Project Report
Yelp Dataset Challenge 2015
(Z534 – Information Retrieval)

Report submitted by -
Charwad, Nayana Yashwant
Chintala, Sri Laxmi
Deshmukh, Renuka
Gaikwad, Aniket Sambhaji
Indana, Venkata Prudhvi Raj

Contents

1. Task: Category Prediction for Businesses

1.1. Research Question

For each yelp business, there is category information (one business may belong to multiple categories). Yelp also has the review and tip information for a particular business provided by users.

Example: "categories": ["Indian", "Restaurants"]

For task 1, the research question is to predict categories for each business by using the review and tip information.

1.2. Flow Diagram

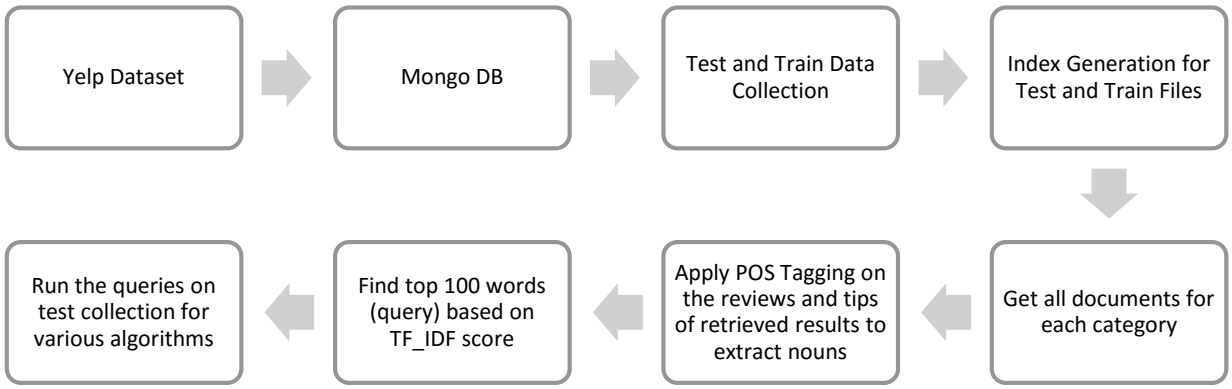

Figure 1: Flow Diagram for Task 1

1.3. Method/Algorithm

- **Index Creation**

1. The review, tip and business data was imported into MongoDB. Mongo DB was used solely as a data store to save the data set. The yelp dataset was then divided into test (30%) and train (70%) collections.
2. This data was then read into a java program to create training and test indices (Business ID, category, review, tip, review-rating) using Lucene API.

➢ **Query Generation**

1. For each category in the Lucene train index, all documents containing the given category are extracted. The extracted documents are then treated a sub-corpus for the category.
2. Review and tip data are retrieved from each document in the sub-corpus and consolidated into a string. This string is then POS tagged to extract nouns.
3. From the obtained array of nouns, TF*IDF is calculated for each nouns to identify the top scoring nouns.
4. Queries are generated in multiple iterations by varying the number of words in the query from 10 to 100, in the interval of 10.

➢ **Category Prediction**

1. Each query from the query file, is run on the Lucene test index for predicting the category of businesses in the test set.
2. Various smoothing algorithms supported by Lucene are applied to test for the best output. Additionally, the maximum number of predicted per category are varied from 50 to 100 with an internal of 25.
3. Finally, the output is compared against the ground truth for evaluation purposes.

1.4. Experiments and Evaluation

We generated task 1 output by for taking top 50 nouns into consideration for some randomly picked categories (Chicken Wings, Chinese, Horseback Riding, Party& Event Planning, Tax Services) and then generated output from our task 1 and compared Lucene similarities Vector space model, BM25, Language Model with Dirichlet Smoothing, Language Model with Jelinek Mercer Smoothing results. Below figures has precision and recall values for above similarities.

