *Book Table(Object)* – Restaurant provides online table booking service (yes or no), useful for predicting by converting it to binary.
6. *Rate* **(Target)–** Overall rating of the restaurant between" 1 – 5" upto one decimal point example 4.2. "New" for new restaurants.
7. *Votes(int)* – Total number of reviews/votes, not useful for predicting the target variable, needs to be dropped.
8. *Phone(object)* – Phone number of the restaurant. The counts of phone numbers provided could help in predicting target.
9. *Location(object)* – Neighborhood of the Restaurant in Bengaluru, important for predicting rating. One- hot encode to change it to dummies.
10. *Rest_type(object)* – Type of the restaurant, important for predicting rating. One- hot encode to change it to dummies.
11. *Dish_liked(object)* – Dish liked by people in the restaurant,
12. *Cuisines(object)* – Cuisines types in the re restaurants. Types are comma separated
13. *Approx_cost(object)* – expected cost for meal of two in Rupees, important for prediction

14. *Reviews_list(object)* - list of tuples containing reviews for the restaurant, each tuple consists of two values, rating and review by the customer. Needs to be dropped for the prediction for new restaurant.
15. *Menu_item(Object)* – List of menus available in restaurants. Contains more than 54% missing values, needs to be dropped.
16. *Listed_in(type)* - type of meal
17. Listed_in(city) - contains the neighborhood in which the restaurant is listed - duplicate column can be dropped

- *Missing in each columns*

775 ratings, 1208 phone numbers, 21 locations, 28,078 liked dishes are missing, 27 restaurants

Figure 4:

The below Picture shows the basic statistics of columns such as rate, phone, cost for two, Online order and book table.

Figure 5:

- small correlation can be seen between rates and cost_for_two which was expected
- a positive correlation can be seen between booking table and rates.

Please Note: The below tables were created using df.describe() after processing the data for 51,717 to 41,660 rows which will be used to train the model.

	rate	phone	cost_for_two	online_order_no	online_order_yes	book_table_no	book_table_yes
count	41660.000000	41660.000000	41418.000000	41660.000000	41660.000000	41660.000000	41660.000000
mean	3.700504	1.076092	603.268048	0.346952	0.653048	0.848680	0.151320
std	0.440468	0.660766	464.327857	0.476006	0.476006	0.358365	0.358365
min	1.800000	0.000000	40.000000	0.000000	0.000000	0.000000	0.000000
25%	3.400000	1.000000	300.000000	0.000000	0.000000	1.000000	0.000000
50%	3.700000	1.000000	500.000000	0.000000	1.000000	1.000000	0.000000
75%	4.000000	2.000000	700.000000	1.000000	1.000000	1.000000	0.000000

Figure 5: Description of zomato bengaluru dataset after preprocessing

	rate	phone	cost_for_two	online_order_no	online_order_yes	book_table_no	book_table_yes
rate	1.000000	-0.039605	0.385183	-0.068326	0.068326	-0.424651	0.424651
phone	-0.039605	1.000000	-0.007960	-0.048984	0.048984	-0.004695	0.004695
cost_for_two	0.385183	-0.007960	1.000000	0.175517	-0.175517	-0.614338	0.614338
online_order_no	-0.068326	-0.048984	0.175517	1.000000	-1.000000	-0.051055	0.051055
online_order_yes	0.068326	0.048984	-0.175517	-1.000000	1.000000	0.051055	-0.051055
book_table_no	-0.424651	-0.004695	-0.614338	-0.051055	0.051055	1.000000	-1.000000
book_table_yes	0.424651	0.004695	0.614338	0.051055	-0.051055	-1.000000	1.000000

