Modelling Rating Prediction Algorithms on Google Local Reviews: A Comparative Study

Abstract— **Google Reviews have become an integral part of our daily lives. We decide places to visit, restaurants to dine, and services to utilize based on these reviews. Furthermore, the general population and their sentiments about a business have a huge impact on the performance of the business. In this project, we utilize the Google Local Reviews data set to predict the potential rating when a user reviews a business. First, we describe our pre-processing and feature engineering strategies for extracting relevant information from the data set. We then proceed to define the predictive modelling problem and subsequently train several models along with baselines. Finally, we analyze the results obtained and conclude by providing a summary of the models' performances. We also discuss the relevant literature.**

Index Terms— **Google Reviews, Regression, Machine Learning, Deep Learning, Data Modelling, Personalized Recommendation.**

I. INTRODUCTION

The Google Local Reviews Data set[1] consists of three files that contain information specific to users, businesses, and reviews, respectively, along with metadata. To derive meaningful information from the data and to define a predictive problem, we first clean, transform, prune, and split the data. Then, we perform exploratory data analysis and data visualization to find interesting trends, patterns, and seasonalities in the local reviews data. We perform feature engineering on the pruned data set, thereby creating useful features for optimizing the regression problem at hand. Here, we intend to predict the rating a user would give to a business based on the user and the business/place metadata. Additionally, we look at a scenario where we do not have these metadata attributes, i.e., we have only the `(userID, placeID)` pairs. We approach both these tasks by proposing several baseline (intuitive) models and the design of a few machine and deep learning models, along with the evaluation metrics. Our aim is to create a model that fits the data and generates precise predictions[2].

II. LITERATURE REVIEW

The data set we use for our analysis and experiments is the raw version of the Google Local Reviews data. We transform the data in such a way that we can prepare regression-like predictive problems and optimize the representation of the data set through encoding tehniques, process and clean the raw data and also bring about notions of data completeness, sparseness and cardinality in the process of analyzing and pre-processing the data. Several regression and classification based problems exist in literature for such data sets. For example, finding the most optimal place and time to start your business of a particular field and analysis of customer trends so that businesses can serve consumers better.

Similar data sets studied in the past include finding sentimental analysis of a particular business based on google reviews.

State-of-the-art models found for recommendation tasks include latent factor models with user-item relation used in movies and products recommendation based on users history. Some of the well known good performing models for regression include XGBoost Regressor and Linear Regressor which are simple models that performs quite well. Now a days there has been a rise of Deep Learning Neural Network models which can learn highly complex coorelation between features very well, which most of the simpler models fails to understand.

We in turn propose a problem statement on the data set in such a way that we can employ some of these techniques and we devise a couple of our own data pre-processing methods to prepare the data such that it optimizes regression performance and makes the data much more intuitive and meaningful. Subsequent sections describe how we achieve the same.

[1]`https://cseweb.ucsd.edu/~jmcauley/datasets.html#google_local`

[2]Our code can be found at `https://github.com/AwsManas/Assigment-2`

III. Exploratory Data Analysis & Insights

In this section we describe the exploratory data analysis carried out on the Google Local Reviews data set (hereinafter referred to as ‘data set’).

A. Data Pre-processing

We perform the following pre-processing on the data set:

1. We merge the ‘user’ and ‘places’ data tables from the data set based on the criterion of ‘gPlusUserId’ and ‘gPlusPlaceId’ matching and we obtain a resulting ‘merged’ table. We drop all the columns that have extremely low completeness or extremely high randomness and uniqueness as they would skew our data and could result in sub-optimal models. The resulting dataframe is shown below. The dataframe has 8649011 rows and 11 columns.

```
merged.head()
```