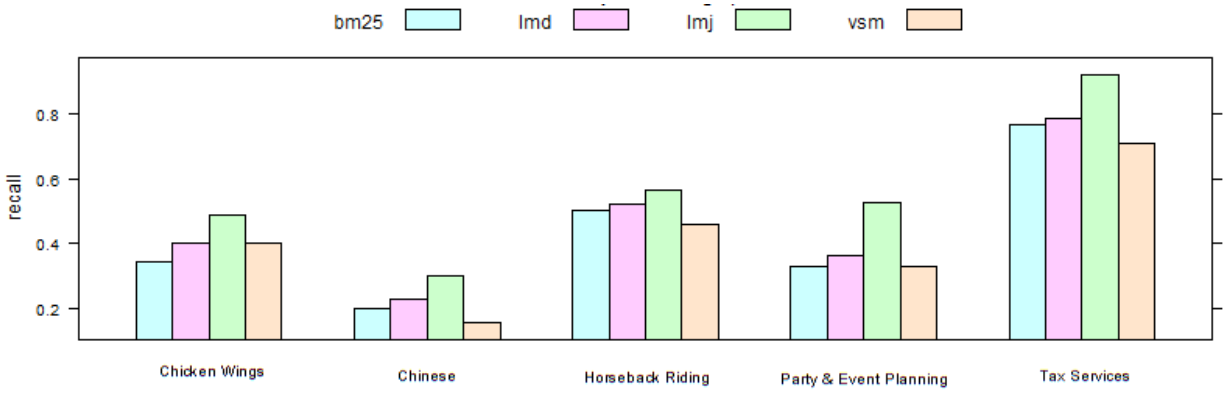

Figure 2: Similarity and Recall Graph

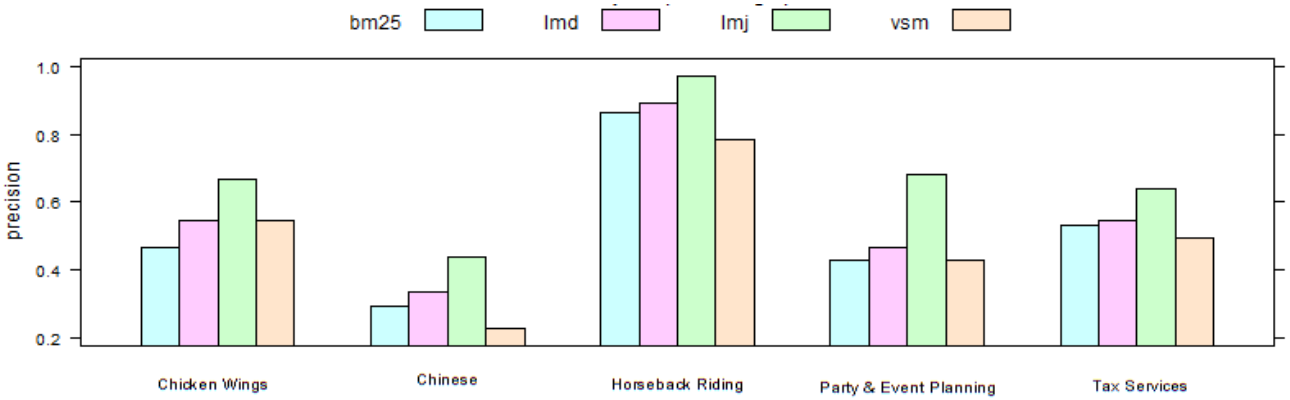

Figure 3: Similarity and Precision Graph

From the plots it is clear that Language Model with Jelinek Mercer Smoothing is performing better than other similarities. Now that we have best performing similarity, for the same randomly picked categories we generated precision and recall graphs by varying maximum number of nouns taken into consideration by our program and again plotted precision and recall to get the best number of nouns to be considered for best results.

