Technical Paper

Final Project - Data Science

General Assembly - Sydney - 2013

Pradeep . Data Science . GA . 27 November 2013

Topics

1. Problem
   (Description, Hypothesis...)

2. Data
   (Structure, Overview...)

3. Solution
   (Approach, Implementation...)

4. Conclusion
   (Observations, What's next...)

Problem

Yelp Recruiting Kaggle Competition

How many "useful" votes will a Yelp review receive?

Hypothesis

Reviews written by frequent and reputed users are most likely to get maximum number of useful votes.

Data description

In the training set:

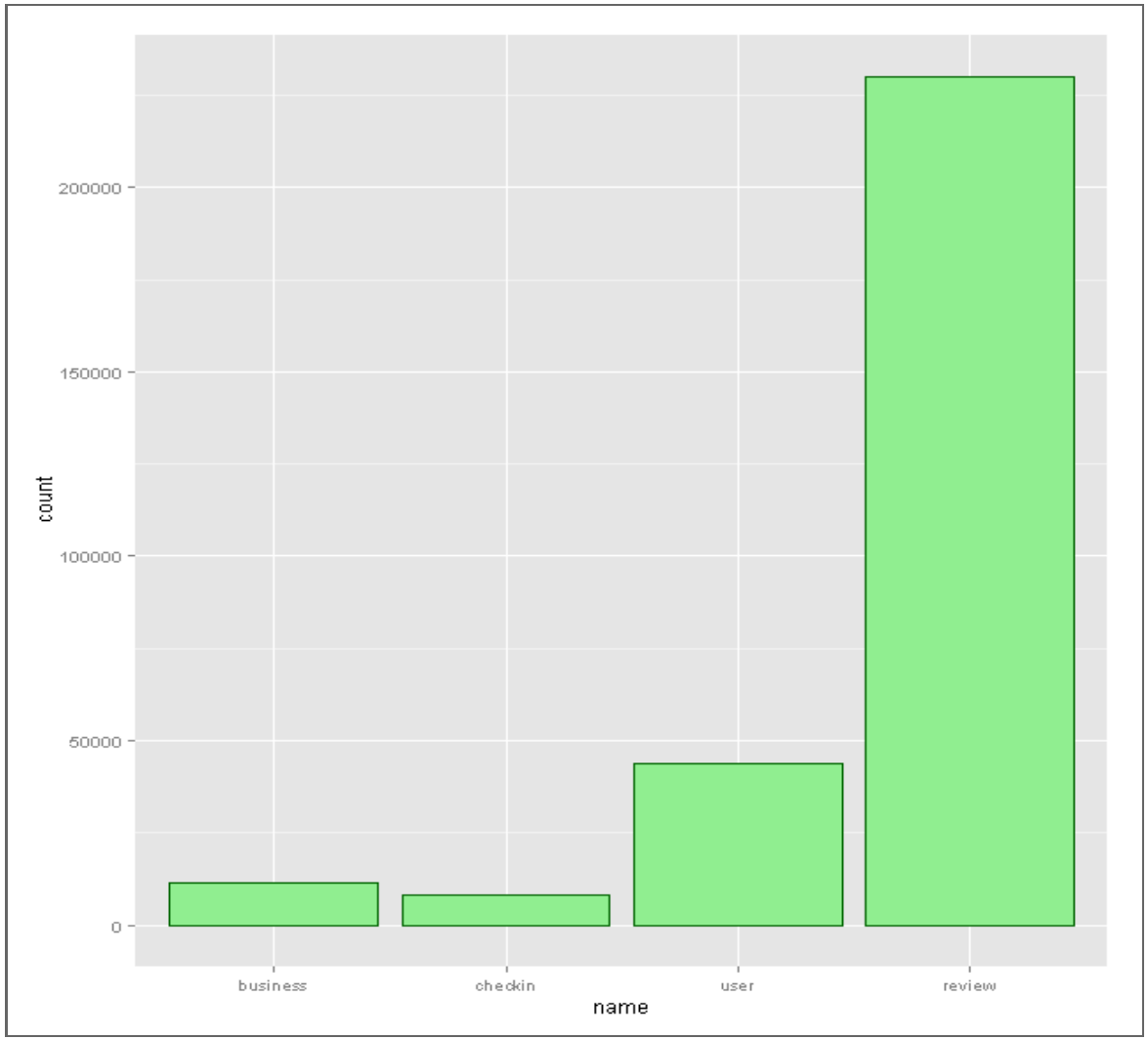
- 11,537 businesses
- 8,282 checkin sets
- 43,873 users
- 229,907 reviews

Each file is composed of a single object type, one JSON object per line. The training data was recorded on 2013-01-19. The testing data contains reviews, businesses, users, and checkins from the period between 2013-01-19 and 2013-03-12.