	userName	gPlusUserId	rating	reviewText	categories	gPlusPlaceId	unixReviewTime	name	price	gps	closed
0	an lam	100000010817154263736	3.0	Chất lượng tạm ổn	[Giải Trí - Café]	108103314380004200232	1.372687e+09	Cà Phê Thăng Long	None	[10.852044, 106.65971]	False
1	hoang long nguyen	101659842775092396018	5.0	Good coffee, nice and peaceful place	[Giải Trí - Café]	108103314380004200232	1.354888e+09	Cà Phê Thăng Long	None	[10.852044, 106.65971]	False
2	Hong Le	107574994242995460712	2.0	Cho heo	[Giải Trí - Café]	108103314380004200232	1.352015e+09	Cà Phê Thăng Long	None	[10.852044, 106.65971]	False
3	HALİL TURGUT	100000013500285534661	5.0	Wc si temiz duzenli..	[Turkish Cuisine]	102194128241608748649	1.342871e+09	Selale Restaurant	None	[37.8037, 29.2209]	False
4	Akudosoft Yazılım	105271324704942360981	5.0	None	[Turkish Cuisine]	102194128241608748649	1.373148e+09	Selale Restaurant	None	[37.8037, 29.2209]	False

Fig. 1. Merged Dataframe

2. We prune the data set by dropping columns with excessive `NaN` values, and perform `NaN` imputation on relevant columns like ‘gps‘. We also expand the gps column to two columns [’lat’, ’lon’] indicating latitude and longitude (geolocation) of the place. We thus have a cleaned data set which we now call the ‘pruned’ version of the data with 1299688 rows × 13 columns.

3. We utilize this pruned data set and geolocations to create a mapping of the various types of ratings that we see across all businesses/places in the world. The pruned data set and its properties are given below.

B. EDA & Outlier detection

We perform the following on the pruned data set to identify meaningful patterns and extrapolate from the data.

1. Compute the basic stats of every column in the data set and understand their relevance for modelling a predictive task. We check the following components:

- Completeness of the entire data set and that of every column in the data set.
- Missingness of the entire data set and that of every column in the data set.

```
In [20]: final_df.isna().sum()

Out[20]: userName                0
         gPlusUserId             0
         rating                  0
         reviewText        2649295
         categories         135813
         gPlusPlaceId            0
         unixReviewTime      42698
         name                    0
         price             6782945
         gps                 27431
         closed                  0
         dtype: int64

In [21]: dfSize = pruned_df.size

In [22]: ## Ratios

In [23]: null_sums = final_df.isna().sum().sum()

In [24]: #Completeness of entire dataframe
         null_sums / dfSize * 100

Out[24]: 57.04435336670149

In [25]: # Missingness of entire dataframe
         (dfSize - null_sums)*100/dfSize

Out[25]: 42.9556466332985
```

Fig. 2. Some Computed Statistics

- Distinctness of the entire data set and that of every column in the data set.

2. We compute the following stats in the data set:

- Mean value
- Standard deviation
- Min, Max Values
- Percentiles
- Entropy

```
In [31]: from math import e

         def pandas_entropy(column, base=None):
             vc = pd.Series(column).value_counts(normalize=True, sort=False)
             base = e if base is None else base
             return -(vc * np.log(vc)/np.log(base)).sum()

In [34]: pandas_entropy(pruned_df['rating'])

Out[34]: 1.4216977063054153

In [43]: pandas_entropy(pruned_df['unixReviewTime'])

Out[43]: 13.92883155136558

In [33]: pandas_entropy(pruned_df['reviewTextLength'])

Out[33]: 6.423664854916784
```

Fig. 3. Some Computed Statistics

We infer that columns with high entropy (or degree of randomness) can be removed and the max and min values gave us a sense of how we can normalize the data. Data scaling and pruning was performed further with the information we obtained through EDA.

3. Next we perform some outlier detection in our pruned data set to remove skewness-causing entries in our data. This will greatly help us in regularization of our models.

Examples to depict some of the outlier detection techniques we employed are:

- Outlier detection based on z-score thresholding: Here we use our ‘reviewTextLength’ feature (discussed in a later section) and we compute the z-score for every entry of that feature. We retain only the rows where -2<`z_score`<2 for the given feature. This way we remove outliers i.e. for ex:- Users who give only 1-10 character reviews or give 1000-2000 character reviews as many of these are junk reviews and will bias our model extensively.

- Cardinality based pruning: We drop columns with low & high cardinalities based on some thresholds we defined for each.
- Percentile based outlier detection: We remove all the reviews/data points that have their review-TextLength on the extremely higher side of the percentile distribution i.e. 99-100 percentile range.
- Outlier filtering based on timestamp: We noticed that there were some extremely old and irrelevant/out of trend reviews such as a review from 1990 and a review from 2001 etc while the most recent review for the specific place was circa 2014. It makes sense to keep only the most recent reviews and not reviews that were given 2 decades ago.

