Topic Modeling and Machine Learning on Hotel Review Data

Yanhua Hou
Mentor: Alex Rutherford

Data Science Intensive Capstone Project

Github: https://github.com/phyhouhou

Outlines

- Introduction
- Data Information and Data Cleaning
- Exploratory Data Analysis
- Machine Learning Models for Predictions
- Conclusions and Future Work

Hotel Reviews

- More and more people are using online reviews in making important decisions, i.e., where to stay, where to eat, …
- A massive amount of reviews are being posted online for sharing opinions or experiences, too many to read through manually.
- What are previous customers saying about hotels?
- Do they like or hate the hotels based on what they say?

Clients

Owners, customers, travel intelligences…

Data Information

- Data is from kaggle. It's about 515k+ reviews data for 1493 luxury hotels across Europe from 08/2015 to 08/2017 with 17 features.

- Hotel features : '*Hotel_Name'*, '*Hotel_Address*', '*lat*', '*lng*', '*Average_Score*','*Total_Number_of_Reviews*', etc.

- Reviewer features: *Reviewer_Nationality*', '*Total_Number_of_Reviews_Reviewer_Has_Given*', etc.

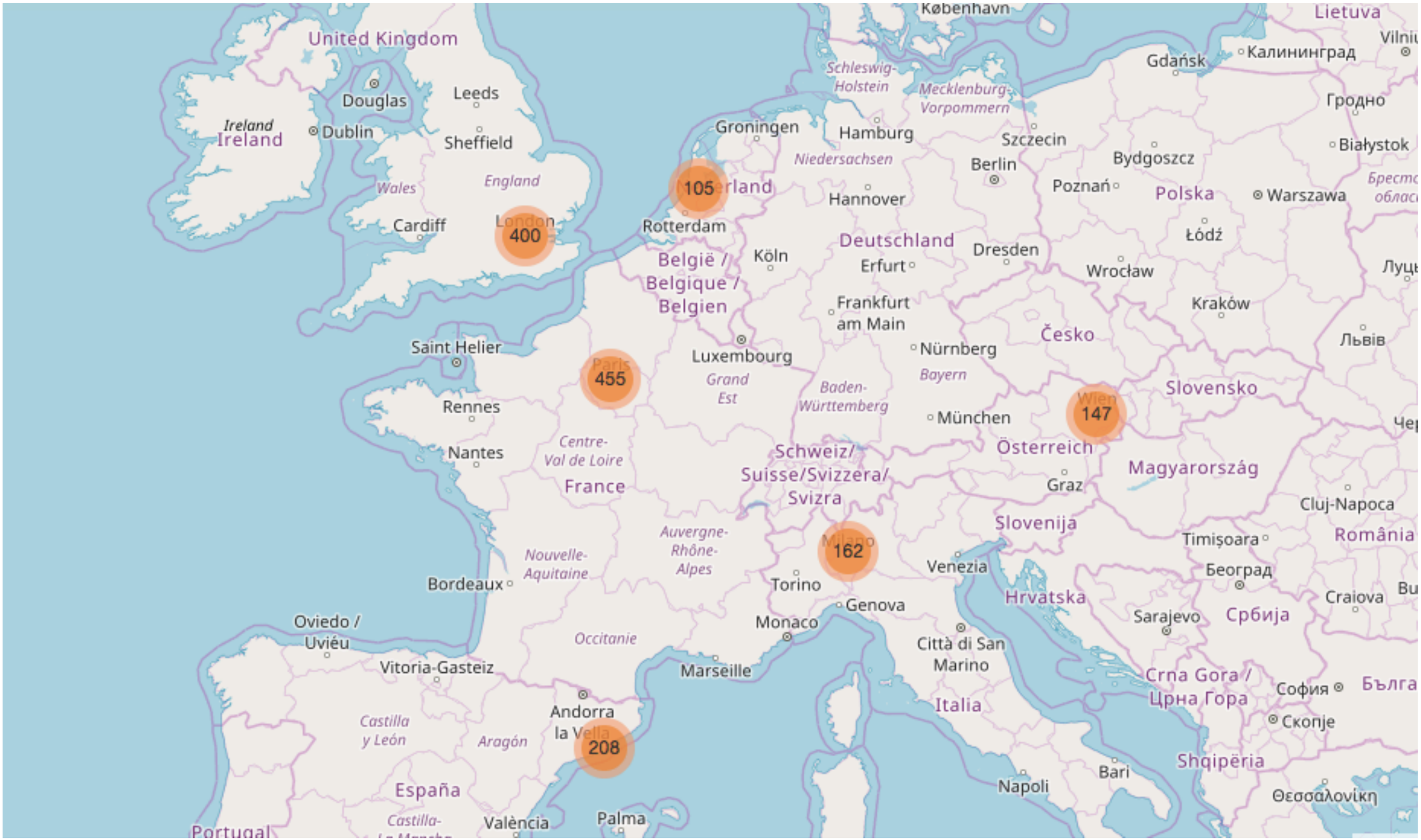
- Review features: '*Review_Date*', '*Negative_Review*', '*Positive_Review'*, 'Review_Score', 'Tags', etc.

Data Wrangling

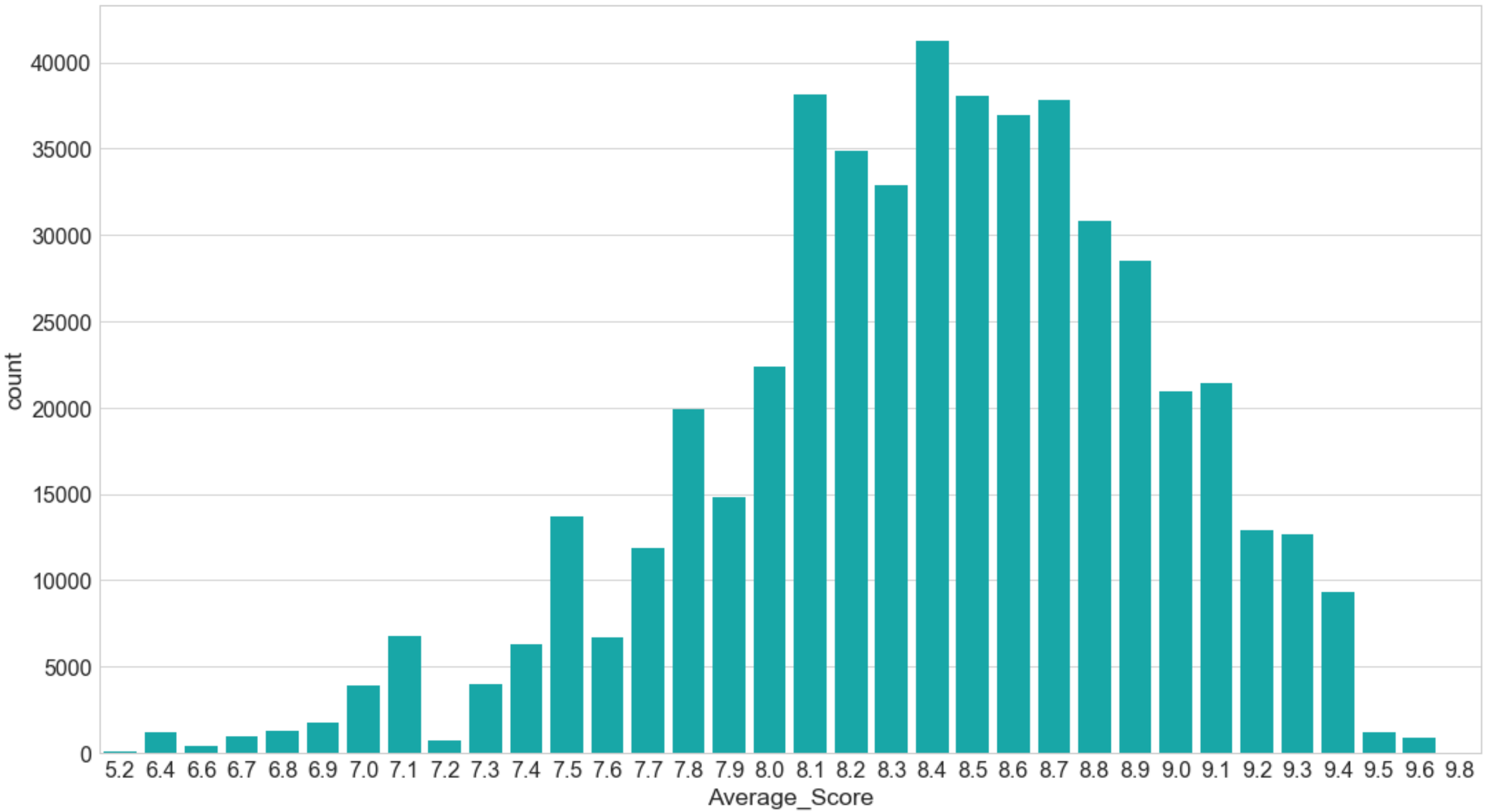
- Missing values and duplicates;
- Extract cities of hotels from its address, i.e., ‘Hotel_Address’.
- Add 'Pos_Rev_WCRatio' and 'Neg_Rev_WCRatio';
- Extract features from ‘Tag’, i.e., ‘Trip_Type’, ‘Traveler_Type’, ‘Num_Nights’;
- Preprocess review texts, i.e., remove stopwords, whitespaces, non-letters characters, word tokenization and lemmatization.