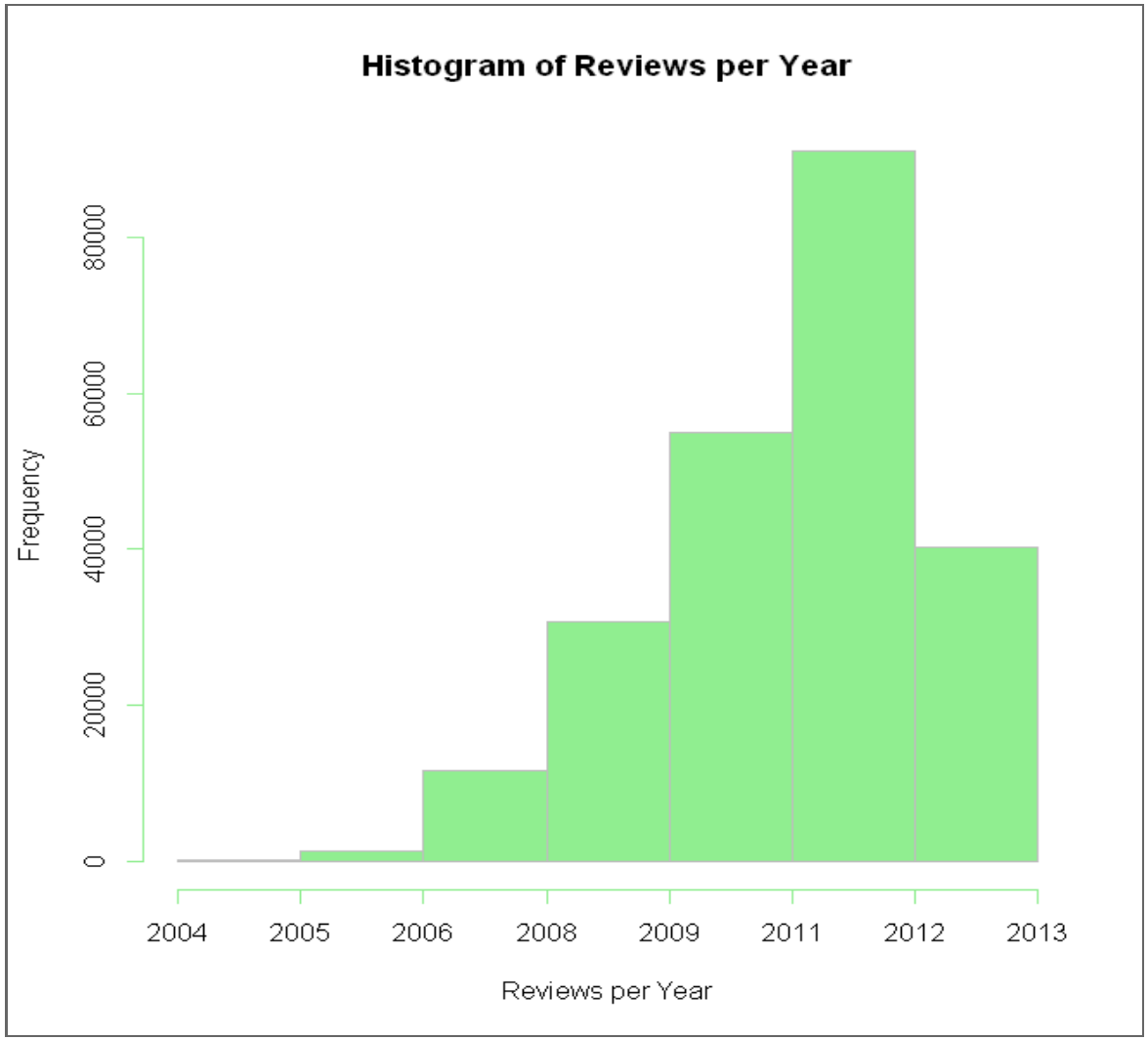
Data Features

Review	User	Business	Checkin
type	type	type	type
review_id	user_id	business_id	business_id
votes_useful	votes_useful	open	checkin_info
votes_funny	votes_funny	categories	
votes_cool	votes_cool	full_address	
stars	average_stars	stars	
date	name	name	
text	review_count	review_count	
user_id		city	
business_id		state	
		neighbourhood	
		latitude	
		longitude	