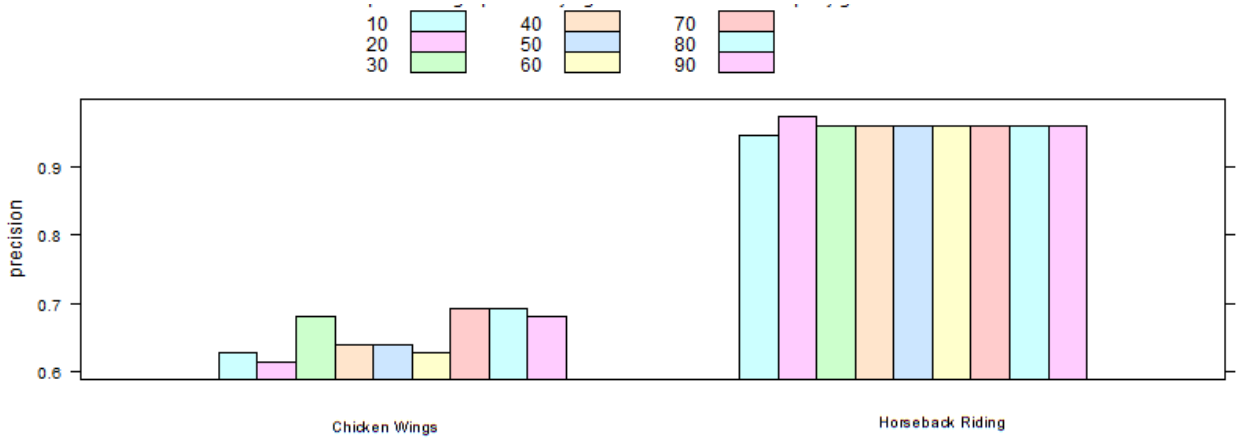

Figure 4: Precision graph for varying maximum nouns taken into query generation from 10 to 100

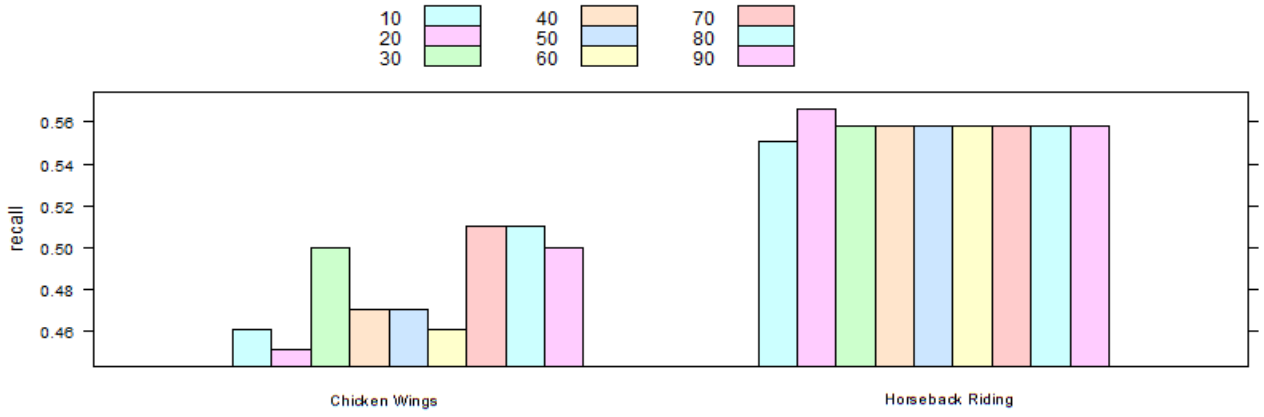

Figure 5: Recall graph for varying maximum nouns taken into query generation from 10 to 100

From above two graphs we can find that precision and recall value is considerable better when the maximum number of nouns used in query are near 70-80.

2. Task: Rating Prediction for Business

2.1. Research Question

Building a recommendation system has been one of the state of the art problem. Powerful recommendation system brings more customers due to its reliability, especially for companies like Yelp whose business has great value for suggesting a quality destinations to its customers. For example, in case of suggesting a restaurant to user is prediction of the possible 'rating' a user can give to particular restaurant. Based on the past history of user, it's possible to predict how much a user will like a particular restaurant in terms of 'rating'. Suppose if a user living in 'California' visited a 'New York' and looking for restaurants in nearby area, based on his/her past history & considering the features related to those particular restaurants, it's feasible to predict the rating user will possibly give. This 'rating 'can be further used a mile stone for building a recommendation system.

The research question we are proposing is "Prediction of restaurant ratings with respect to user". For this problem, class label will be 'rating' which is multinomial value. We considered this a 'regression' problem & build two sets of feature based on two approached we took namely:

a. User-Business attribute model

This model is more constrained to attributes related to 'businesses and 'user' relationship. We focus on identifying patterns from user's past history which determine his/her future preferences.

b. Sentiment Analysis

This model focuses on finding how well we can guess reviews ratings from its text alone using sentiment analysis. Yelp review file has enormous data related to review text and we used this data to perform sentiment analysis to predict user ratings.

Usefulness to Yelp:

The model which can predict the ratings could be part of the recommendation system. Furthermore, it will reduce the feature space for recommendation system results into improvement in performance of the recommendation system.