Visualizations

Visualization of Hotels

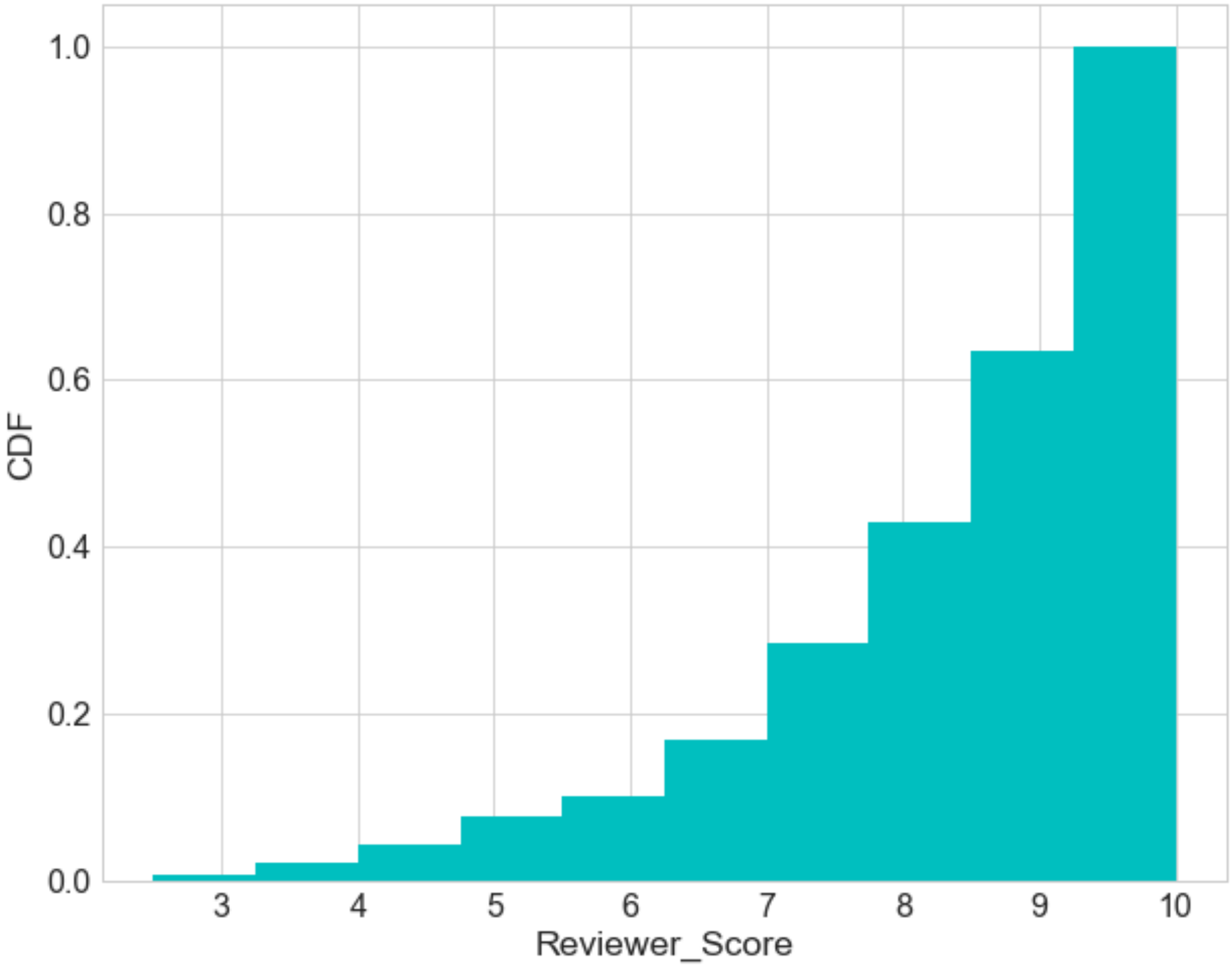


Average Score of Hotels

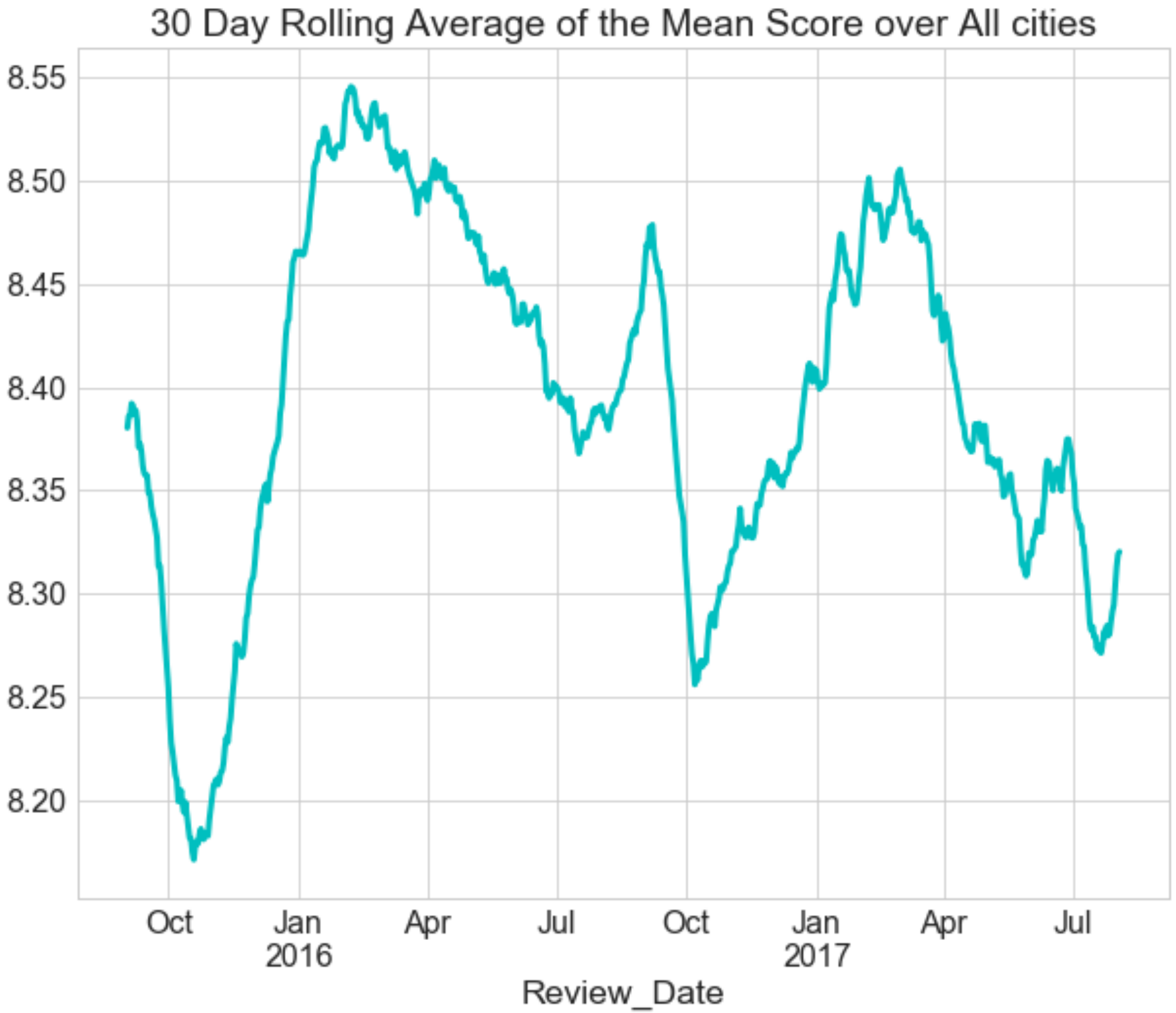


All average scores are above 5.0, most are above 8.0

CDF of Reviewer Score

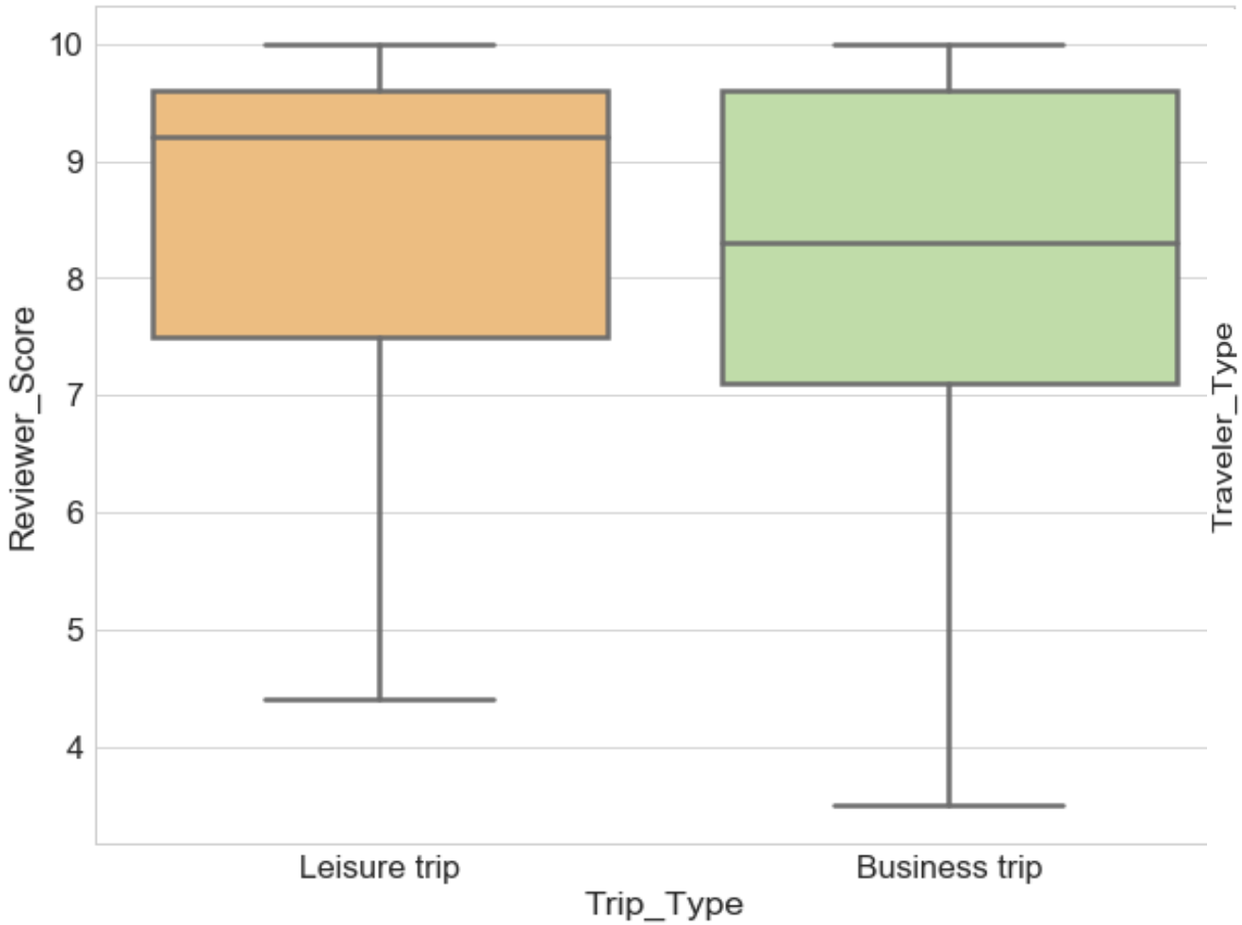


A majority of scores are high except a minority of very low scores.