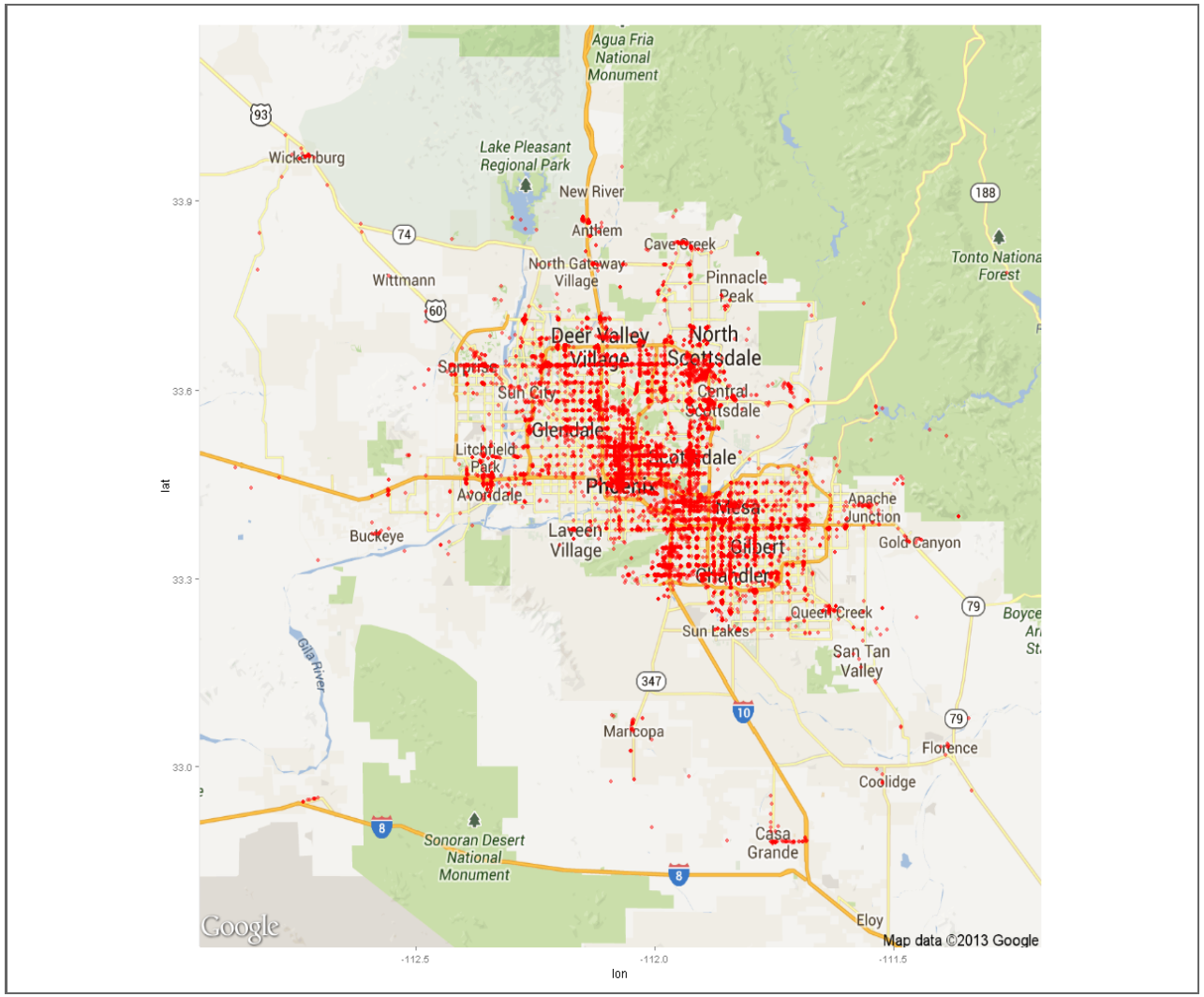
Dataset


200000
150000
100000
50000
0
count
business
checkin
user
review
name

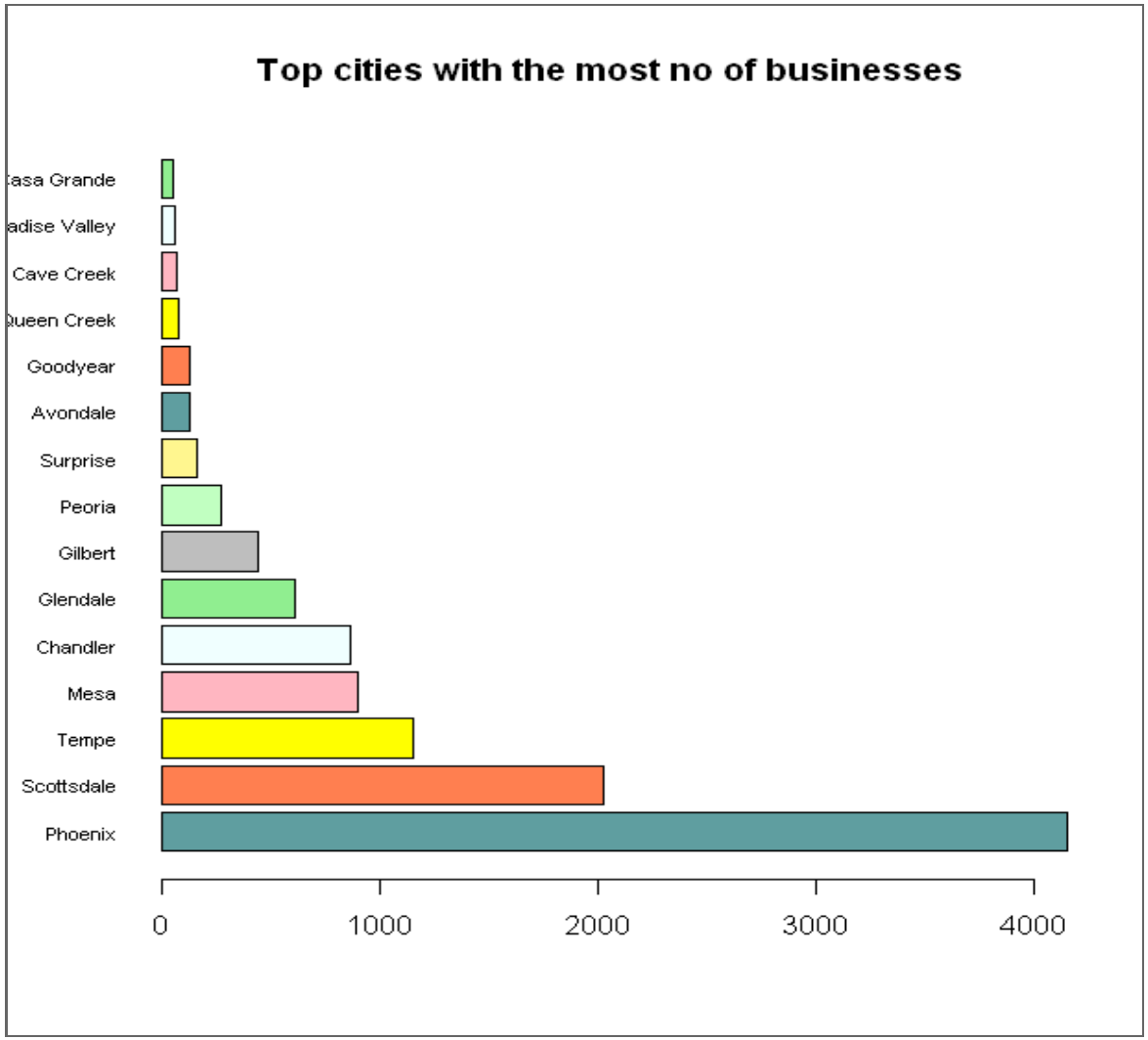
Reviews Age


Histogram of Reviews per Year
Frequency
0
20000
40000
60000
80000
2004
2005
2006
2008
2009
2011
2012
2013
Reviews per Year