2.2. Flow Diagram

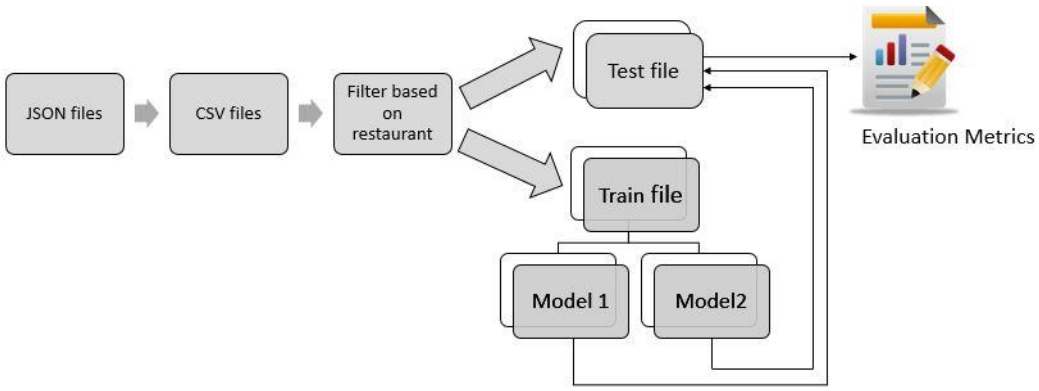

Figure 6: Flow Diagram for Task 2

2.3. Model 1: User-Business attribute model

2.3.1. Method/Algorithm

Steps:

1. Vectorize the class features to get binary class labels (782).
2. Extract features from 'Business' data.
3. Append the vectorized class labels to Business feature extracted in above steps.
4. Build a features using the 'User' data.
5. Filter the features to select features related to 'Restaurant' only.
6. Split feature set into train (66%) and test (34%).
7. Apply 10-fold cross validation on train set and select hyper-parameters for each classifier.
8. Predict on Test set & perform the evaluation.

2.3.2. Experiments and Evaluation

Experiments:

We parse the JSON files and convert it into CSV files, as it helped in preprocessing using Python. As the class labels were 'multinomial', we 'vectorize' the class labels to get binary features. So, we get 782 features by considering all the features .Please find the snippet for example.

vegetarian	sushibars	pizza	mexican	food	sportsbars	asianfusion	bars	thai	breakfast&brunch	delis	seafood	restaurants
0	0	0	0	0	0	0	1	0	0	0	0	1
0	0	0	0	0	0	0	1	0	0	0	0	1
0	0	0	0	0	0	0	1	0	0	0	0	1
0	0	0	0	0	0	0	1	0	0	0	0	1

Figure 7: Vectorized class labels for binary features

From 'Business' file we extracted features like 'attributes_Ambience_casual', 'attributes_Ambience_romantic', 'attributes_Ambience_upscale' which we find more relevant for our task. From 'user' file we selected the features that will attribute the user. All these feature were filtered out to get only 'restaurant' related features. This lead to total 73 features and around 1 Million records. Please find the snippet of the final feature set.

user_id	business_id	day	month	year	votes_cool	votes_funny	votes_useful	attributes_Ambience_classy	attributes_Ambience_touristy	italian	sandwiches	american(new)	nightlife	label
LWbYpcangjBMm4KPxZGOKg	mVHrayjG3uZ_RLHkLj-AMg	1	12	2012	0	0	5	1	1	0	0	1	1	5
m1FpV3EAeggaAdfPx0hBRQ	mVHrayjG3uZ_RLHkLj-AMg	15	3	2013	0	0	0	1	1	0	0	1	1	5
8fApIAMHn2MZJFUiCQto5Q	mVHrayjG3uZ_RLHkLj-AMg	30	3	2013	1	0	2	1	1	0	0	1	1	5
uK8tzraOp4M5u3uYrqIBXg	mVHrayjG3uZ_RLHkLj-AMg	20	10	2013	0	0	1	1	1	0	0	1	1	4
6wvlM5L4_EroGXbnb_92xQ	mVHrayjG3uZ_RLHkLj-AMg	7	11	2013	0	0	0	1	1	0	0	1	1	5
u9ULAsnYTdYH65Haj5LMSw	mVHrayjG3uZ_RLHkLj-AMg	29	9	2014	0	0	0	1	1	0	0	1	1	4
pdHC0oAcG7gNdhuFRAUu0Q	mVHrayjG3uZ_RLHkLj-AMg	29	9	2014	0	0	0	1	1	0	0	1	1	5

Figure 8: Final feature set

This feature set was divided into Train (size: 640K) and Test (size: 340K) sets. We performed 10-fold cross validation using following classifiers.

