Task 6: Hygiene Prediction

The objective of this task was to predict whether a restaurant is hygienic or not based on all the yelp reviews given for the restaurant. A balanced labelled training data was provided where each restaurant's reviews were combined together and corresponding label was provided. Exactly 546 total labels were provided for training the classifier. There were remaining 12753 test instances which had to predicted using the trained classifier. F-measure was used for evaluation of the classifier. Specifically F1-score was used. The test instances were assumed to be unbalanced i.e. most of the restaurants in that set passed the inspection and very few did not pass. This is reason F1 measure was used instead of accuracy.

Different Techniques applied:-

A lot of different techniques were tried with MeTA and Python scikit-learn. The important attempts are given below. The entire summary is at the end of report that gives a good high level different things attempted to improve the F1-score.

MeTA C++ toolkit.
Text Representation:bag of words - unigram, bigrams, trigrams and ngram-pos/trees attempted**.**
Features: - Bag of words frequencies
Classifiers - SVM Stochastic Gradient, Naive Bayes
Initially, **MeTA** was selected for training the classifier. Various **bag of word** models were tried (unigram, bigram and trigram). The very first **SVM Stochastic Gradient** model with **logistic** loss function gave the F1 score of **0.542**. This classifier used unigram, bigram and trigram word frequencies generated as the features. 30-fold cross validation was used for optimizing the classifier.

To improve the score using MeTA even further, some other strategies were tried and some success was achieved using Naive Bayes classifier with unigrams and bigrams frequencies which gave F1-score of **0.544**. It seems that strong independence assumption in Naive Bayes gave little improvement to score.

For making a reasonable improvement, a different text representation technique was attempted such as "**ngram-pos**" and **trees** representation. But with basic features configuration, MeTA indexing took a lot of memory and time and did not ever complete the process. Trees representation seem like a very **CPU intensive** task which the personal mac could not handle.

With less documentation and guidance in MeTA for adding new features/classifiers, a different technology was selected for improving classifier even further.

Python sci-kit/gensim:

Text Representation:- Bag of Words - Unigrams, Bigrams and Trigrams
Features:- Word frequencies, Topic distribution, zip code, number of reviews, average rating, cuisines (various combinations of features attempted)
Features Optimization:- univariate - Pearson's correlation co-efficient, histograms on reference data with known unhygienic labels
Classifiers:- Random Forest, SVM, Stochastic Gradient Descent

Random Forest:

Random forest classifier was applied using different combination of features and different types of features. The training data was split into training and test data using ratio split of 0.3 The test data was used to evaluate and optimize the classifier. The number of estimators was kept to 100 unless explicitly mentioned.

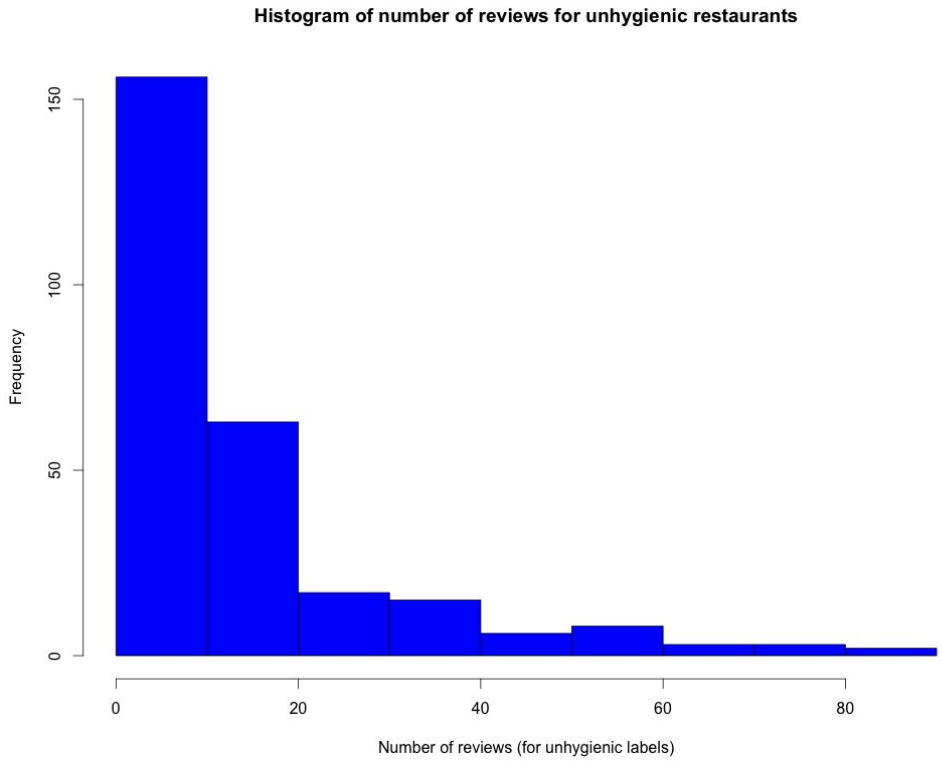
1) Word Frequencies using bag of words model:-
   Word frequencies of unigrams and bigrams were used as features. Different number of estimators were used in random forest classifier. The default (lower) number of estimators gave a lower F1 score. Once this was changed to 100 then the **F1 scored was improved to 0.558**