Figure 6: Correlation of columns after a stage of preprocessing

2.2 Exploratory Visualization

2.2.1 Bar Plots of Online Order, Booking table columns

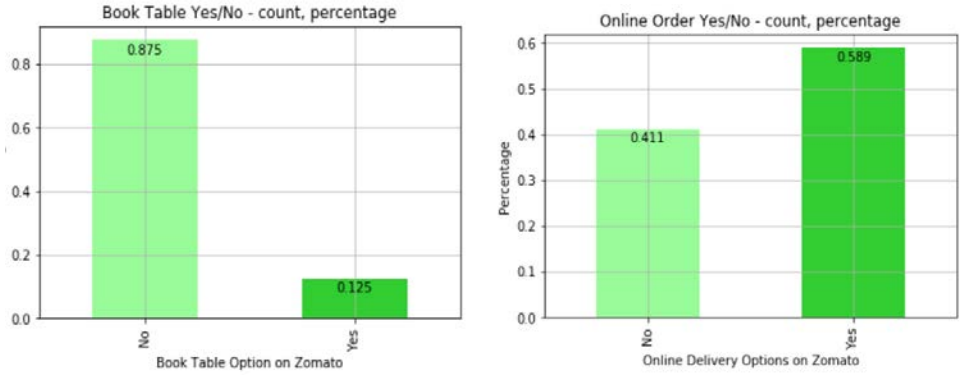


Figure 7: Ratios of restaurants having online table booking service and Online delivery service

Inference

Left :

Out of 51,717 records 87.5% of restaurants do not have options to book table online using Zomato.

Right:

Out of 51,717 records 41.1% of restaurants do not have options to deliver using Zomato application