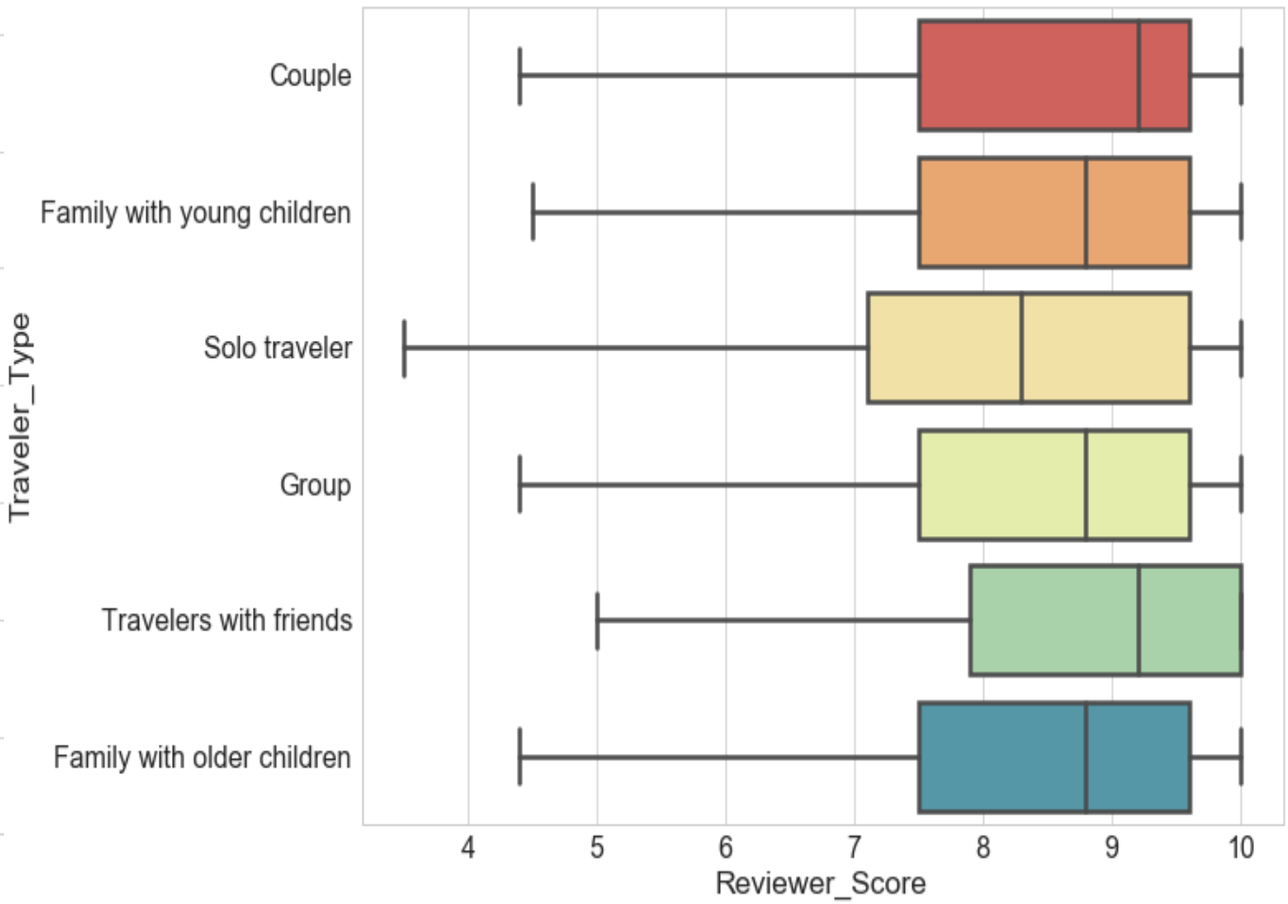


The average reviewer score of hotels is higher in January and lower in October.

What factors affect scores?



Reviewers on a leisure trip tend to rate higher than those on a business trip.



‘solo traveler’ tends to rate lower than ‘couples’.