- Gradient Boost Regressor
- Guassian SVM
- Sigmoid SVM
- Decision Tree Regressor
- Random Forest Regressor

Evaluation:

Regression with 10 fold cross validation		
Leaner	RMSE	R2 score
Gradient Boost	1.2147	-0.0968
Naive Bayes	1.6135	-0.5953
SVM with RBF kernel	1.3434	-0.1057
Decision Tree	1.2318	0.0713
Logistic Regression	1.4491	-0.2837
Random Forest	1.2279	0.0769

Figure 9: Regression with 10 fold cross validation

As a part of evaluation metric for this task we choose Root mean square error and R2 metric. Closely examining the results we find that Random forest and decision tree performed better than other algorithms (Gradient boost, Naïve Bayes, SVM with RBF kernel, Logistic regression).

2.4. Model 2: Sentimental Analysis Model

2.4.1. Method/Algorithm

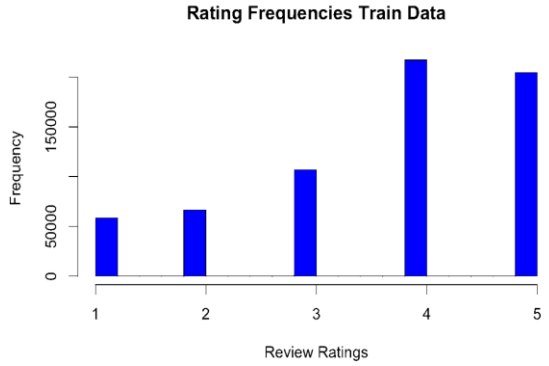

Figure 10: Rating frequencies in Train data

As we can see from adjacent figure rating 4 and rating 5 represent most of the data as compare to rating one and rating two. If we extract features directly on whole data then feature set may contain maximum features related to ratings four and five. In order to avoid this we divided train data further based on review ratings.

Features generated from each rating were then combined together to form a single feature file for prediction.

1. Feature Generation

Since review data is free text written by multiple users we first processed text data using Lucene and followed below steps.

a. Removed stop words.
b. Generate tokens using unigrams/bigrams.

Frequencies of all generated features are then used to decide top features for each rating value. Below diagram shows word cloud for generated features for rating one and rating five. Overall we can see rating one contains negative attitude features whereas rating five have various features expressing positive attitude or happiness.

Top selected features for each rating were then combined together and then filtered to remove noise to get final feature set.

Figure 11: Bigram features for rating One

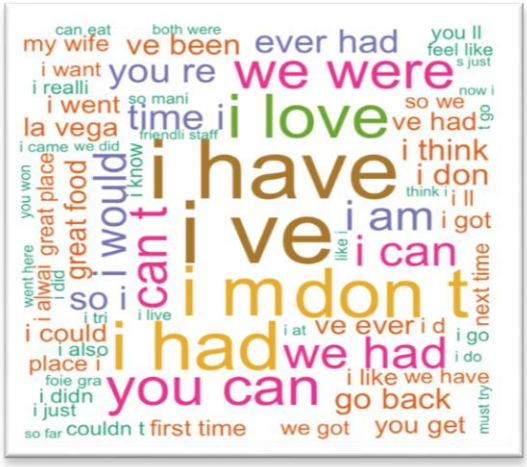

Figure 12: Bigram features for rating Five

2. Prediction Algorithms

We used various regression algorithms such as support vector machines and Random Forest (CART Family) to predict review ratings using generated features.

2.4.2. Experiments and Evaluation

Experiments:

1. Features

Features were generated using various methods such as unigrams, bigrams and combination of unigrams and bigrams. As per experimental results, we found that root mean square value (RMSE) is minimum for unigrams and bigrams combined together. On the other hand bigrams gave us maximum RMSE value. Unigram features performed better than bigrams for both random forest as well as support vector machine.

2. Algorithms

For algorithms, we tried regression model for CART family algorithm Random Forest and also for Support Vector Machine (SVM). Although RMSE values for both algorithms are almost equal SVM performed slightly better as compare to Random Forest.

Evaluation:

As a part of evaluation for task 2, model 2 we choose RMSE. Below table has RMSE results obtained for Random forest and Support Vector Machines. In the table low value of RMSE for SVM indicates SVM performed better than Random forest. Also combination of unigram and bigram features performed better as compare to unigrams and bigram features alone.