C. Data Visualizations & Inference

Once we completed our data pruning, outlier detection, scaling and normalization, we visualized various stats and columns in the data so as to derive more meaningful patterns that we can exploit in our models. This information is subsequently used in data set localization and for feature engineering purposes.

Given below are the screenshots of several visualizations that we performed on the data set and we provide all of our insights subsequently.

```
In [16]: plt.hist(pruned_df['rating'],bins = 5)
         plt.show()
```

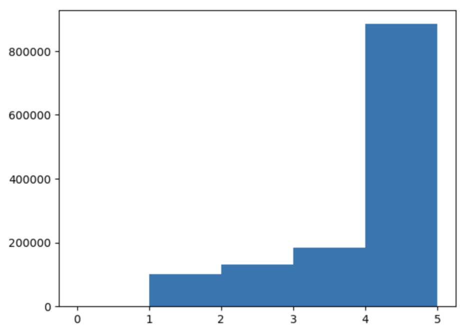

Fig. 4. Rating distribution cross-population

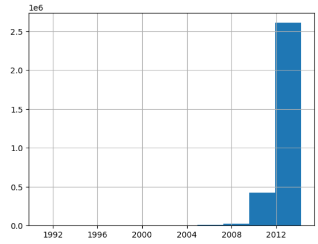

Fig. 5. No. of Ratings available year-wise

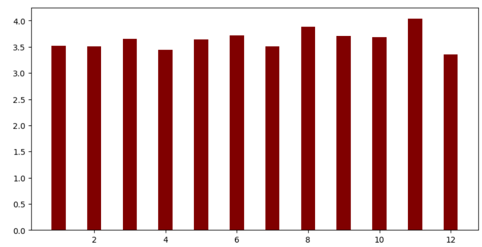

Fig. 6. Month Wise Rating Distribution in U.S.

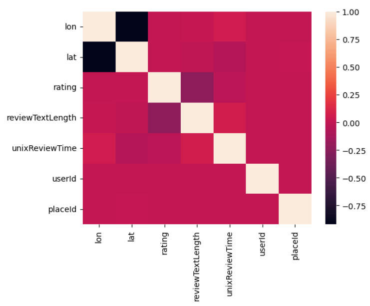

Fig. 7. Correlation Map of features for all data

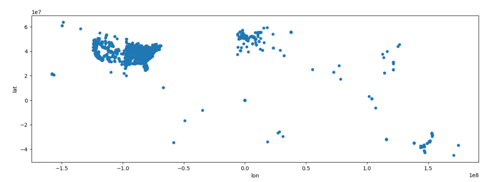

Fig. 8. World Map of Locations present in Places data

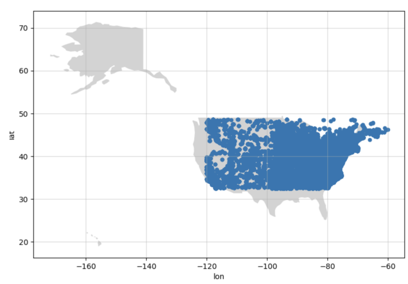

Fig. 10. Localized reviews distribution across U.S.