2.2.2 Normal Curve for Ratings of all Restaurants in Bengaluru

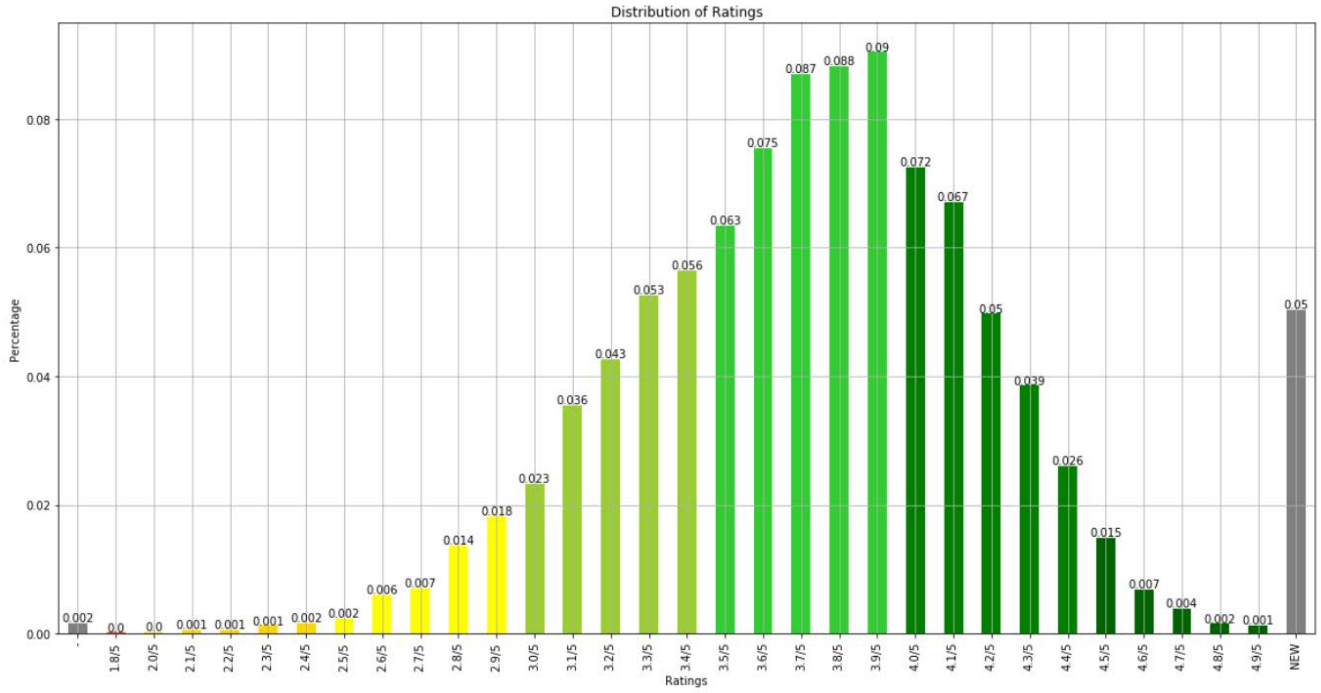


Figure 8: Ratings ratios bar graph

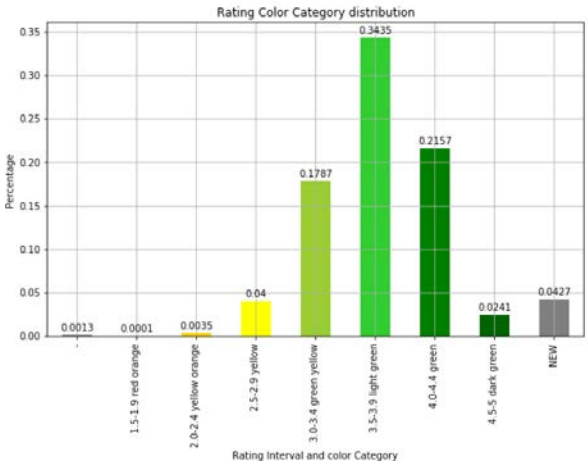


Figure 9: Bins and the ratios

Color Bin Ratios

- 34.35% of the restaurants in Bengaluru have ratings between 3.5 - 3.9.

- 21.57% of the restaurants have ratings between 4.0 - 4.4.

- Only 2.41% of the restaurants have ratings between 4.5 - 5.

- Only 23.98% of the restaurant falls on the right of the curve

Normal Curve

The rating count is fitted inside a normal curve centered around 3.8 rating. .2% of ratings are not given yet to the Restaurants and 5% of the restaurants are new on Zomato.

2.2.3 Restaurant Listed in City in Bengaluru

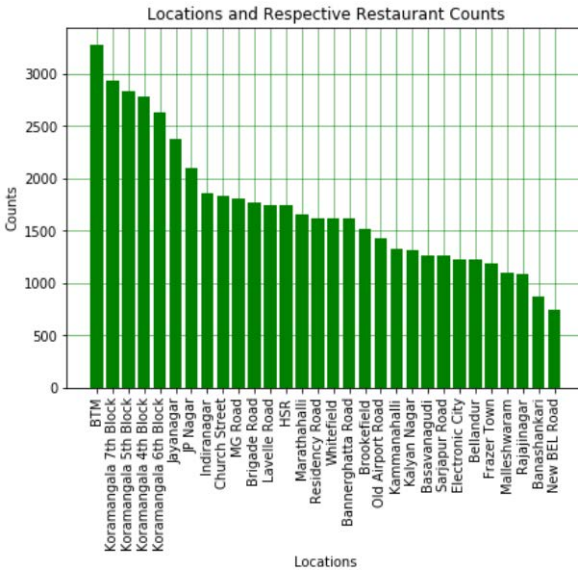


Figure 10: Locations and restaurant counts

Zomato has divided Bengaluru into 30 Zones/locations in the city which all the restaurants are a part of. The bar plot shows the restaurants counts in the 30 parts of the city. This is for quick searches on the front page.

2.2.4 Restaurant types listed on Zomato

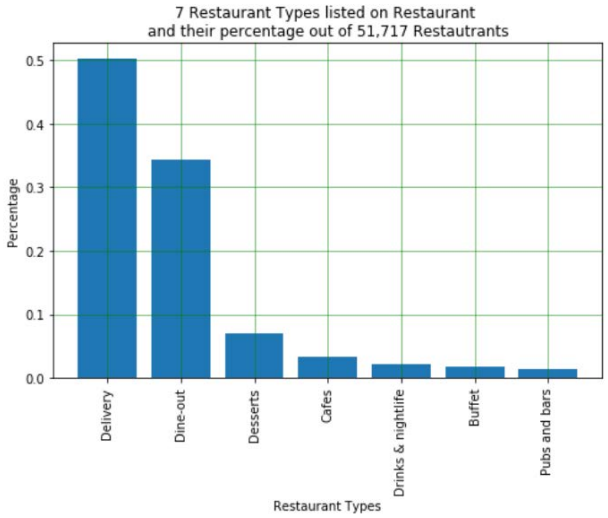
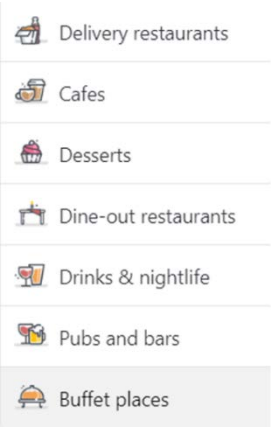


Figure 11: 7 Restaurant types for quick search and their counts



listed_in(type) column

Zomato has listed all the restaurants in 7 main restaurants categories for quick searches. The image shows the options which can be seen on website and application

rest_type column

This is a mixed categorical columns with restaurant type s which are comma separated.