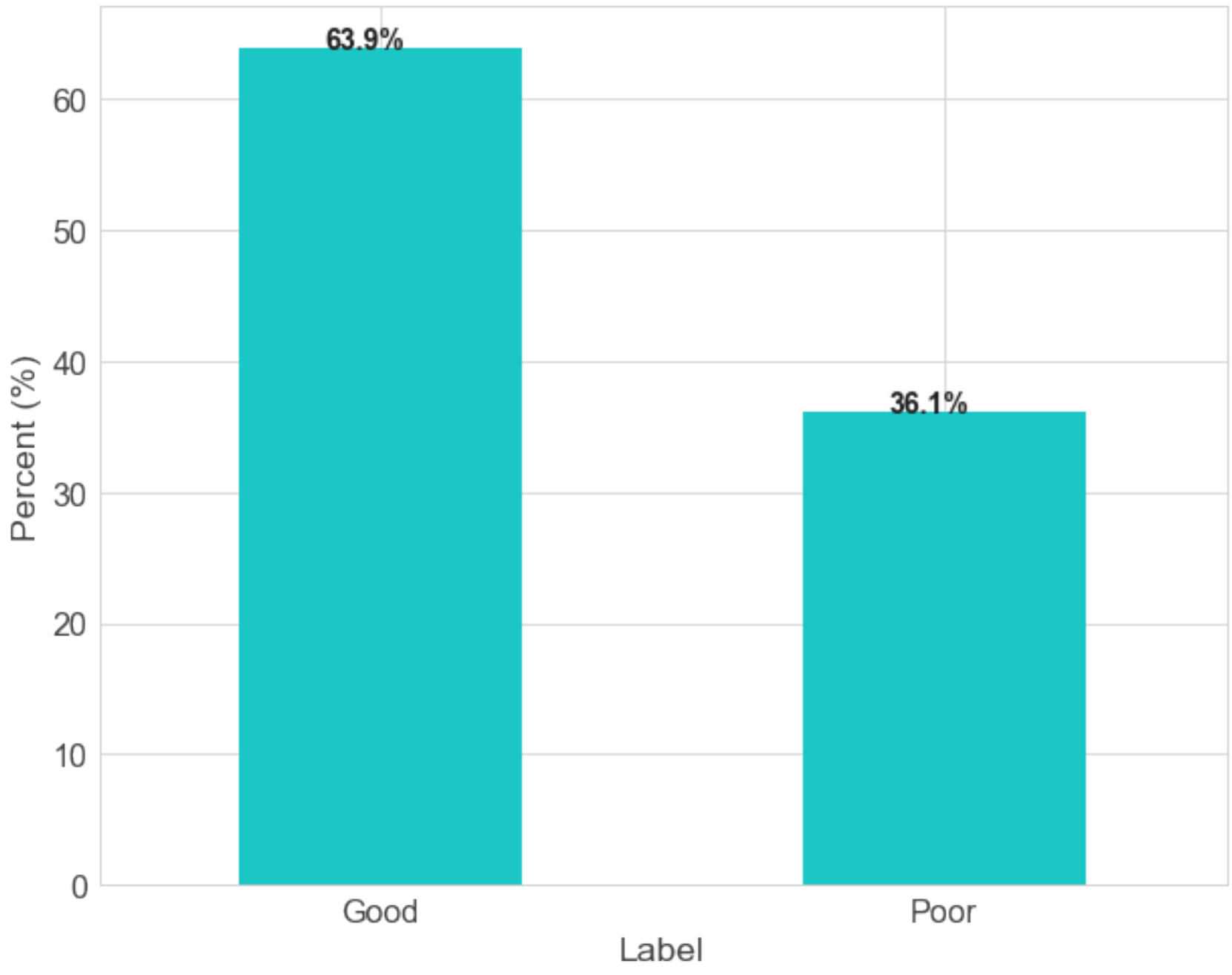
Word Cloud of Reviews

Most frequent words indicates content on location, staff, facilities, food, cleanliness, comfort, value for money (price)…

Classifier

Can we build a model to tell if reviewer likes or dislikes hotels based on what they posted?

Classes



Add label: ‘poor’ (reviewer_score<8), ‘good’ otherwise

Machine Learning Model for Predictions

- Feature engineering / feature selection
- Model comparison/evaluation
- Model optimization

Classification Steps

Data pre-processing steps:
1. Split texts into training and testing sets (70-30 splits)
2. Vectorize text: Text->Features->Labels
3. Add supplemental features:'Trip_Type','Len_LemRev_char'…

Build ML pipelines to train classifier on predicting labels of reviews

Test classifier on unseen test set and compare model performance

Evaluate and optimize model performance

Topic Modeling with LDA

Display Top 20 Keywords in Topics

Topic 0: [facilities]

bed room bathroom shower comfortable small comfy clean nice water good location pillow big bath size great coffee double large

Topic 1: [staff, food; positive]

staff breakfast room location friendly helpful good great excellent clean nice bar food comfortable service lovely really restaurant facility stay

Topic 2: [location]

location close walk station good great city metro nice restaurant pool minute easy room area clean train centre value parking

Topic 3: [reception service, negative]

stay staff check room time day book make pay ask service reception night say charge come tell help leave extra

Topic 4: [room, noise; negative]

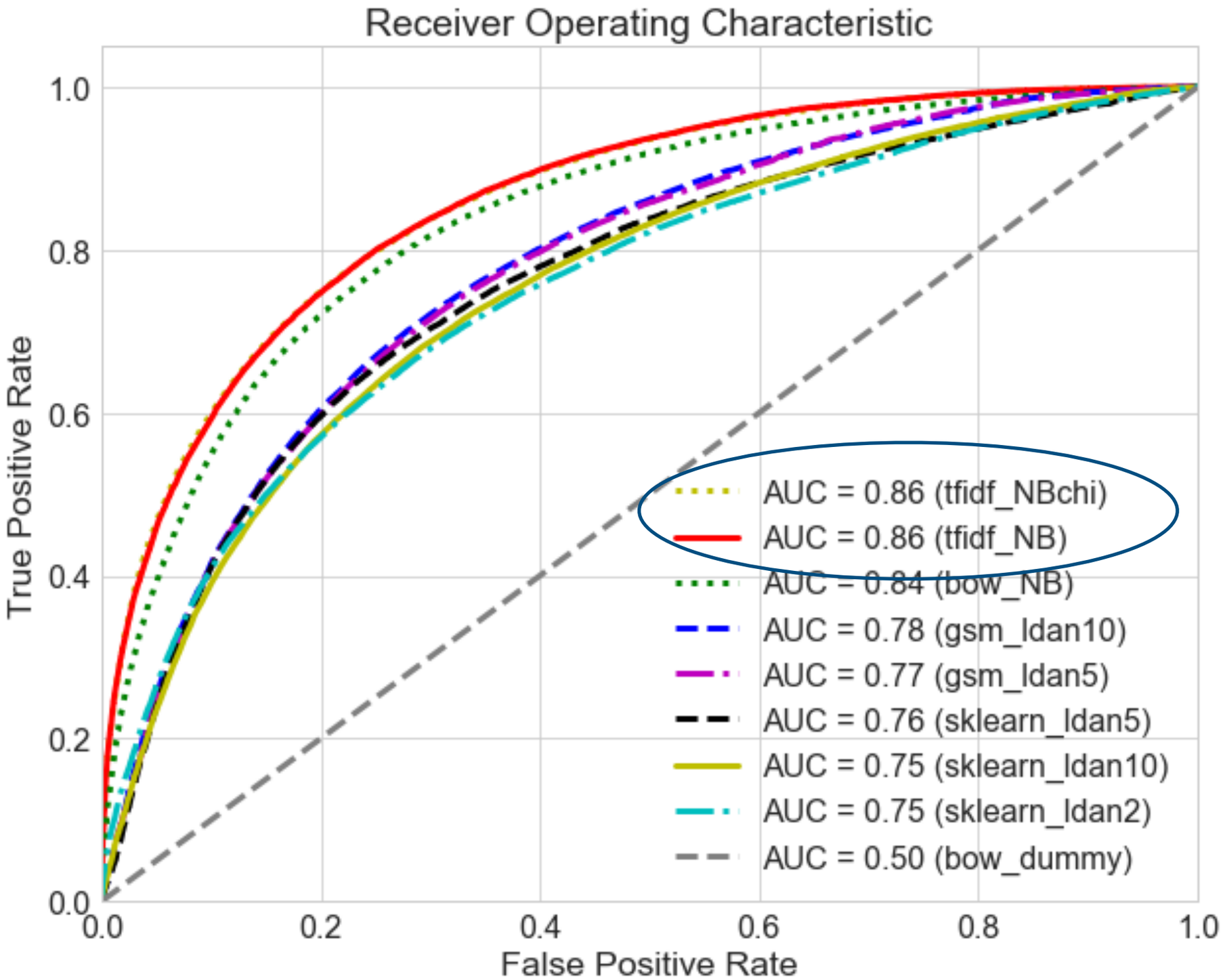
room location good work night breakfast floor window air door noise star open noisy small old bad need clean poor

Display a Document - Topic Table

	Topic0	Topic1	Topic2	Topic3	Topic4	dominant_topic
Doc0	0.03	0.00	0.07	0.47	0.43	3
Doc1	0.12	0.45	0.11	0.32	0.00	1
Doc2	0.20	0.41	0.01	0.12	0.26	1
Doc3	0.08	0.00	0.06	0.25	0.60	4
Doc4	0.00	0.00	0.06	0.68	0.26	3
Doc5	0.01	0.38	0.30	0.16	0.14	1
Doc6	0.89	0.01	0.08	0.01	0.01	0
Doc7	0.01	0.88	0.09	0.01	0.01	1
Doc8	0.22	0.02	0.02	0.34	0.41	4
Doc9	0.27	0.22	0.25	0.01	0.26	0

Model Performance:

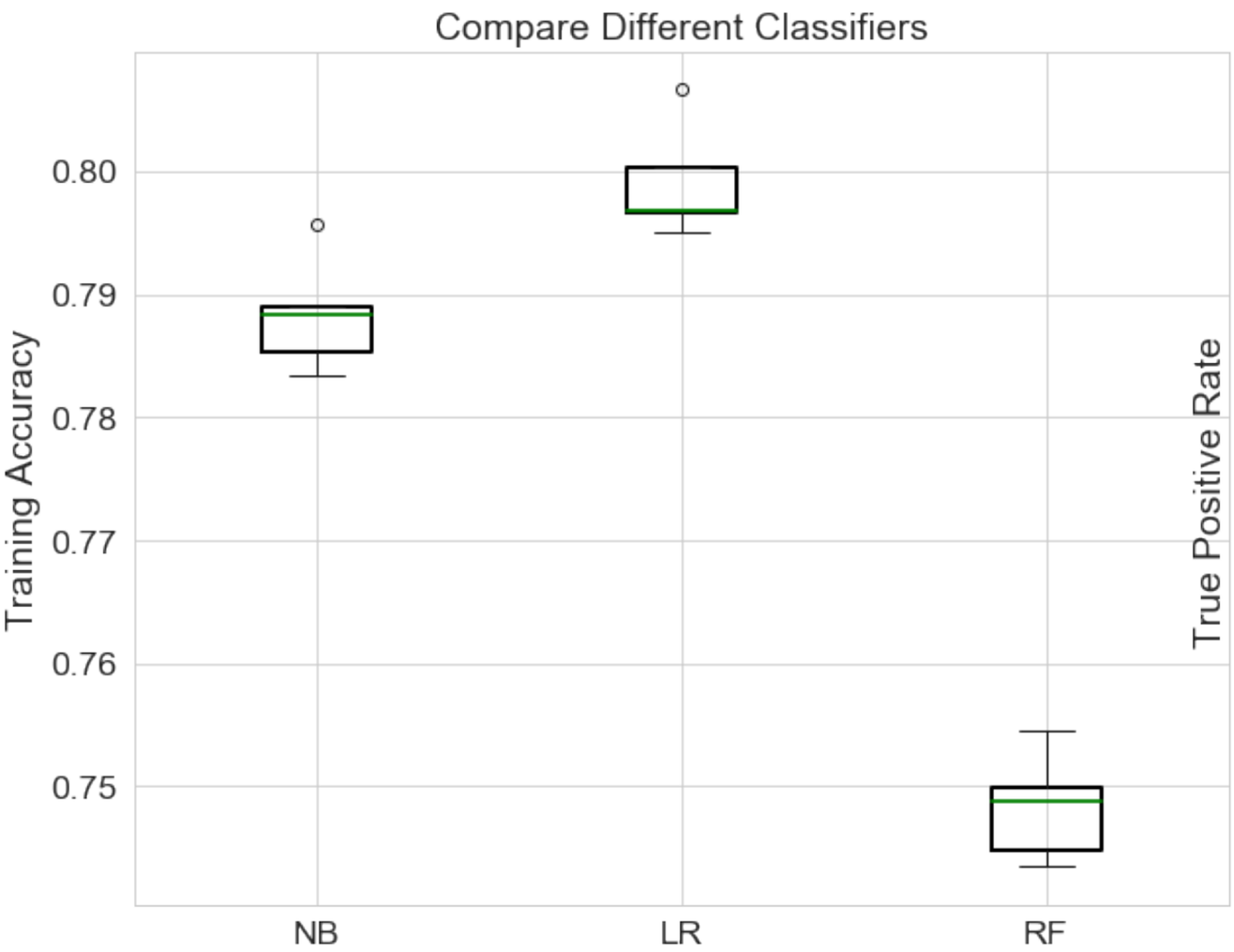
Text Features

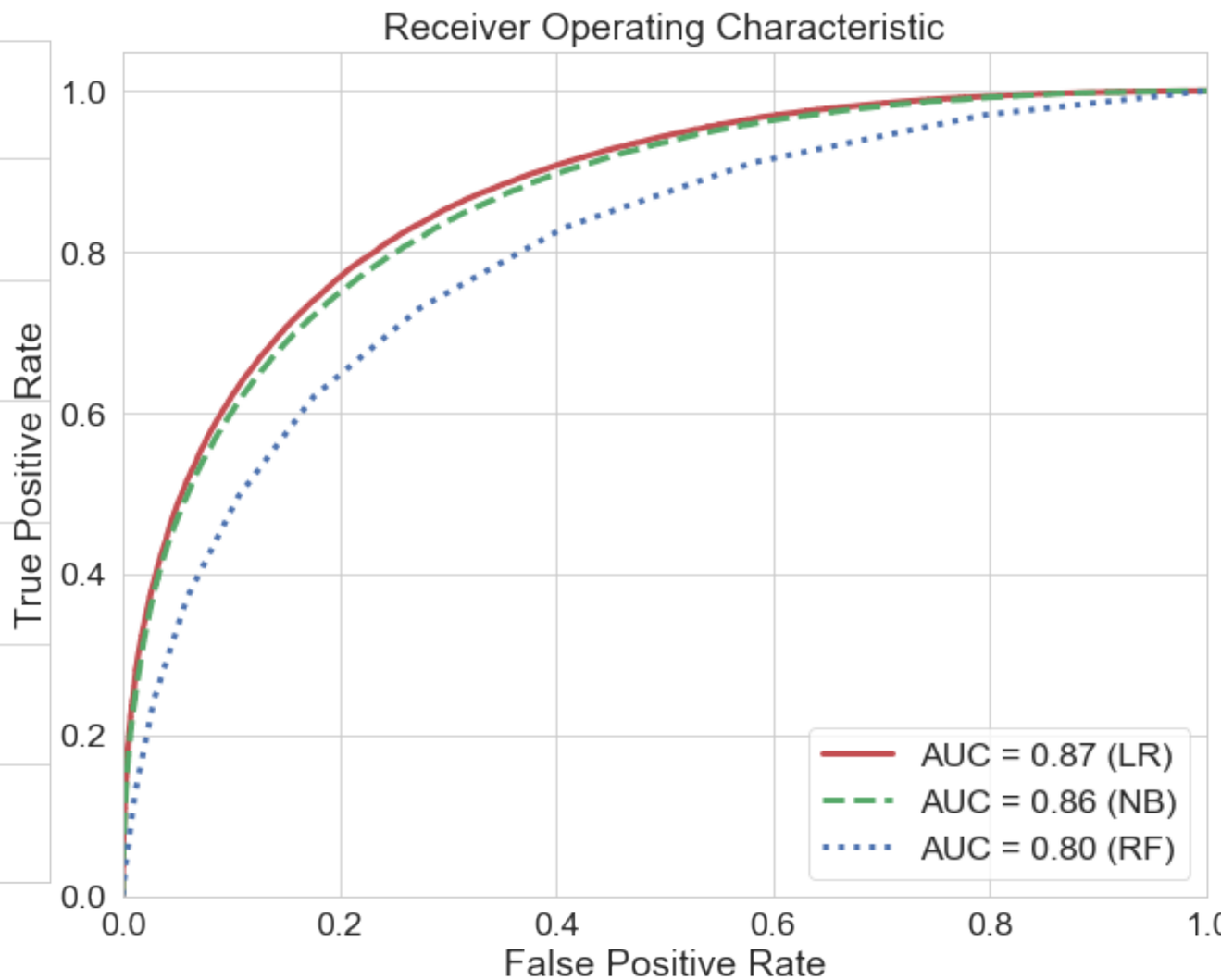


Concerning texts, the TF-IDF vectorized features as inputs give the best model performance.