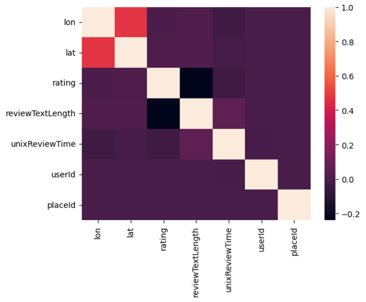

Fig. 9. Correlation Map of features of ratings data across the US

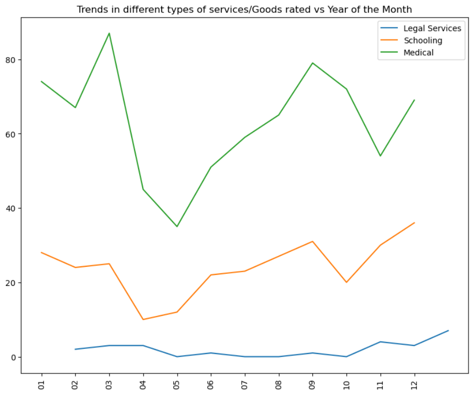

Fig. 11. Monthly Trends in Rating of Various Services

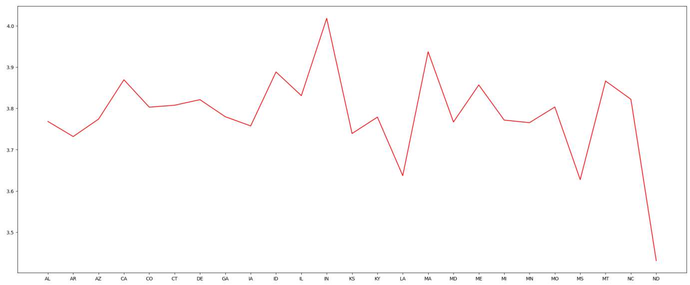

Fig. 12. US state-wise average rating across all services

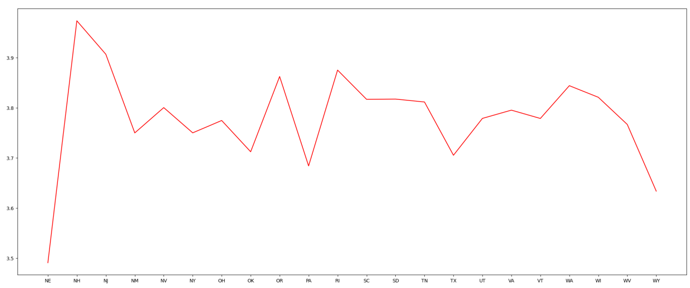

Fig. 13. US state-wise average rating across all services

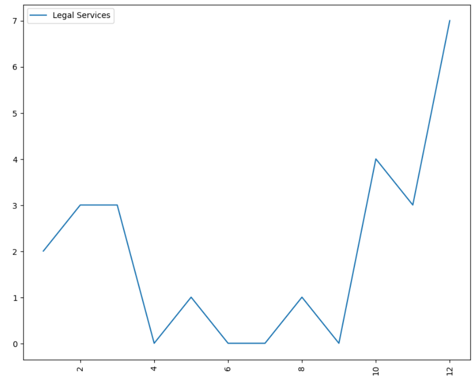

Fig. 14. Monthly Trends in Rating of Legal Services

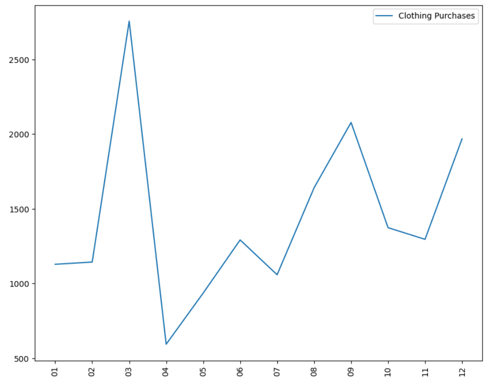

Fig. 15. Monthly Trends in Rating of Cloth Purchases

Fig. 17. Temporal rating trends in 4 Types of Restaurants

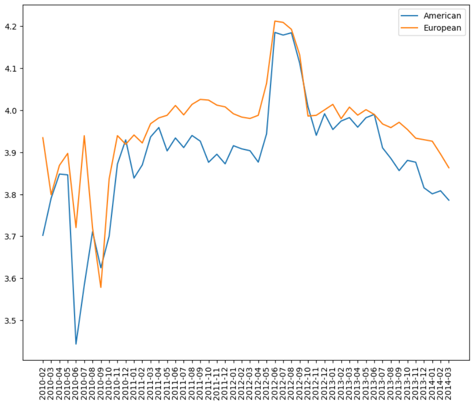

Fig. 16. Temporal rating trends - American & European Restaurants

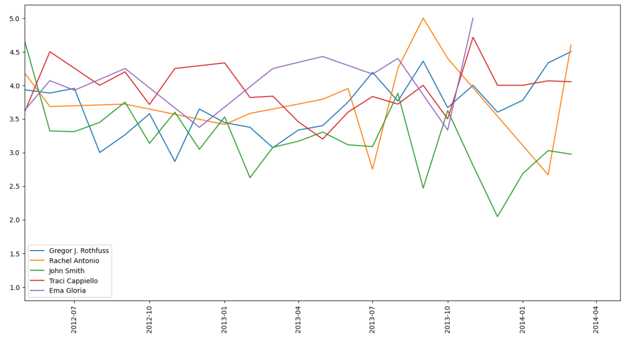

Fig. 18. User rating trends for a subset of users

1) Analysis and Insights from the Visualizations: :
Apart from some obvious preliminary intuitions, we've collated some deeper insights and correlations that we observed in the data in this section as a preclude to feature engineering.