Restaurants have listed them on Zomato differently. In rest_type Restraunt type is listed with 30 unique types as shown in the graph below.

Each restaurant can have 2 types or 1. The column can have “Casual Dining, Cafe” as restaurant types or “Bar”.

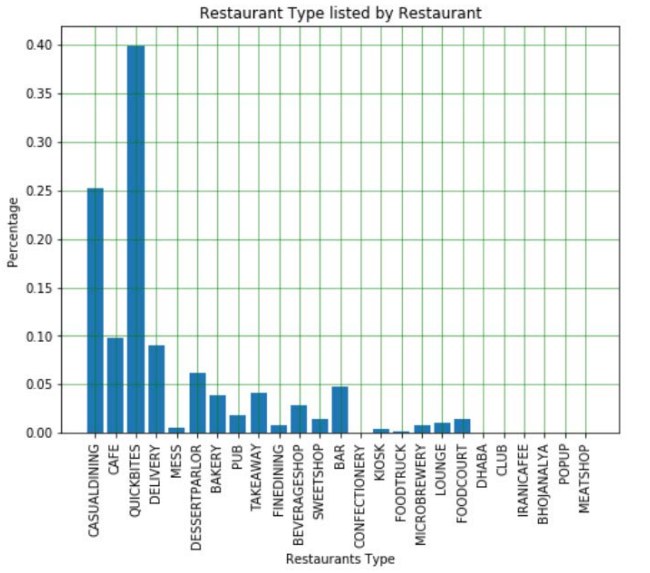


Figure 12 Types mentioned by Restaurants and their counts

2.2.5 Cuisines

This is a mixed categorical columns with cuisine names which are comma separated.

Each restaurant have one or more cuisines which are comma separated if they are more than one example “North Indain, Chinese, South Indian”. The graph below shows the count of restaurants with 1 or more cuisines.

Example 17500 restaurants have 2 cuisines which they serve.

Figure 13: Counts of Restaurant with 1 or more cuisines

There are 107 unique cuisines served. Each restaurant serves minimum of 1 cuisine and maximum of 8. NorthIndian, chinease, southIndian and fastfood is the most popular. Malwani, Panasian, hotdogs,sindhi, australian and few more cusines are the least served cuisines with only 2-10 places serving them.