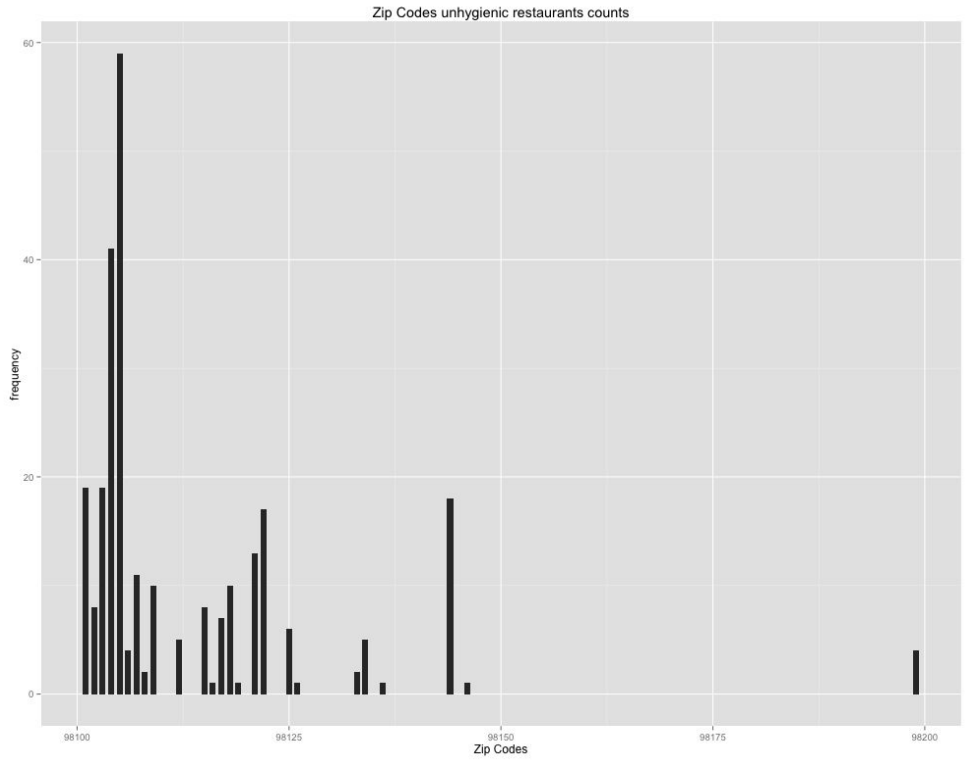
2) Document-Topic distribution (using LDA Topic Modelling) as features:-
   LDA Topic Modelling (using gensim library) was applied on bag of words model (unigrams and bigrams). The number of topics was selected as 50. The resulting document-topic distribution was used as features for training the classifier. The F1 score of 0.53 was achieved using just the topic modelling as features

3) Reference details as features:
   Additional reference information such as cuisine, zip code, average rating, number of reviews were considered as features as well. It was observed that average rating and certain cuisines had negative correlation with the labels. Univariate analysis was done using **Pearson's correlation coefficient** to compare the strengths of the reference features. It was observed that none of the reference feature had any strong correlation. But zip code and number of reviews did have some very weak positive correlation with the labels (~ 0.1). Certain cuisines also had some weak correlation with the labels. Using the pearson coefficient analysis on each of the cuisine associated was done. Cuisines which had some positive correlation with the labels were only kept as final features for training the classifier.