- Majority of the ratings >62.5% tend to be on the higher side of 4-5 stars. This indicates that users tend to be more biased towards rating higher than lower.
- We see from the world map distribution that there is a dense set (8 million) of ratings in the United States alone, which we can further localize into states to obtain state-wise business/place ratings.
- The categories column has a strong correlation on the type of people that rate a business and the influence of categories are dynamic over the years.
- We see from temporal visualizations that the ratings tend to vary drastically based on the time of the year that a particular business is rated. This means that the 'year-month' data point would be a strong indicator of user ratings.
- Categories can be clubbed. For Ex:- a specific place can have multiple categories associated with it, but on an n-dimensional vector space, we can see that these categories lie close enough that they can be grouped to a single category thereby dimensionality reduction can be employed.
- The location of a specific business has a huge impact on the business rating, performance (i.e. whether the business would close or not) and also on user sentiments. The variation can be seen in multiple graphs.
- It is also evident from the visualizations that though the sentiments/ratings of 2 individuals for the same place/business in a large populus may vary drastically based on their tastes. The general rating trend for two countries (or large populations) as a whole are collective and are similar in distribution. The ratings are consistent over time.
- Demand, Utilization and rating of certain services or businesses such as 'Gift Shops', 'Hospitals', 'Schools', 'Legal Services' etc. vary greatly based on the time of the year, and a trend in the data is noticed across countries as well.

Considering these insights, we perform some feature engineering on the data set next.

D. Feature Engineering

The following describe our feature engineering steps:

- First we reduce the scope of our data set to account for only interactions (i.e. user-business ratings) that occurred within the United States. This gave us about 3 million data points to work with. The way we extracted this information was on the basis of the latitude and longitude of the GPS data.
- We wrote python scripts to find the Geospatial distance in miles from the <latitude, longitude> of our business/place to the center of each state in the US, and used the nearest neighbour approach to zero-in on which state in th US a particular business lies in. Thus we extracted a new piece of relevant information as an indicator of the rating given to a place.
- Since it was evident from our temporal correlation analysis that the column 'unixReviewTime' is highly relevant, we extract the 'month-year' information from it and create 'month' and 'year' features.
- Our EDA showed that the 'categories' column in the dataframe had extremely high cardinality but since the 'categories' column is too important of a feature to drop we resorted to a customized

category grouping technique using 'word' similarities (i.e. of all the categories that a particular place/business belongs to, which category best describes the place). We posit that there exists 10 such mutually exclusive categories that we can optimally 'bin' our places into. which are ('Associations/Organizations', 'Entertainment', 'Legal Services' 'Medical', 'Public', 'Restaurant', 'School', 'Shops and Stores', 'Venues','Others') and we bin all the data points into each of these categories. Thus a new high-correlation feature is created.
- Additionally, Our category wise user rating prediction analysis in the visualizations showed a strong dependency of average user rating for a category with the places that belong to those categories. Therefore, we systematically generated the "User Average Ratings" for every business grouped "Category-Wise" and included that as another data point in the parameter vector.
- Finally we perform a one-hot-encoding of the columns 'year', 'month','state','final_category' to get the final data set.