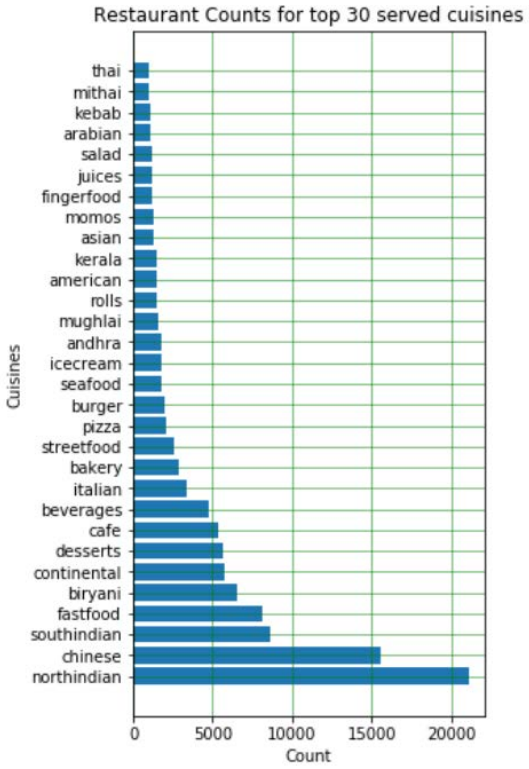


Figure 14: Restaurant Counts for top 30 Cuisines served in Bengaluru

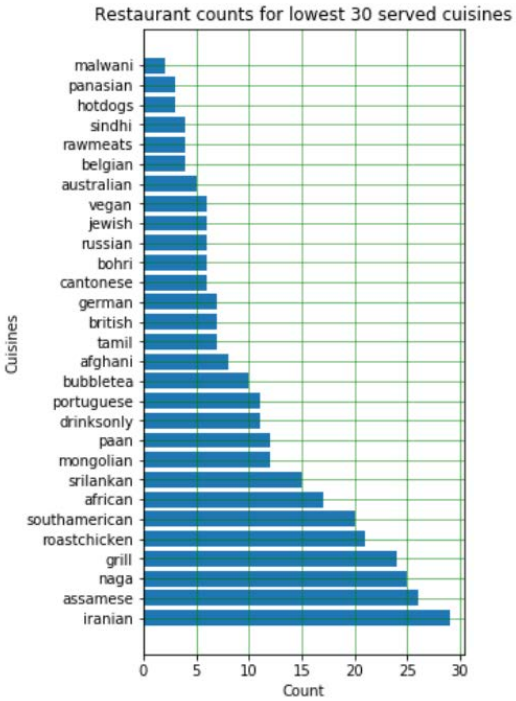


Figure 15: Restaurant counts for lowest 30 Cuisines served in Bengaluru

2.3 Algorithms and Techniques

The Model is a Neural network with 307 inputs and 3 hidden layers and 8 categories/bins:

- The neural Network Architechture is created using torch nn module, which has 8 output. Log_softmax is performed to get the label.
- nn.dropout(.02) is used so that the network does not over fit.
- nn.NLLLoss is used to calcuate the loss. since the output is a softmax which is in the form of exponential, Natural Log loss will be best to get the loss
- loss.backward() performs the backward pass
- optimizer.step() updates the new weight
- optimizer.zero_grad() sets the grdient to zero for next backword propagation
- model.eval() freezes the gradient and removes the dropout for evaluation/validation during training

Training parameters for the Neural Network:

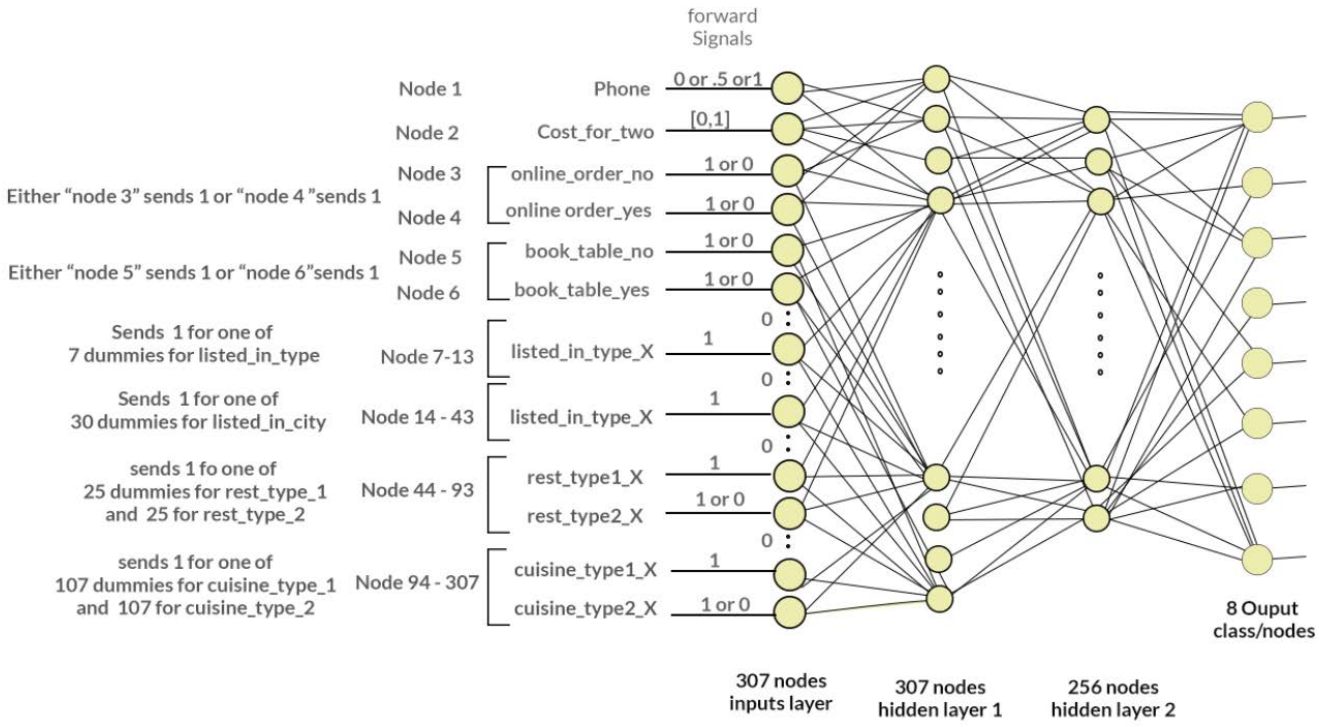
- Epochs
- Learning rate
- Mini Batch size
- Number of Hidden layers
- Number of Hidden nodes