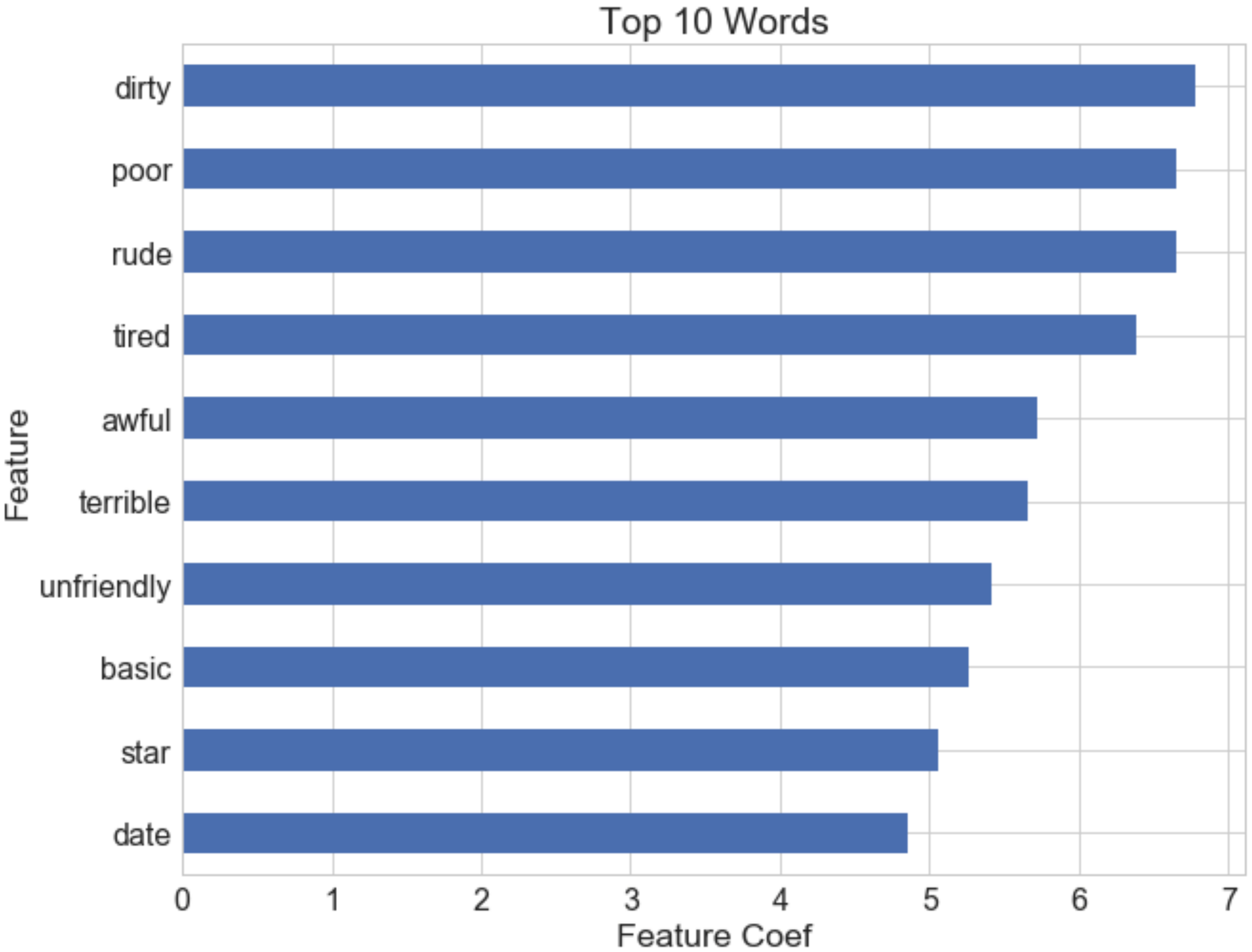
Compare Classifiers

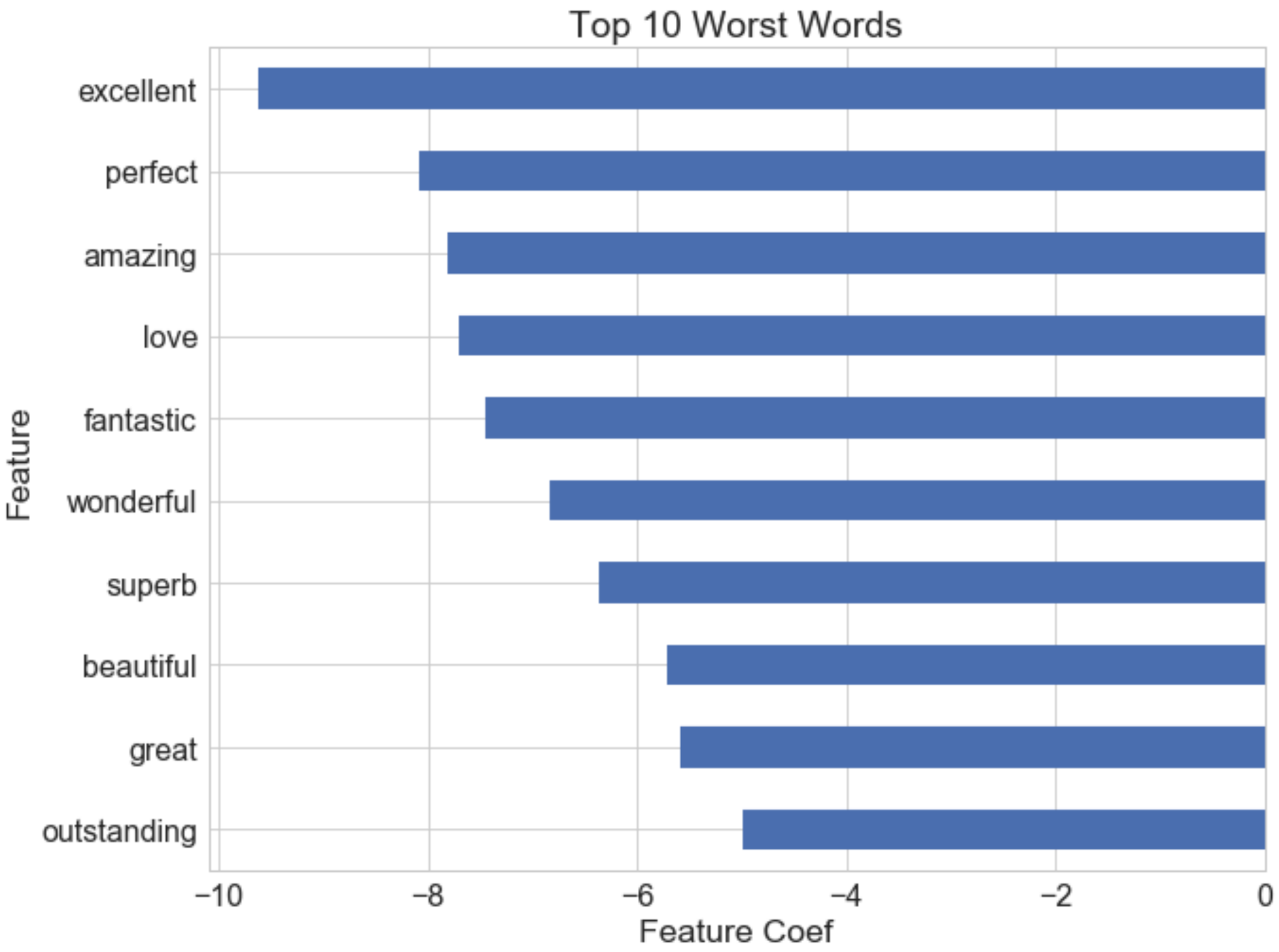
Take TF-IDF vectorized features to represent review texts.