IV. Predictive Modelling & Tasks Identification

Before we define our predictive model, here we attach a glimpse of what our final data set looks like after all the steps in Section 2.

	gPlusUserId	gPlusPlaceId	userCategoryAvgRating	rating	year_1990	year_2001	year_2002	year_2003	year_2004	year_2005	...
8	100000059843227870895	117682556988480065346	3.000000	3.0	0	0	0	0	0	0	...
9	102006303856203035288	117682556988480065346	4.250000	5.0	0	0	0	0	0	0	...
10	102013085596157142926	117682556988480065346	4.250000	4.0	0	0	0	0	0	0	...
11	103615206841486883067	117682556988480065346	3.500000	4.0	0	0	0	0	0	0	...
13	104424487436220482372	117682556988480065346	4.285714	3.0	0	0	0	0	0	0	...
...	...	...	...	...	...	...	...	...	...	...	...
3078329	114674093439947536559	107746309351558167302	5.000000	5.0	0	0	0	0	0	0	...
3078339	114674456986871638917	115173587425360514264	3.250000	3.0	0	0	0	0	0	0	...
3078346	114674905157577133464	112706745020560872721	1.000000	1.0	0	0	0	0	0	0	...
3078355	114674995719058628862	103651536920568572150	5.000000	5.0	0	0	0	0	0	0	...
3078356	114674995719058628862	108487630248484037894	5.000000	5.0	0	0	0	0	0	0	...

807629 rows × 88 columns

Fig. 19. Peek into our Final Prepared Dataset

```
Index(['gPlusUserId', 'gPlusPlaceId', 'userCategoryAvgRating', 'rating',
       'year_1990', 'year_2001', 'year_2002', 'year_2003', 'year_2004',
       'year_2005', 'year_2006', 'year_2007', 'year_2008', 'year_2009',
       'year_2010', 'year_2011', 'year_2012', 'year_2013', 'year_2014',
       'month_01', 'month_02', 'month_03', 'month_04', 'month_05', 'month_06',
       'month_07', 'month_08', 'month_09', 'month_10', 'month_11', 'month_12',
       'final_category_AssocOrgs', 'final_category_Entertainment',
       'final_category_Legal', 'final_category_Medical',
       'final_category_Others', 'final_category_Public',
       'final_category_Restaurant', 'final_category_School',
       'final_category_ShopsStores', 'final_category_Venues', 'state_AL',
       'state_AR', 'state_AZ', 'state_CA', 'state_CO', 'state_CT', 'state_DE',
       'state_GA', 'state_IA', 'state_ID', 'state_IL', 'state_IN', 'state_KS',
       'state_KY', 'state_LA', 'state_MA', 'state_MD', 'state_ME', 'state_MI',
       'state_MN', 'state_MO', 'state_MS', 'state_MT', 'state_NC', 'state_ND',
       'state_NE', 'state_NH', 'state_NJ', 'state_NM', 'state_NV', 'state_NY',
       'state_OH', 'state_OK', 'state_OR', 'state_PA', 'state_RI', 'state_SC',
       'state_SD', 'state_TN', 'state_TX', 'state_UT', 'state_VA', 'state_VT',
       'state_WA', 'state_WI', 'state_WV', 'state_WY'],
      dtype='object')
```

Fig. 20. All Columns in our Final Prepared Dataset

Based on this, Some of the possible predictive tasks for this data set include:

1. Given user and place metadata (i.e. all columns in the data set except rating), predict what rating a user would give to a business.

2. Given the location, and historical user-ratings of a place, predict whether a business would fail/succeed.

3. Predicting the top-k most similar users for a particular user given the way they rated different places in different categories.

In this report we discuss the implementation, experiments and analysis of Task-1. For modelling Task-1 we first concretize our problem statement, and suggested line of thinking. We narrow down the various approaches we can use and algorithms that fit the task, and discuss how we should evaluate our model.

Problem Statement: For a user with rating characteristics 'x' and for a place with metadata such as 'location (state)', 'month' ,'year' etc. develop a model that predicts what rating the user would give to the place.

We propose regression-based techniques to develop solutions to this problem as we need to make real-valued predictions in the range 1-5 (star-ratings). Multivariate regression is intuitively the ideal way to approach this problem. So Initially we start by building a naive baseline and then suggest intelligent baselines. We later on model the regression problem and perform a comparative analysis of the performance of several models for this task. The performance is evaluated by choosing suitable evaluation metrics. We also attempt to avoid over-fitting by exploring various techniques of regularization.