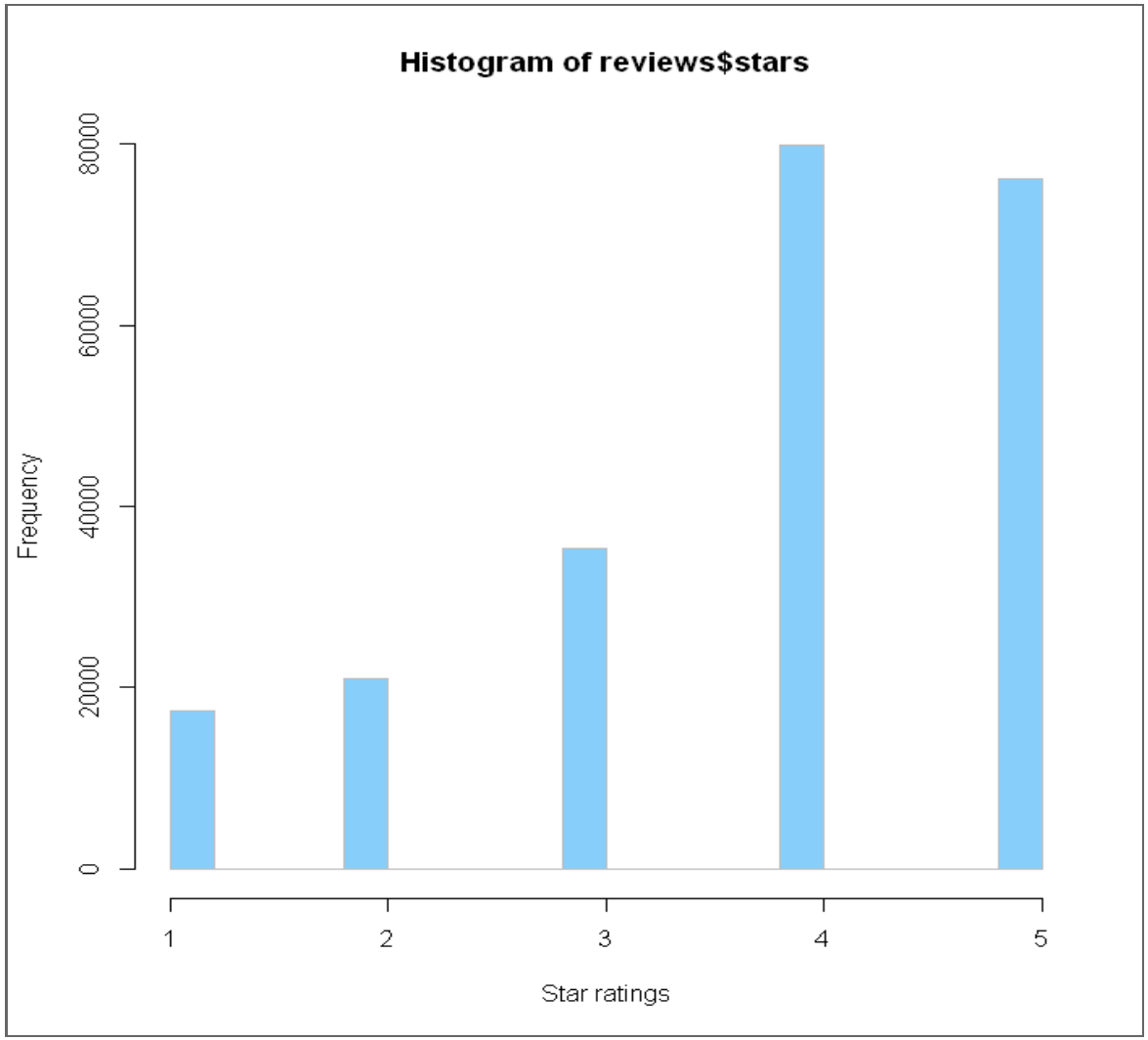
Business Locations



Top cities


Top cities with the most no of businesses
asa Grande
adise Valley
Cave Creek
ueen Creek
Goodyear
Avondale
Surprise
Peoria
Gilbert
Glendale
Chandler
Mesa
Tempe
Scottsdale
Phoenix
0
1000
2000
3000
4000