LR: LogisticRegression, NB: MultinomialNB, RF: RandomForestClassifier

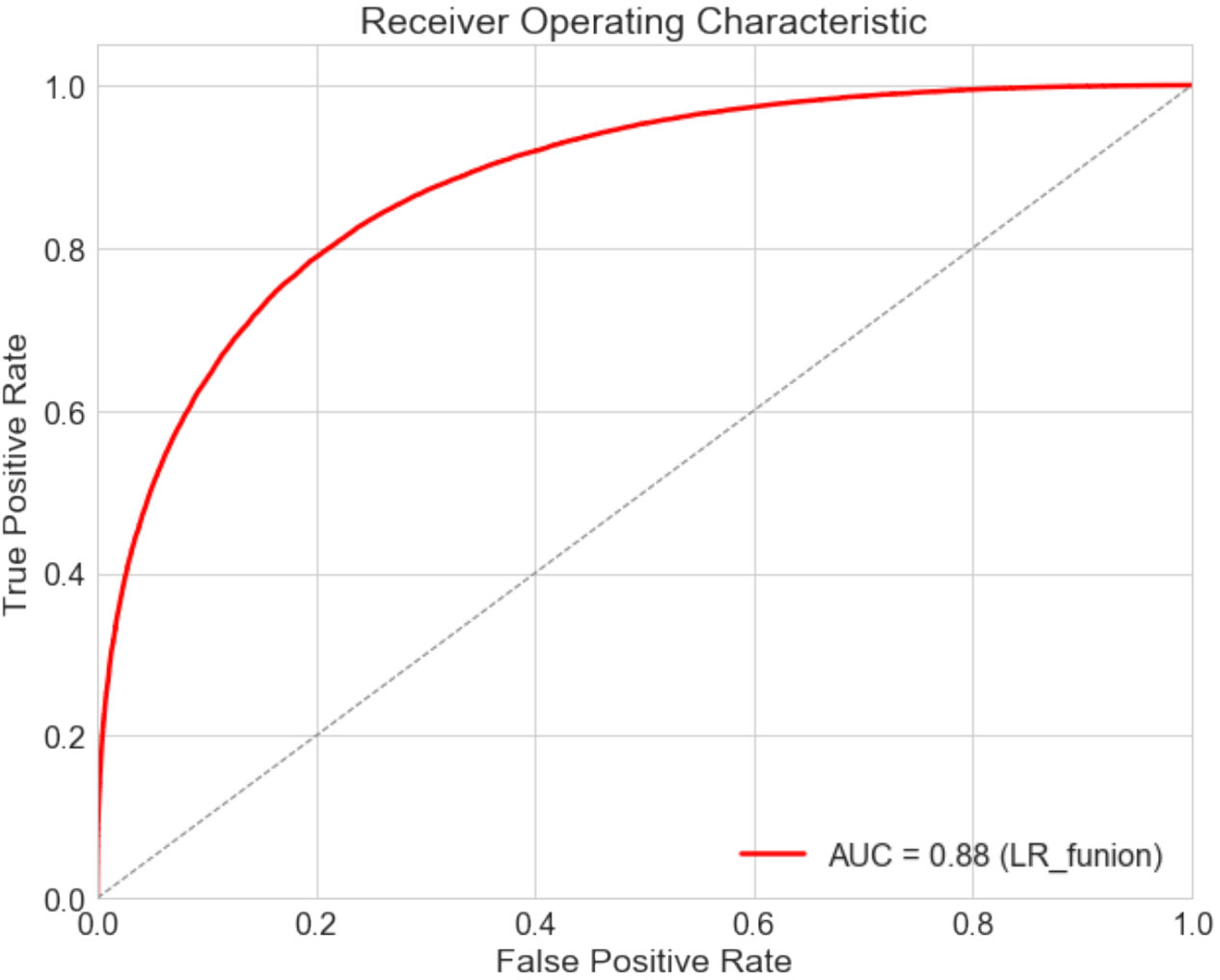

Top 10 Words
dirty
poor
rude
tired
awful
terrible
unfriendly
basic
star
date
Feature
0
1
2
3
4
5
6
7
Feature Coef


Top 10 Worst Words
excellent
perfect
amazing
love
fantastic
wonderful
superb
beautiful
great
outstanding
Feature
−10
−8
−6
−4
−2
0
Feature Coef

Further Improvement

- Grid search hyperparameters for LogisticRegression classifier

- Enrich predictors with categorical and numerical features via FeatureUnion



Classification Report

	precision	recall	f1-score	support
Good	0.83	0.88	0.85	82592
Poor	0.76	0.68	0.72	46248
avg / total	0.8	0.81	0.8	128840

Conclusions

- We've performed data wrangling and exploratory analysis on hotel review data regarding hotels, reviewers and reviews.

- We've implemented topic modeling on review texts. The topics are not as separable as expected.

- We've conducted NLP analysis and generated vectorized (BOW, TF-IDF, LDA topics) text features.

- Out of different types of vectorized features TF-IDF weighting works best.

- With TF-IDF features as input, logistic regression classifier performs best. With 70%-30% splitting, the test data set gave ROC AUC=0.88

Next Steps

Improve performance:

- Improve text cleaning, I.e., correcting mis-spelling, including n-grams
- Add interactions between features
- Resample to tackle imbalanced dataset

Other interesting questions include:

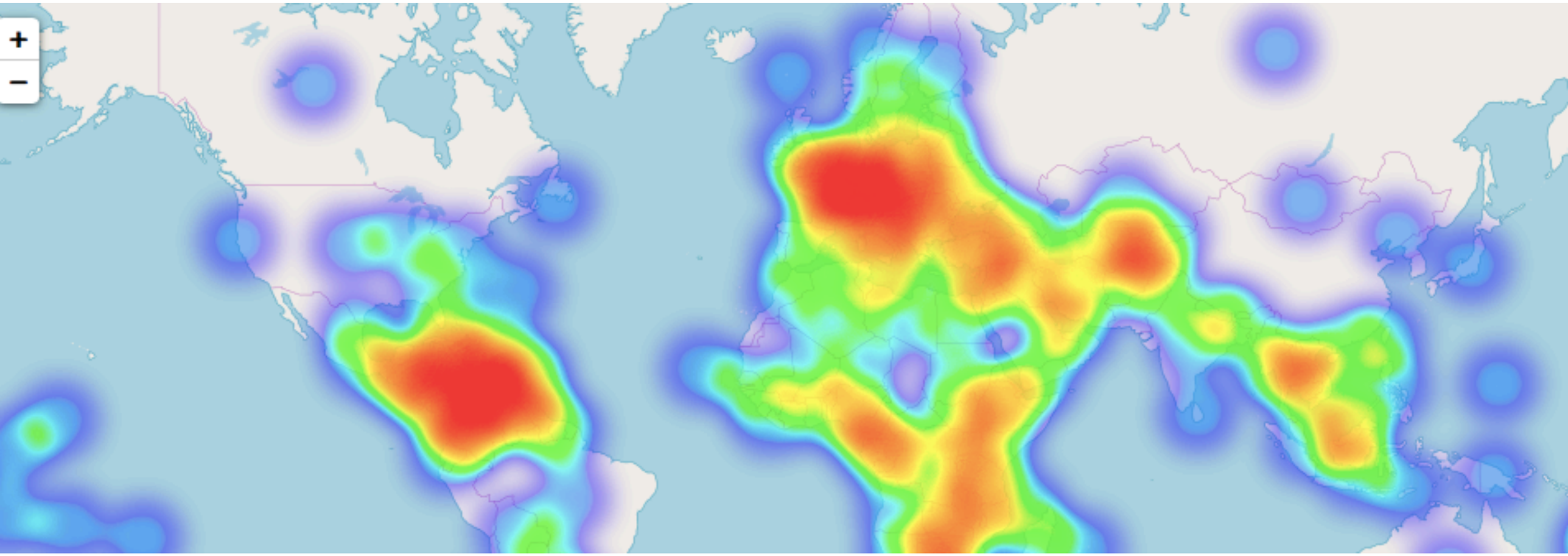
- Extract topics by months to explore if there is some trend in topics in a time series.
- Adjust the model and tune to a potential client.
- …

Thank you!

Q&A

Visualization of Reviewers

Where are reviewers from?



Redder indicates more reviewers. While hotels are located in Europe, many reviewers are from America and Europe.

Classifier Ingredients: Vectorizer

- Classifier interacts with features, not the text itself.

- Vectorizer or feature extractor transforms a text into quantifiable information about the text.

- Common word feature extractors:
  - Bag-of-Words (word counts)
  - TF-IDF weighting
  - LDA model topics

Bag-of-Words (BOW) Representation

Bag of words is a kind of feature extraction where:

- The set of features is the set of words in the text.
- A single text is represented by how many of each word appears in it.

Simple example

Original text is
Hop on pop
Hop off pop
Hop Hop hop

Words for each feature:
['hop', 'off', 'on', 'pop']

Transformed text vector is
[1 0 1 1]
[1 1 0 1]
[3 0 0 0]

	hop	off	on	pop
Hop on pop	1	0	1	1
Hop off pop	1	1	0	1
Hop Hop hop	3	0	0	0

Bag-of-Words (BOW) Representation

Use CountVectorizer to create document-word matrix:

```
CountVectorizer(     min_df=10,    #words have occurred at least 10 times
                     stop_words='english',
                     lowercase=True,
                     token_pattern='[a-zA-Z]{3,}', # char length at least 3
                   )
```

Shape of Sparse Matrix: (429464, 10670)
Amount of Non-Zero occurrence: 6941938
sparsity: 0.15%

TF-IDF Weighting Representation

- Term-Frequency * Inverse Document Frequency (weighted by the inverse of its popularity in all documents).
- TF-IDF is essentially a measure of term importance, and of how discriminative a word is in a corpus.
- TF-IDF weighted features can be used as inputs to any classifier.

