**Machine Learning Final Project**

Authors: Lior Shachoach 206284960 and Moriya Barel 207304353

**About the dataset**:

Yelp dataset – the Yelp dataset is a collection of information contained in 5 separate files about businesses, including details about their products and services, customer reviews, and ratings. It contains a large amount of information about businesses in various industries, such as restaurants, shops, and entertainment venues. The dataset contains around 150,000 businesses, 700,000 user generated reviews, user’s data, check ins and more. We focused on the businesses and the reviews – each review contains a business id which allows gathering all the reviews of a specific business. This allowed us to learn about businesses through their respective reviews.

**Questions**:

1. Can a model be trained to predict if a given review is negative/neutral/positive.
2. Can a model be trained to predict the type of business given its reviews.

**Classifiers/Techniques**:

Classifiers:

* Support Vector Machine
* K-Nearest-Neighbor
* Random Forest
* Regression

Techniques:

* Stop Words
* TF-IDF
* Contractions (expand words such as “I’ve” to “I have”)
* High level categorization
* Text Aggregation
* Multi Label Classification
* Normalization

**Sentiment analysis of Yelp reviews**

Can a model be trained to predict if a given review is negative/neutral/positive?

**Corpus**

The dataset consists of 700,000 reviews, we decided to perform tests on the first 10,000 because handling 700,000 was hard to understand and hard to compute.

**Feature Extraction and Normalization**

We extracted two types of features: (i) star ratings and (ii) textual features consisting of unigrams, bigrams, and trigrams.

For star ratings we created a trinary feature representing rating 1-2 stars, 3 stars, and 4-5 stars as -1, 0, and 1 respectively.

For extracting textual features, we used Contractions to gain homogeneous text as we noticed that different users express themselves differently for example changing “I’ve” to “I have”. We did not remove the stop words as they play an important role in understanding user sentiments. The cleaned text is then tokenized to collect unigrams (individual words) and calculate their frequencies across the entire corpus using TF-IDF technique. This results in around 25,000 unique unigrams. To condense this feature set we set the minimum DF (document frequency) to 100 which results in 943 unigrams. We did the same for bigrams and trigrams.

**Results – 10,000 reviews 80% train 20% test**

SVM:

CountVectorizer, removing stop words, only unigrams - Accuracy: 0.799

CountVectorizer, only unigrams - Accuracy: 0.8125

TF-IDF, not setting minimum DF, only unigrams - Accuracy: 0.847

TF-IDF, minimum DF 100, only unigrams - Accuracy: 0.8425

TF-IDF, minimum DF 100, unigrams bigrams and trigrams - Accuracy: 0.8535

KNN k=7:

CountVectorizer, removing stop words, only unigrams - Accuracy: 0.73

CountVectorizer, only unigrams - Accuracy: 0.718

TF-IDF, not setting minimum DF, only unigrams - Accuracy: 0.7235

TF-IDF, minimum DF 100, only unigrams - Accuracy: 0.7085

TF-IDF, minimum DF 100, unigrams bigrams and trigrams - Accuracy: 0.7195

k=29 TF-IDF, minimum DF 100, unigrams bigrams and trigrams: Accuracy: 765

k=49 TF-IDF, minimum DF 100, unigrams bigrams and trigrams: Accuracy: 767

k=199 TF-IDF, minimum DF 100, unigrams bigrams and trigrams: Accuracy: 0.7605

k=499 TF-IDF, minimum DF 100, unigrams bigrams and trigrams: Accuracy: 0.73

Random Forest 100 estimators:

CountVectorizer, removing stop words, only unigrams - Accuracy: 0.7935

CountVectorizer, only unigrams - Accuracy: 0.766

TF-IDF, not setting minimum DF, only unigrams - Accuracy: 0.7565

TF-IDF, minimum DF 100, only unigrams - Accuracy: 0.7995

TF-IDF, minimum DF 100, unigrams bigrams and trigrams - Accuracy: 0.8015

LogisticRegression:

CountVectorizer, removing stop words, only unigrams - Accuracy: 0.8255

CountVectorizer, only unigrams - Accuracy: 0.834

TF-IDF, not setting minimum DF, only unigrams - Accuracy: 0.836

TF-IDF, minimum DF 100, only unigrams - Accuracy: 0.843

TF-IDF, minimum DF 100, unigrams bigrams and trigrams - Accuracy: 0.8425

**Analysis**

Overall, we can see that the best results are obtained with SVM and Logistic Regression, with the highest accuracy of ~ 0.85 when using TF-IDF with minimum document frequency (DF) set to 100 and using unigrams, bigrams, and trigrams. The results also show that TF-IDF alone does not improve a lot without setting a minimum document frequency to exclude irrelevant features and reduce the dimensionality, moreover, the addition of bigrams and trigrams as features does not seem to have a significant impact on the performance, but still improve a little. In general, stop words are not removed in sentiment analysis because they help understand the emotion in the text, truly in most cases we can see that removing stop words had a negative effect on the results.

We believe that KNN is worse than the other classifiers for the task at hand because of the high dimensionality of the data, in high dimensional space there may be a lot of irrelevant features which the KNN model does not know how to handle.

We believe that Random Forest is not the best tool here because our data is imbalanced and while the decision-making process is fine at handling some overfitting, it does not perform well on imbalanced datasets in contrary to its counterparts (SVM and Logistic Regression) which handle imbalanced datasets better.

**Classifying Yelp’s businesses to categories**

Can a model be trained to predict the type of business given its reviews?