Reviews Star rating frequency

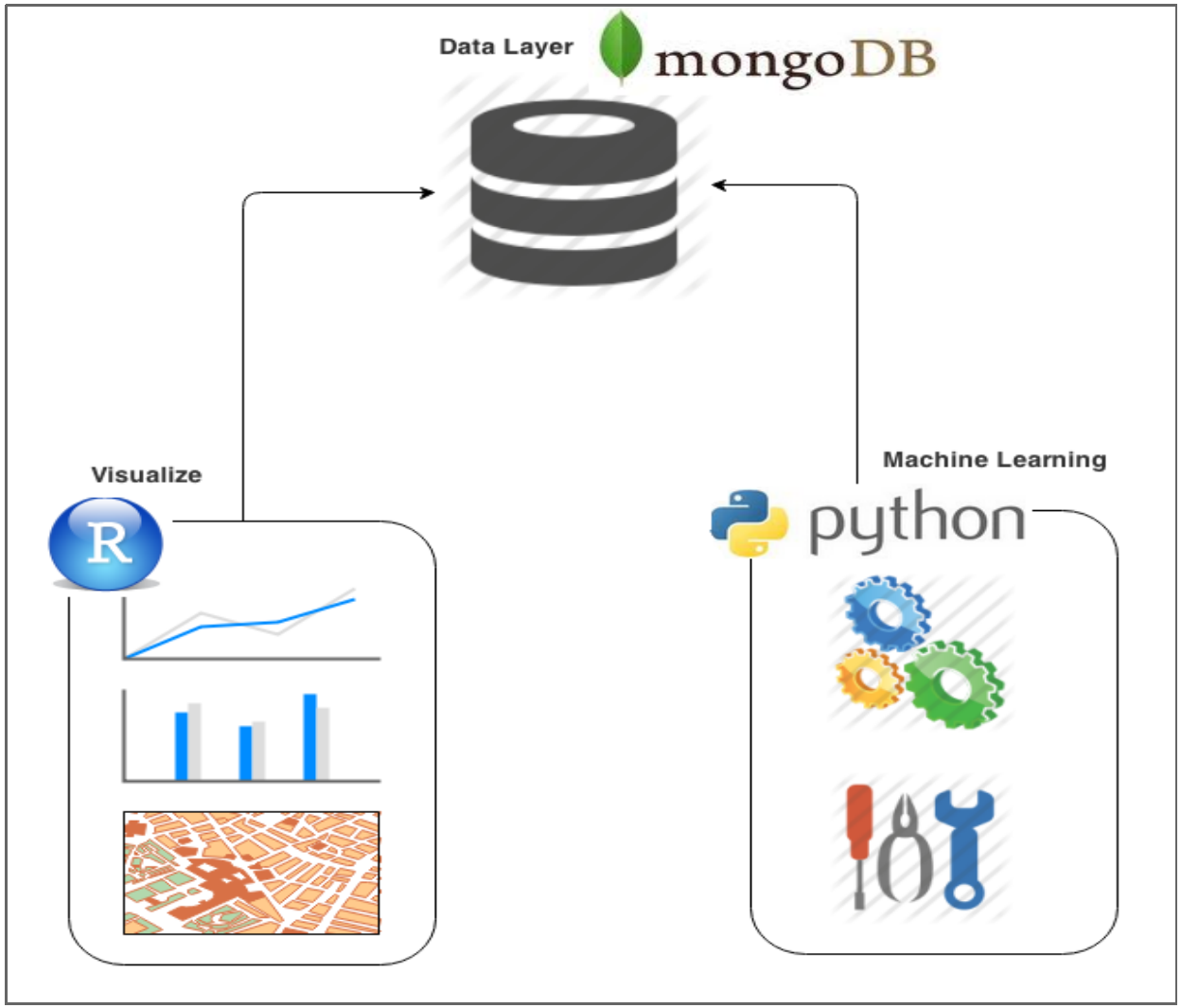


Solution

Breaking the problem into subtasks

- Preprocessing (Python)
- Data Mining (Python + MongoDB)
- Visualization (R, Python)
- Statistical Methods (Python - Sklearn)

Solution Environment



Approach

generalisation - making predictions from data

(Supervised - Regression)

Steps

- Determine input feature set
- Design algorithm
- Cross Validate
- Predict
- Evaluate the accuracy
- Repeat ...

Input features

Started with simple numerical features picking few at a time from review, business and user datasets. The idea was to quickly get to a working model and continue improving on it. I was able to achieve reasonably decent scores with handful of obvious ones such as review text length, business star rating, business review count, user review count, user average votes, and total number of business checkins.