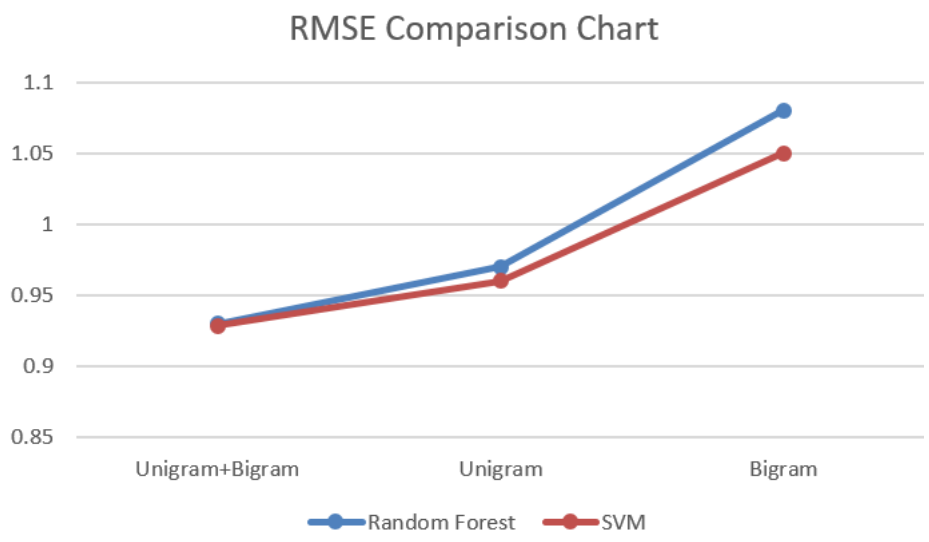

Figure 13: Regression RMSE Comparison Chart

3. Tool-kit and API

- Stanford POS Tagger
- Mongo DB
- Java
- Lucene
- R
- Python-scikit

4. Analysis and Conclusion

4.1. Task1

We analyzed precision and recall for randomly selected categories and found that we get the best results when the count of query words ranges between70-80. Additionally, we tested performance for various algorithms and we see best results for Language Model with Jelinek Mercer Smoothing algorithm when assigning top 75 business to each category.

4.2. Task2

For Model 1 algorithms for Decision tree and Random Forest from CART family performed better as compare to other algorithms. For Model 2 combination of unigrams and bigram features performed better as compare to unigrams or bigrams alone. Also SVM

performed better as compare to random forest. When we compared both models, RMSE for model 2 was better as compare to Model 1.

5. Future Scope

5.1. Task1

We saw that the generated queries for categories had noisy words like des, und, ich, etc. which got selected as part of query due to their uniqueness in the corpus (resulting in high TF*IDF score). As a future scope, additional filtering can be introduced to remove such words to obtain a more meaningful and relevant query.

Additionally, the generated query was a result of a single iteration over the reviews and tips per category. The query obtained in this step can be fed back to Lucene in more iterations to get obtain a better query.

5.2. Task2

Currently model 1 and model 2 preform independent of each other. We think ensemble of model 1 and model 2 may give us better result in future.

6. References

[1] Deshmukh, Renuka, Venkata Prudhvi Raj Indana, Aniket Sambhaji Gaikwad, Nayana Yashwant Charwad, and Sri Laxmi Chintala. "Yelp-Dataset-Challenge-2015." GitHub. Accessed December 14, 2015. https://github.iu.edu/renudesh/Yelp-Dataset-Challenge-2015.git.

[2] "3.3. Model Evaluation: Quantifying the Quality of Predictions." 3.3. Model Evaluation: Quantifying the Quality of Predictions — Scikit-learn 0.17 Documentation. Accessed December 14, 2015. http://scikit-learn.org/stable/modules/model_evaluation.html.

[3] "Documentation of Scikit-learn 0.17." Documentation Scikit-learn: Machine Learning in Python — Scikit-learn 0.17 Documentation. Accessed December 14, 2015. http://scikit-learn.org/stable/documentation.html.

[4] "Quick-R." Accessed December 14, 2015. http://www.statmethods.net/advstats/cart.html.

[5] "Package ‘randomForest’." Accessed December 14, 2015. https://cran.r-project.org/web/packages/randomForest/randomForest.pdf.

[6] "Package ‘e1071'" Accessed December 14, 2015. https://cran.r-project.org/web/packages/e1071/e1071.pdf.