Feed forward

  - Relu activation
  - Softmax function
  - NNLoss
  - Drop out
- Back propagation
  - Adam optimizer
  - Weight updation
- Validation
  - Model evaluation
  - Validation loss
  - Freezing grad
- Torch.optim Adam optimizer

The Neural network structure is shown in the next page.

NN Architecture



2.4 Benchmark

For this problem the data will be split into training and testing sets with test sample size of 5% using train_test_split. This randomly splits the dataset. 95% of the data will be used to train the benchmark model and the Neural Network model. 5% of the data will be reserved to test the scores of the final models, linear model and optimized NN model for comparison.

The project's benchmark model is going to be a linear regression model.

y_cap= Wx + b

$W = (W_1, W_2, \ldots, W_n)$ $x = (x_1, x_2, \ldots, x_n)$

b = intercept

Linear model is underfitting because it is assuming the problem is linear and that the prediction follows a linear equation. It also gives a prediction as an output and does not say what is the probability of getting that prediction. The linear regression classifier does not work well with binary inputs as it multiplies the weights with zero, this provides less information during training. This model has been used to predict wine quality rating, Movie ratings and other ratings.

Neural Network on the other hand is a non linear model and is expected to perform better than the linear model. There are 41 possibilities for the prediction from range 1 to 5 with interval of .1. So we will train the neural network as a classification problem to provide the probability of the output along with other n-best probabilities. The total number of output nodes will be 41, each telling the probability of the prediction.

Finally the score of optimized NN model should be better than Linear model on the same test dataset. The score of the linear model will be the threshold for the NN model to cross.

3. Methodology

3.1 Data Preprocessing

3.1.1 Assessing columns

i) Renaming columns

- approx_cost(for two people) to cost_for_two
- listed_in(type) to listed_in_type
- listed_in(city) to listed_in_city

ii) Assessing columnms

- Visualizing missing counts per columns

iii) Dropping Unwanted columns

- **The URL**
- **Name** - most of the names have their short address with their names. The number of words can not be used to analyse the sdata
- **Address** - The locality and city in Bengalure is mentioned in another column.
- **Votes** - The new restaurants will not have any prior vote counts
- **dish_liked** - More than 50% of the data is missing, hence can not be used for nanylysis and machine learning
- **menu_item** - More than 75% of data is not provided, This column can not be used for analysis or prediction
- **review_list** - The new restautrant won't have a review list as it is a new restaurant on Zomato

3.1.2 Assessing row missing

i) Removing rows with missing ratings and "-" no rating provided.

ii) Replacing missing phone values with "0" indicating it is not provided. Assuming missing phone numbers are not provided.

iii)Removing rows with more than 1 missign values

3.1.3 Pre- processing columns

i) Non Numerical categorical columns,

- online_order - yes or no for oline order booking
- book_table - yes or no for table booking

Created dummy variables for both the columns which gave 4 columns with binary values. Few restaurants types don't have tables example few quick bites, cafes don't really need booking tables hence did not binaries the columns to 1 for yes and 0 for No.

ii) Numerical, float in form of string columns

- Rate - 1,1.1,1.2,....,4.9,5

Ratings are given in the form of a string "1.5 /5". A function is defined in helper.py which removes spaces separates rates where"/" and returns 1.5 in form of float.

- phone
  - 0 for no phone numbers
  - 1 for one phonr number provided
  - 2 for two phone number provided

Phones are in the form of str "+91 9865745477\n\r 080 67577578" which are either 8 digit for landline or 10 digits for mobile.