Regressors tried

from sklearn

- linear_model.Ridge
- linear_model.SGDRegressor
- linear_model.LassoCV
- linear_model.ElasticNet
- linear_model.BayesianRidge
- ensemble.RandomForestRegressor
- ensemble.ExtraTreesRegressor
- ensemble.GradientBoostingRegressor

In short

```
def train(modelnames=[], features=[], predict=True, plot=True ...):
    #! -----------------------------------------
    #! train - cross validate - predict - plot
    #! -----------------------------------------
        X, y = get_features(limit=limit, features=features....)
        for name in modelnames:
            # create model
            # .....
            cross_validate(X,y,clf,folds=5,model_name=model_name,plot=plot ...)

            if predict:
                print '====== predicting ......'
                #! grab the complete test set for prediction
                Xtest, ytest = get_features(features=features ...)
                predict_and_save(X, y, Xtest, clf, features ...)
                print '====== predicting done ......'

        if plot:
            plot_error(rmsles,"Fold", "RMSLE", "Cross Validation using XYZ model")
```

Evaluation Criteria

Root Mean Squared Logarithmic Error (“RMSLE”) to measure the accuracy of an algorithm

$$\epsilon = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\log(p_i + 1) - \log(a_i + 1))^2}$$

Where:

- epsilon is the RMSLE value (score)
- n is the total number of reviews in the data set
- pi is the predicted number of useful votes for review
- a is the actual number of useful votes for review
- log(x) is the natural logarithm of (x)

Sentiment Scores

Calculated sentiment scores of each review by simply tokenizing using python-nltk library and extracting emotion score of each word from a precalculated bag of words.

- with different stemmers from Natural Language Toolkit *(Python)*
  - PorterStemmer
  - LancasterStemmer
  - RegexpStemmer

- Precalculated valence scores *(Online)*

Word	Score
ability	2
abuse	-3
accept	1

Score (user + business)

202	↓24	jprusa	0.57466	4	Mon, 17 Jun 2013 02:56:27 (-1.2h)	
203	↓24	Gaffer	0.57466	2	Thu, 11 Apr 2013 10:10:14	
204	new	davidkunio	0.57551	2	Sun, 30 Jun 2013 23:36:27	
205	↓25	Anonymous 38602	0.57569	13	Wed, 03 Apr 2013 02:14:50 (-24.5h)	
206	new	CP_Data	0.57575	3	Sun, 30 Jun 2013 23:31:11	
207	↓26	Analyticsbd	0.57592	5	Tue, 25 Jun 2013 15:12:37 (-38.4d)	
208	↓26	YelplessKaggler	0.57662	1	Sun, 21 Apr 2013 01:24:52	
209	↓26	sujitpal	0.57724	2	Thu, 13 Jun 2013 05:41:26	
210	↓26	Dave Masog	0.57732	6	Tue, 14 May 2013 20:40:42 (-6.8d)	
211	↓20	Zwicky	0.57859	24	Sun, 30 Jun 2013 05:22:59 (-3.2d)	
-		**pradeep pradhan**	**0.57862**	-	**Tue, 26 Nov 2013 08:09:39**	Post-Deadline

Post-Deadline Entry
If you would have submitted this entry during the competition, you would have been around here on the leaderboard.

212	↓27	Nancy R.	0.57952	6	Sun, 30 Jun 2013 17:09:55 (-12.8d)	
213	↓27	Anonymous 39412	0.58114	13	Sun, 16 Jun 2013 04:50:36 (-3.9d)	
214	new	Igor Bobriakov	0.58119	4	Sun, 30 Jun 2013 21:59:59	
215	↓28	Anonymous 56860	0.58149	9	Tue, 04 Jun 2013 11:17:33 (-32.6d)	
216	↓28	Anonymous 36668	0.58150	1	Sat, 15 Jun 2013 18:17:37	

Score (user + business + sentiments)

199	↓24	Rohit	0.57149	5	Tue, 09 Apr 2013 01:11:49 (-1.2h)
200	↓24	AlteryxAd	0.57371	2	Wed, 10 Apr 2013 22:51:30
201	↓24	Blue Ocean	0.57396	10	Sun, 16 Jun 2013 05:36:00 (-7.7d)
202	↓24	jprusa	0.57466	4	Mon, 17 Jun 2013 02:56:27 (-1.2h)
203	↓24	Gaffer	0.57466	2	Thu, 11 Apr 2013 10:10:14
204	new	davidkunio	0.57551	2	Sun, 30 Jun 2013 23:36:27
-		**pradeep pradhan**	**0.57555**	-	**Tue, 26 Nov 2013 23:10:54** Post-Deadline

Post-Deadline Entry
If you would have submitted this entry during the competition, you would have been around here on the leaderboard.