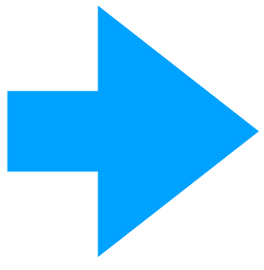
Simple example

Original text is
Hop on pop
Hop off pop
Hop Hop hop

IDF: ‘hop': 1.0, 'off': 1.69, 'on': 1.69, 'pop': 1.29

Transformed text vector is

[1 0 1 1]
[1 1 0 1]
[3 0 0 0]

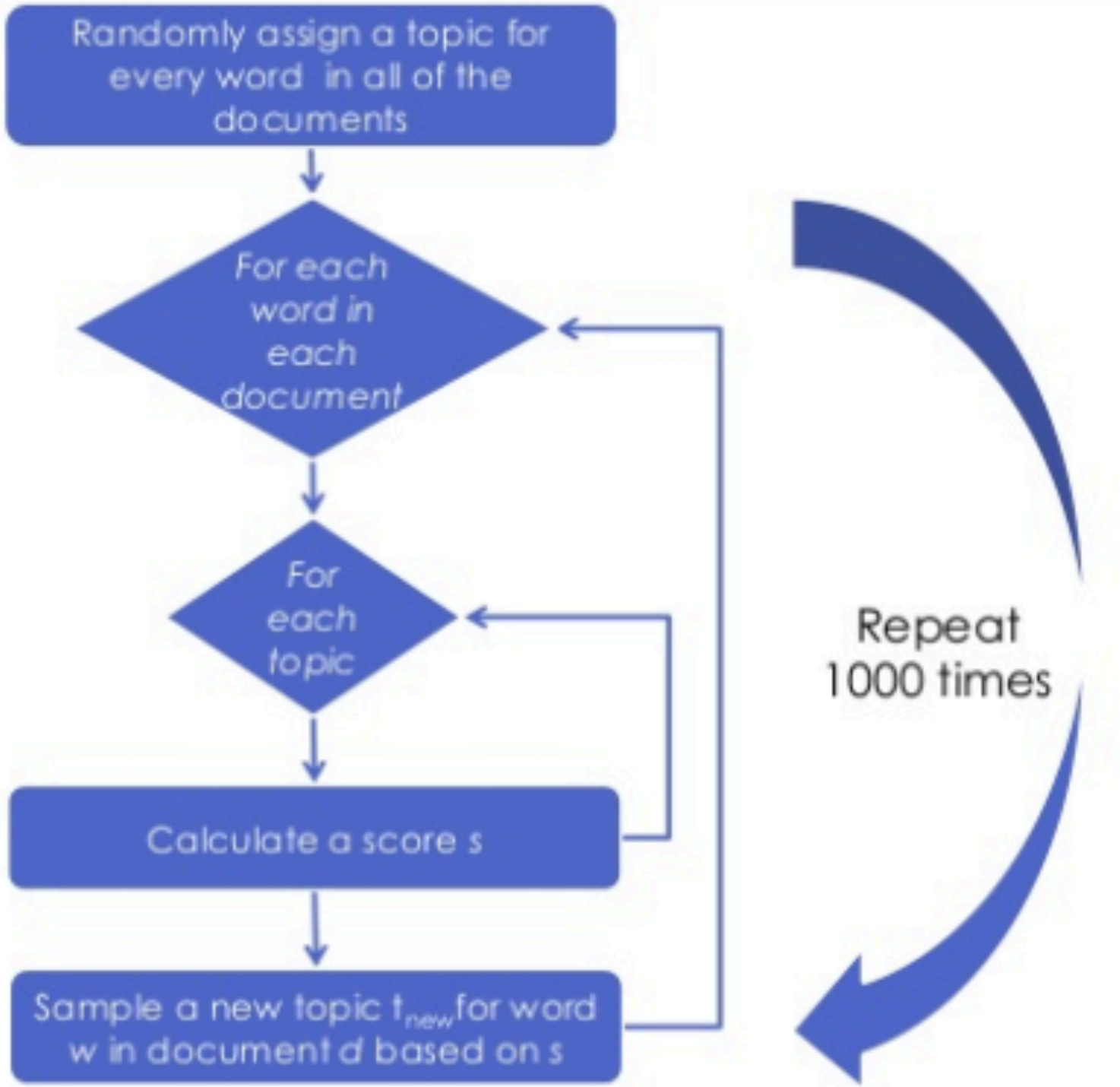


[1 0. 1.69 1.29]
[1 1.69 0. 1.29]
[3 0. 0. 0.]

LDA Topics Representation

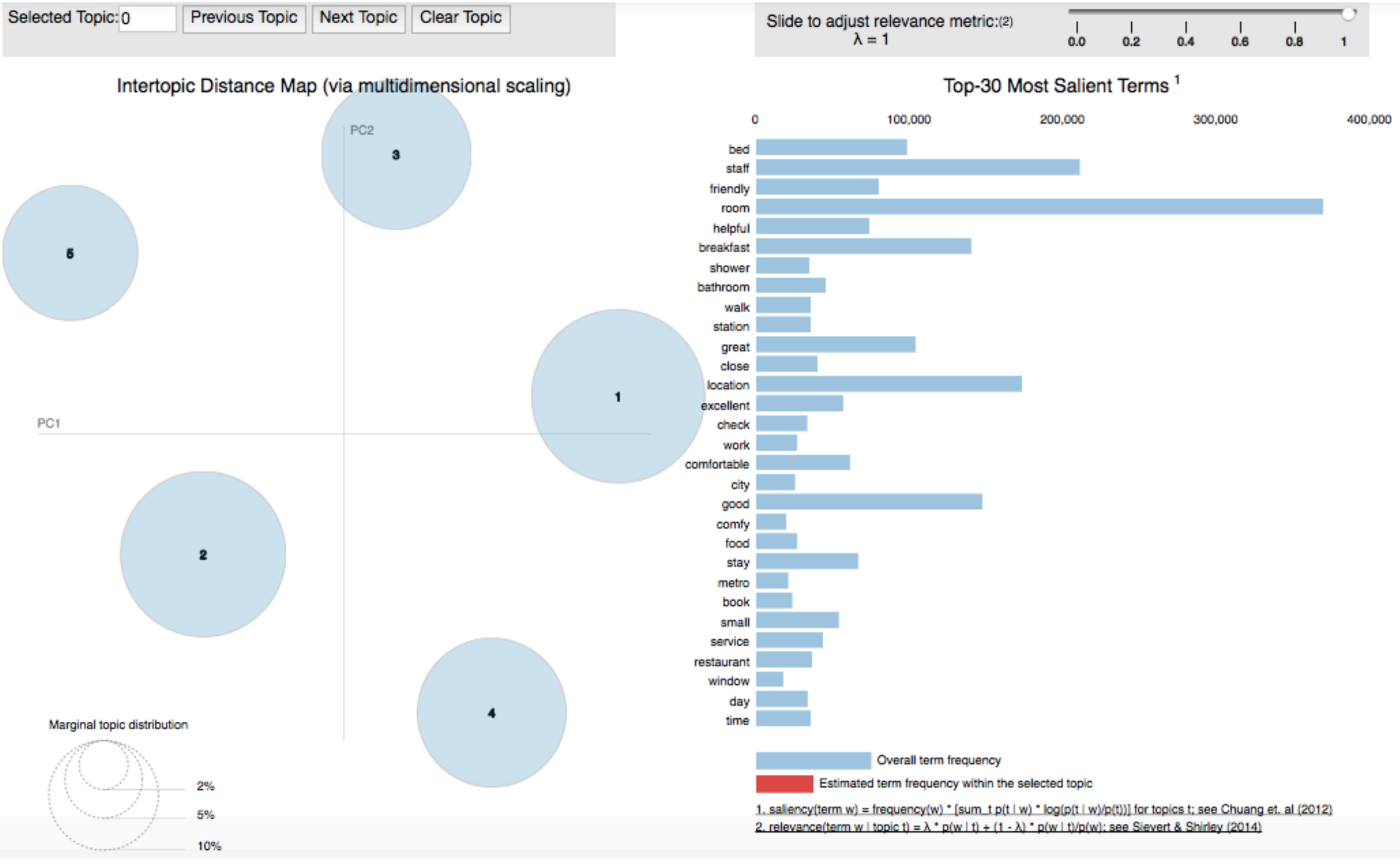
- Group words into topics applicable for prediction
- Each document is represented by a vector of its topic portions.
- Num_topics needs to be determined empirically
- Provide dimension reduction(10670 features->5, …100)
- Implemented via Scikit-learn library and Gensim.

Topic Model Training



$$\frac{N_{td}+\alpha}{N_d+T\alpha}\times\frac{N_{wt}+\beta}{N_t+V\beta}$$

pyLDAvis


Selected Topic: 0
Previous Topic
Next Topic
Clear Topic
Slide to adjust relevance metric:(2)
λ = 1
0.0
0.2
0.4
0.6
0.8
1
Intertopic Distance Map (via multidimensional scaling)
PC2
PC1
1
2
3
4
5
Marginal topic distribution
2%
5%
10%
Top-30 Most Salient Terms 1
0
100,000
200,000
300,000
400,000
bed
staff
friendly
room
helpful
breakfast
shower
bathroom
walk
station
great
close
location
excellent
check
work
comfortable
city
good
comfy
food
stay
metro
book
small
service
restaurant
window
day
time
Overall term frequency
Estimated term frequency within the selected topic
1. saliency(term w) = frequency(w) * [sum_t p(t | w) * log(p(t | w)/p(t))] for topics t; see Chuang et. al (2012)
2. relevance(term w | topic t) = λ * p(w | t) + (1 - λ) * p(w | t)/p(w); see Sievert & Shirley (2014)