A function is created which checks for continuous 8 numerical numbers after removing any space in the string and returns the count of phone numbers 0 for missing phone numbers,1 or 2.

- cost_for_two - Approx. cost for two in the restaurants

Cost for two is given in string with commas "1,700". A function on the column removes comma and returns the integer value example "1,700" to 1700.

iv) Multi level categorical columns

- listed_in_city - part of the city in Bengaluru eg Church Street, BTM - 30 unique names

The part of the city the restaurant belongs to in Bengaluru. Zomato has broken down the areas in the city into 30 parts or zones/locations. This can be seen in the bottom of the index page when Bengaluru is selected.

Created dummies using one hot encoding to get 29 more columns containing binary information of the restaurant location.

- location - nearest location of therestaurant in the city eg. Marathahalli, Shivajinagar - 93 unique location

The closest location from the restaurant. There are 93 unique locations/streets. Most of them are listed in the listed_in_city column. For the ease of computing this column is dropped.

- listed_in_type - Type of the restaurant on zomato - 7 unqiue types.

Created dummy columns for this multilevel nominal categorical column.

iv) Mixed Nominal Categorical Columns

- rest_type - restaurant type as mentioned by the restaurant - few have two types
- cuisines - cuisines type as menstioned by the restaurant - restaurant may have cuisines type between 1-8.

For rest_type two columns are created

rest_type
Cafe,Casual Dining

rest_type1	rest_type2
cafe	casualdining

Dummies were created using for loop to make 25 dummy columns for rest_type1 and 25 dummy columns for rest_type2. Which stores 1 for one the 25 columns if that rest_type is there else 0. As there are 25 unique restaurants types 25 dummies were created for each.50 columns in total

For cuisines 2 columns are clreated which stores the first two cuisines

cuisines
North Indian, Fast Food, South Indian

cuisines_type1	cuisines_type2
northindian	fastfood

Dummies were created using for loop to make 107 dummy columns for cuisine_type1 and 107 dummy columns for cuisine_type2. Which stores 1 for one the 107 columns if that cuisine is there else 0. As there are 107 unique restaurants types 107 dummies were created for each. 214 columns in total.

3.2 Implementation

1. Training the model - Train.py has function *create_train_model()* which taxes in X_train and y_train parameters, where X_train are the features and y_train is the target column. It uses *iterate_minibatches() to get mini_batches for training and validation during the training.*

There are total 39,577 rows to train the model.The rows are broken down into 4 mini batches of 9,900 with last batch having rows 9,877. 3 mini batches are used for training and 1 is used for validation and printing training loss, validation loss and validation accuracy of each epoch. For each epoch the model is trained 3 times and validated 1 time.

Fixed on the epochs count to 400 after getting an almost flat validation curve.

2. Testing the accuracy of the model with the test datasets X_test and y_test. The accuracy is calculated with sklearn.metrics accuracy_score.

3. Processing the NEW restaurants- predict.py dataframe containing information for NEW restaurants using process() function which returns the processed data having 307 columns.

4. Predict a Row – predict.py A row is predicted with index number which returns the top three categories with the probability of those category.

3.3 Refinement

Problems faced during creating mini batch function:

i) Creating Mini batch function – train.py

```
1 indices = np.arange(X_train_t.shape[0])
2 for e in range(epochs):
3     np.random.shuffle(indices)
4 for start_ind in range(0, X.shape[0], mini_batchsize):
5     end_ind = min(start_ind + mini_batchsize, X.shape[0])
6     new_in = indices[start_ind:end_ind]
7     yield X[new_in], y[new_in]
```

Line 1 takes the total number of rows in the training set and creates an 1D array of indices

Line 2- 3 for each epoch : start by shuffling the indices

Line 4 this is in the function iterate_mini which takes X_train, y_test, numbr of rows in a minibatch and shuffled indices. It produces the start index of the mini batch and stores in start_ind.

Line 5 produce end index by adding start index with the number of rows in a mini bacth or if the end of indexes is reached, take the last index. This is done using minimum of both to avoid getting error : out of index.

Line 6-7 using start and end index value yield the rows from X_train and y_train.

The next time the function is called it calls the next set of start and end index till end of index is found.