205	↓25	Anonymous 38602	0.57569	13	Wed, 03 Apr 2013 02:14:50 (-24.5h)
206	new	CP_Data	0.57575	3	Sun, 30 Jun 2013 23:31:11
207	↓26	Analyticsbd	0.57592	5	Tue, 25 Jun 2013 15:12:37 (-38.4d)
208	↓26	YelplessKaggler	0.57662	1	Sun, 21 Apr 2013 01:24:52
209	↓26	sujitpal	0.57724	2	Thu, 13 Jun 2013 05:41:26
210	↓26	Dave Masog	0.57732	6	Tue, 14 May 2013 20:40:42 (-6.8d)
211	↓20	Zwicky	0.57859	24	Sun, 30 Jun 2013 05:22:59 (-3.2d)
212	↓27	Nancy R.	0.57952	6	Sun, 30 Jun 2013 17:09:55 (-12.8d)
213	↓27	Anonymous 39412	0.58114	13	Sun, 16 Jun 2013 04:50:36 (-3.9d)

Score (user + business + sentiments + votes)

172	↓28	WHO?	0.55448	16	Thu, 09 May 2013 00:12:50 (-0.4h)	
173	↓22	WL	0.55484	14	Sun, 30 Jun 2013 01:17:37 (-2.5d)	
174	↓29	Javy	0.55535	3	Fri, 03 May 2013 18:38:41 (-37.8h)	
175	↓29	lemon	0.55565	5	Sat, 20 Apr 2013 23:57:02	
176	new	Kalyne Chagas	0.55567	3	Wed, 26 Jun 2013 18:10:23	
177	↓30	Will Hannah	0.55635	8	Fri, 21 Jun 2013 16:25:46 (-38.3h)	
178	new	Krishan Gupta	0.55693	1	Mon, 24 Jun 2013 05:42:15	
179	↓29	ScarletKnight	0.55797	14	Tue, 16 Apr 2013 21:18:40 (-24.3h)	
180	↑14	Anonymous 21146 ‡	0.55882	2	Sat, 29 Jun 2013 08:50:09	
-		**pradeep pradhan**	**0.55970**	-	**Wed, 27 Nov 2013 05:13:21**	Post-Deadline

Post-Deadline Entry
If you would have submitted this entry during the competition, you would have been around here on the leaderboard.

181	↓29	Anonymous 14983 ‡	0.55975	3	Tue, 16 Apr 2013 15:10:40	
182	↓29	Grant Watson	0.56063	10	Thu, 25 Apr 2013 14:18:00 (-10.7d)	
183	↓29	Icolladotor	0.56195	2	Thu, 02 May 2013 10:55:19 (-1.6h)	
184	↓29	Dai Li	0.56247	2	Sun, 02 Jun 2013 21:49:14	
185	↓29	Anonymous 45038	0.56378	5	Tue, 18 Jun 2013 16:42:23 (-21.3h)	
186	↓29	Anonymous 83354	0.56379	10	Tue, 16 Apr 2013 17:25:32 (-5d)	
187	↓29	Gus123	0.56415	10	Mon, 06 May 2013 17:09:55 (-12.9d)	

Conclusion

Challenges...

- Wanted to implement a model using Tf-idf (term frequency - inverse document frequency), but couldn't do it because of computational limitations

Observations...

- Sentiment scores did help but not much
- Of all my regressors, *GradientBoostingRegressor* gave the best score
- The solution is applicable to wide range of businesses where predictions are made based on user reviews and ratings
- I can easily reuse my solution (atleast parts of it) for solving similar problems

What next...

- Try some complex models like stacking, ensembling...
- Build interactive visualizations using R's "shiny" package
- Explore further into Textual data and Sentiment analysis

The End

Pradeep Pradhan

pradeeppradhan@gmail.com

@tweetpradhan