V. Model Selection & Experiments

The Experimental setup we use are the following:

1. Data is split into 3 subsets 'train','validation','test' in the ratio 60:20:20.

2. Regularization is done wherever applicable ensuring to optimize the validation MSE.

3. Experiment epochs are set such that they converge when the difference in inter-trial MSE's are < 0.0001.

4. We progress from simpler to more personalized and complicated models and evaluate them across with the same baselines.

The models that we chose, with parameters and a couple of evaluation metrics we considered for our experiments are as follows:

A. Baselines

1) Always predict mean: Our first naive baseline is predicting the mean rating of all businesses for all users satisfying any condition such as falling within a specific state/category. The different variants of this baseline are to Always predict mean:

- Per state: Find avg of all ratings per state and return the average rating for every state as the predicted rating.
- Per category: Do the same but based on category
- Per unit of time: Do the same but based on month and year of the rating.

2) Always predict popular mean: An alternative baseline can be to Always predict the mean rating of the most popular place/business satisfying a specific condition. The variants of this baseline are:

- Per state: predict the average rating of the most popular place in that state.
- Per category: Do the same but for the most popular place of that category.
- Per unit of time: Do the same but for the most popular item within that month-year time frame.

Table I below summarizes the baseline performances.

TABLE I
BASELINE PERFORMANCE ON TEST SET

Baseline Model	Test MAE	Test MSE
Always Predict Mean		
Per State	1.0677	1.674
Per Category	1.0670	1.673
Per unit of time	1.0711	1.662
Always Predict Popularity		
Per State	1.1808	2.335
Per Category	1.0675	1.802
Per unit of time	1.1980	2.688

B. Models

- Linear Regression: Simple multi-variate Linear regression with no regularization. The weights assigned by the model to our features are given in the figure below.

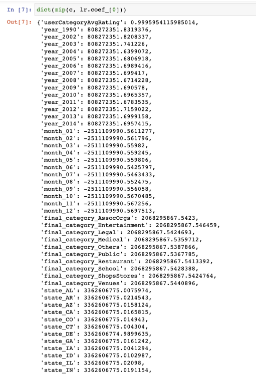

Fig. 21. Linear Regression model feature weights/coefficients

- Ridge Regression: Default Ridge regression i.e. with alpha = 1.0. A variant of linear regression which regularizes given the L2-norm.
- Fix regularization alpha in Ridge regression: Here we found out the optimal alpha (regularization term) which turned out to be alpha = 100. We can clearly see that the weights are regularized and fall within the scale of our ratings in the figure below.

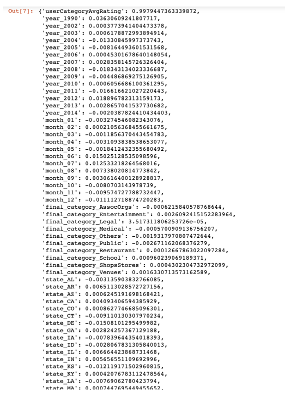

Fig. 22. Ridge model with regularization Coefficients

- Random Forest Regression: Random forest regression with hyper-parameters set. `n_estimators = 100, min_samples_split = 250`.
- XGBoost Regression: A gradient boosted decision tree or GBDT algorithm that does extremely well for sparse data. we train an `XGBRegressor` with `reg_lambda = 2.0,learning_rate = 0.1`.
- Bayesian Personalized Ranking: A personalized ranking technique, where we pass only the user-place interaction pairs and the model learns user-features based on the rating that a user gave to a specific place. This is a simple model with just user/place biases (`alpha, betaUser, betaPlace`).

- BPR with latent factors: BPR with LF is a model that not only takes in the User-Place biases as parameters to train but also utlilzes a `gamma` term which are the latent factors of users and items. The resulting function we train has the parameters (`alpha, betaUser, betaPlace, gammaUser, gammaPlace`).
- Tensorflow DNN: A feed-forward deep-neural network trained on tensorflow with 5 dense layers and 4 dropout layers for regularization. The sequential model is described in the figure below.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (484576, 512)             43520