4) Reducing the reference values: [Struggles with adding reference values as features] It was discovered the lot of reference values were not helping the classifier. Some more filtering and correct mapping needed to be done for specific zip codes and cuisines. It was observed that certain zip codes had lot of unhygienic labels, same applied for cuisines. Keeping this in mind, only those high risk zip codes and cuisines were considered as reference features. Others were ignored before sending it to classifier. Please see the below plots for more information on how zip codes were transformed to different weights based on below plots.
5) Topic Modeling tweaking: Some tweaking to improve the F1 score was done in LDA topic modelling. The model was attempted to re-train using the part of the data first and then with entire data. It was observed that LDA topic model evaluation was difficult. The best F1 score achieved with topic models and reference values as features was **0.543**

Here are some histograms of reference features from the training data which had labels as unhygienic:





From the plots, it is clear that certain zip codes had lot of unhygienic restaurants. Also, it can be observed that if number of reviews is < 20 then probability of restaurant being unhygienic increases. This signals as weights were considered as features for zip codes and number of reviews. Similar analysis was attempted with average ratings, but there wasn't much correlation with hygiene as lot of unhygienic restaurants had 4-4.5 ratings.

The max F1 score achieved with topics distribution as features and reference fields such as zip code, number of reviews and selected cuisines was lower than the score achieved with the word frequencies model. The topic model features were definitely somehow more overfitting to the training data.

Best Performing Method:-

- What toolkit was used?

  Python scikit-learn was used for the best performing method

- How was text preprocessed? (i.e., stopword removal, stemming, or any data cleaning technique)

The preprocessing was done using TFIDF vectorizer as part of scikit-learn. The stopwords were removed. No explicit stemming related parameters were passed.

Below are the exact parameters that were used to preprocess the text and build features:

TfidfVectorizer(max_df=0.5, max_features=5000,min_df=2, stop_words='english',ngram_range=(1,2),use_idf=True)

Maximum features: 5000, minimum document frequency = 2, maximum document frequency = 0.5 and Inverse Document Frequency = True

- How was text represented as features?

Bag of words model was used to represent the text. Unigram and Bigram frequencies were used as features.

- What was the learning algorithm used?

The learning algorithm used was Random Forest with number of estimators kept as 100. The labelled data was split into train and test data using split ratio of 0.3

An F1 score of **0.558** was achieved for the above best performing method.

Below is the list of techniques tried. It gives summary of which toolkit was used, how was the text represented and which features was used to represent the text, classifier (learning algorithm) used, max F1 score achieved with the technique and short description.

Tools Used	**Text Representation**	**Features**	**Classifier**	**Max F1 score (actual from feedback)**	**Description**
MeTA	bag of words	1-gram,2-gram and 3-gram phrases frequencies	**One vs All Base:** stochastic gradient descent	0.542	This achieved the baseline required for task6.(used cross

			loss function: logistic		validation - 30 fold)
MeTA	bag of words	unigram and bigram phrase frequencies	Naive Bayes	0.544	It gave better F1 score. It seems like independence assumption for features improved the f1 score (used cross validation - 30 fold)
Python sci-kit	bag of words	unigram words frequencies	Random Forest	0.537	It gave less f1 score because only unigram words were considered as features
Python sci-kit	bag of words	unigram and bigrams words frequencies	Random Forest	0.533	It gave almost a similar score as unigrams.(because number of estimators was kept default=10)
Python sci-kit	**bag of words**	**unigram and bigrams words frequencies, idf=true, min_df=0.2,max_df=0.5,stop_words='english'**	**Random Forest**	**0.558**	**It gave better score than previous two once number of estimators was changed to 100**
MeTA	tree		Naive Bayes		Failed due to large space usage - More RAM needed for indexing

					with tree features
Python sklearn	bag of words	topic distribution as features	Random Forest		document-topic features did not exceed in performance than unigram/bigram frequencies features.
Python sci-kit	bag of words (unigrams, bigrams)	topic distribution as features with additional reference features: selected cuisines, number of reviews	Random Forest	0.53	document-topic features did not exceed in performance than unigram/bigram frequencies features.
Python sci-kit	bag of words	word frequencies of unigrams and bigrams with additional reference features	SGD logistic loss function and l1 regularization penalty, iterations=1000	0.51	SGD did not give the same performance as in MeTA
Python sklearn	bag of words	topics and selected cuisines, zip code, number of reviews and average rating	KNN with neighbors = 5 , 300. weights=uniform and distance respectively	0.53	document-topic features did not exceed in performance than unigram/bigram frequencies features.