This algorithm was formed by taking inspiration from stack over flow. https://stackoverflow.com/questions/38157972/how-to-implement-mini-batch-gradient-descent-in-python

Building the right Architecture

For arriving at this I have experimented with architecture having 1,2 or three hidden layers. The nodes in the hidden layers where experimented with before reaching to 2 hidden layers for the final architecture.

Using the right number of epochs and mini batches

The whole dataset was broken down into multiple parts for each epoch, into mini batches of 2(19,800 rows each), 3(13,200 rows each), 4(9,900 rows each), mini batches with 500 rows, 1000 rows.

Val_print takes integer n value to validate the model every nth mini_batch.

4 Results

4.1 Model Evaluation and validation

- the NN optimized model has an accuracy of 76%

- The NN model takes in 307 inputs out of which 1 node i for phone count, 1 for cost for two, 2 for yes/no online order, 2 for yes/no book table option, 7 for rest_type, 30 for location listed in, 50 for restaurant type1 and type2, 214 nodes for cuisines type 1 and type 2.

- The NN model gives out best three classes with there probabilities

-The model is capable of more learning as the validation loss(green) is still below training(loss) which shows the model and be trained more. The model is trained for 460 epochs with 4 mini batches out of 39,577 rows. i.e 9,900 rows for each mini batch.

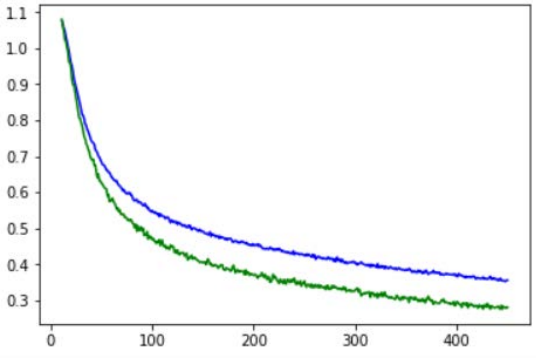


Figure: Training and validation losses

4.2 Justification

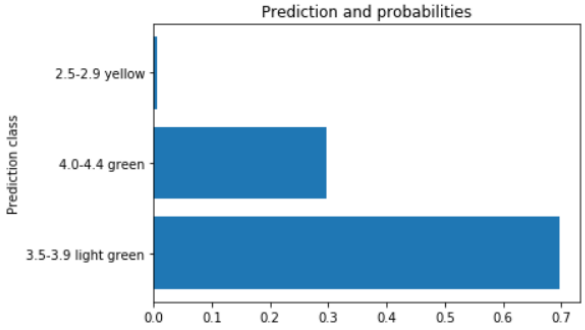
The model is able to predict with 78% of accuracy. The model gives out best 3 classes along with their probability. The image below shows the prediction for the NEW restaurant at 2110.

Predicting for selected restaurant index

```
In [146]: print("The Top 3 Classes for the restaurant below with probabilities are:")
          display(bengaluru_res_df_new[index_:index_+1])
          pr.predict(model,X_predict_t)
```

```
The Top 3 Classes for the restaurant below with probabilities are:
```

	online_order	book_table	rate	phone	location	rest_type	cuisines	cost_for_two	listed_in_type	listed_in_city
48206	Yes	No	NEW	+91 8043774001	MG Road	Casual Dining	Biryani, Chinese, North Indian, South Indian	800	Dine-out	Residency Road



5 Conclusion

5.1 Reflection

Limitations:

- The model can predict scores only for the Restaurants provided in csv and have rate value as "NEW"
- Assuming that missing numbers as count 0 i.e no phone number provided for restaurant
- 346 restaurants have cost for two missing which is replaced by median
- The model is only limited to predict Bengaluru Data
- Due to computation time only two columns were prepared from cuisines to store first two in "cuisine_type1" and cuisine_type2" respectively. There are restaurants with cuisines types upto 8.

conclusion

- The model is able to predict with 76% of accuracy the colored bin the New restaurant will fall into
- only 12.7% of restaurants have book table options in Bengaluru
- Only 23% of restaurants have ratings between 4-5.

5.2 Improvement

- User interface can be made which takes inputs for a new restaurants
- Neural network can undergo pruning to remove unwanted excess nodes
- creating more columns for cuisine types to separate the comma separated cuisines
- XGBoost and other algorithms can be used to check if it performs better than NN
- Creating unit tests for testing stages at the time of training the model, processing and predicting.