 dropout (Dropout)           (484576, 512)             0

 dense_1 (Dense)             (484576, 256)             131328

 dropout_1 (Dropout)         (484576, 256)             0

 dense_2 (Dense)             (484576, 64)              16448

 dropout_2 (Dropout)         (484576, 64)              0

 dense_3 (Dense)             (484576, 16)              1040

 dropout_3 (Dropout)         (484576, 16)              0

 dense_4 (Dense)             (484576, 1)               17

=================================================================
Total params: 192,353
Trainable params: 192,353
Non-trainable params: 0
_________________________________________________________________
```

Fig. 23. Sequential TF-DNN Layers & Parameters

TABLE II
REGRESSION MODEL PERFORMANCE

Regression Model	Test MAE	Test MSE
Linear Reg.	0.3567	0.4172
Ridge Reg.	0.3566	0.4172
Ridge with reg.	0.3565	0.4172
Random Forest	0.3616	0.4245
XGBoost	0.3555	0.4173
BPR <u,i>	1.0235	1.5905
BPR w/ LF <metadata>	1.0235	1.5905
TF-DNN	0.3654	0.4166

C. Evaluation Metrics

- MAE: The Mean Absolute Error which seems optimal for evaluating our models due to the characteristic of penalizing errors by the same magnitude as the deviation.
- RMSE / MSE : Mean Squared Error or Root Mean Squared Error which penalizes heavily for large deviations and very little for smaller deviations.
- R-square / Adjusted R-square: While MSE captures the residual error, the adjusted R-square on the other hand represents the fraction of variance of the actual value predicted by the regression model.

The table above depicts the performance of each of these models on the data set. We consider the MAE and MSE to be our evaluation metrics.

VI. RESULTS & CONCLUSION

From our experimental analysis we infer that several forms of regression perform reasonably well on the data set. We see that the Tensorflow-Deep Neural Network with dropout regularization performs the best on the data considering MSE to be the evaluation metric. In terms of MAE, XGBoost regressor was the best-performing model. While the performance of all models are reasonably close we see that Bayesian Personalized Ranking both with and without Latent Factors perform really bad on the data. This is predominantly due to the fact that our data is sparse, with either few interactions or many new interactions. The model has to deal with cold-start and sparseness more often than not, and this renders a personalized ranking strategy ineffective for such data.

We explored several data cleaning, pruning and encoding techniques via EDA which provided us a head start with a great baseline and Linear Reg. performance. We've prepared the data and regression models in such a way that it avoided over fitting by introducing regularization wherever necessary.

Data Visualization experiments performed on the data yielded great obvious and inherent insights from our data which helped us to define underlying patterns and visualize the interplay of multiple features within the data set.

Future work on the realm of predictive analysis and modelling on the data set could include the prediction of places/businesses shutting down, predicting similar users, development of recommendation engines based on user ratings, modelling the review text data and finding N-most-similar reviews/places based on user sentiment, N-most similar places based on the place description and category etc.

In conclusion, While our analysis of Google Local Reviews data was fairly limited in scope, we believe that multiple inherent characteristics exist in the data set that can be put to great use in development of future models. We see that identifying and exploiting subtle relations and correlations in the data go a long way in the development of optimal ML predictive models.

References

[1] Pasricha, Rajiv, and Julian McAuley. "Translation-based factorization machines for sequential recommendation." Proceedings of the 12th ACM Conference on Recommender Systems. 2018.

[2] He, Ruining, Wang-Cheng Kang, and Julian McAuley. "Translation-based recommendation." Proceedings of the eleventh ACM conference on recommender systems. 2017.

[3] https://en.wikipedia.org/wiki/Degeneracy_(graph_theory)

[4] https://polyglot.readthedocs.io/en/latest/Detection.html#mixed-text

[5] https://www.tensorflow.org/tutorials/keras/regression

[6] Singh, Ruchi, Yashaswi Ananth, and Dr Jongwook Woo. "Big data analysis of local business and reviews." Proceedings of the International Conference on Electronic Commerce. 2017.

[7] He, Ruining, Wang-Cheng Kang, and Julian J. McAuley. "Translation-based Recommendation: A Scalable Method for Modeling Sequential Behavior." IJCAI. 2018.

[8] Muñoz, Jesús, and Ángel M. Felicísimo. "Comparison of statistical methods commonly used in predictive modelling." Journal of Vegetation Science 15.2 (2004): 285-292.

[9] https://towardsdatascience.com/plotting-geographical-data-with-geopandas-338cc7e17e4e

[10] https://towardsdatascience.com/data-visualization-using-seaborn-fc24db95a850