We formulated the task of classifying a business into relevant categories as a learning problem. However, since a business is unexclusively associated with multiple categories at the same time, it is not a simple binary classification or a multi-class classification. It is rather a multi-label classification problem.

**Corpus**

The dataset consists of 150,000 businesses. For our study, we have considered only those businesses that are categorized as either “Restaurant” / “Shopping” / “Health & Medical”. This reduced the number of businesses to ~94,000. Furthermore, since we are classifying each business by its respective reviews, we chose to eliminate businesses with less than 60 reviews and those with more than 80 reviews. The lower bound is to make sure each business has enough data to work with and the upper bound is to reduce the number of businesses because it takes too long to compute. At last, we had 4489 businesses each with 60-80 reviews.

**Feature Extraction and Normalization**

Categorizing the businesses was one of the main difficulties in the task. Each business in the dataset is already categorized to many categories, for example, "Doctors, Traditional Chinese Medicine, Naturopathic/Holistic, Acupuncture, Health & Medical, Nutritionists". Since there are so many categories for each business, we had to perform high level categorization to reduce the number of classes. At first, we tried to use MultiLabelBinarizer to predict as many categories as possible, but that resulted in near 0 accuracy since the number of possible labels for a business was exponential in the number of categories. After failing to categorize in this manner, we tried an easier approach: predict whether a given business is a restaurant or not (since many businesses in the dataset are restaurants). This approach worked but wasn’t sophisticated enough. We decided to divide the businesses into 3 main categories - “Restaurant” / “Shopping” / “Health & Medical”.

Example:

| Restaurant | Shopping | Health & Medical

Business #1 | 1 | 0 | 0

Business #2 | 1 | 1 | 0

Business #3 | 0 | 1 | 1

... ... ... …

The other main difficulty in the task was: how could the same model be trained on reviews and tested on businesses? When training the model, the input was reviews and their respective labels, later on, to answer the question, the model had to classify a set of reviews of a certain business and then label the business based on a majority vote of the classification of the reviews. The problem was that the features of each set of business reviews (test set) were different than those of the train set, we had to transform each set of business reviews features according to the training features. Before understanding the problem and how to solve it, we tried to avoid it altogether by performing text aggregation and creating one big document of reviews for each business to match the number of features, both methods provided great results.

**Classification**

As mentioned above, we had 2 primary approaches.

1. Text Aggregation – the idea is to append/aggregate all the reviews of each business into a single document which represents each business. This way training and testing the model didn’t require any special adjustments and was straight forward.
2. Majority Vote – slightly more advanced technique, for the training set, we had a list of reviews for the model to be trained on, afterwards, for the test set, we had a list of lists, each inner list contained all the reviews of a single business, we had to run the model on each review of a specific business and then take the majority vote of the prediction of each review.

We had around 260,000 total reviews in the training sample alone, without aggregating the reviews for each business, given the high dimensionality and large number of vectors (reviews), training any classifier other than regression took too long to compute, SVM for example could not calculate the optimal hyperplane in a reasonable time. Therefore, we will show results for aggregation technique with 4 different classifiers as well as majority vote with regression only.

**Results – 4489 businesses, 80% train 20% test, min-df 300, removing stop-words, unigrams bigrams & trigrams.**

Text Aggregation:

SVM – Accuracy: 0.9710467706013363

KNN (k=5) – Accuracy: 0.9543429844097996

Random Forest – Accuracy: 0.9599109131403119

Regression - Accuracy: 0.967706013363029

Majority Vote:

Regression - Accuracy: 0.967670011148272

**Analysis**

Overall, both approaches provided amazing results while the best classifiers remain SVM and regression as they handle large dimensions exceptionally well in contrast to KNN and Random Forest. Unlike part 1 (sentiment analysis), the removal of stop-words here increased the accuracy as they do not actually contribute to the classification process. Like part 1, we kept the min-df boundary to drop irrelevant features as well as kept the bigrams and trigrams detect and distinguish between different types of reviews.

We feared the results may be exceptionally high due to the dataset being imbalanced, therefore we experimented with different categories as well as different number of categories.

For example – 6 categories (Health & Medical, Arts & Entertainment, Hotels & Travel, Automotive, Local Services, Home Services) – granted close to 80% accuracy rate.

Fixes:

Tested more K’s for KNN on sentiment analysis – results added above (with the rest of the results), we can see that while a slightly larger K actually did improve, the general statement remains the same. Observation – 29 < K < 199 seems to give the best results, for K = 499 there seems to be overfitting and therefore lower results.

Why is business classification much more accurate then sentiment analysis:

We feel like sentiment analysis is harder to learn and classify mostly because reviews that are related to different types of business share potentially less similarities, for example a review about a restaurant is very likely to include words like ‘food’ or ‘eat’ while reviews about a clothing shop definitely shouldn’t, in contrast to sentiment analysis which does not have that kind of distinct or unique identifiers.

Separate multilabel to single label:

Began by extracting businesses that had a unique category out of the 3, restaurant but not shopping and health and medical, shopping but not restaurant and health and medical et cetera, tested results on 450 businesses in total, 150 of each category.

Results: min\_df=100, stop\_words='english', ngram\_range=(1, 3)

SVM - Accuracy: 0.9444444444444444

KNN - k = 19: 0.9444444444444444, k = 49: Accuracy: 0.9

Random Forest - Accuracy: 0.9333333333333333

Regression - Accuracy: 0.9666666666666667

We believe the results are pretty much the same (and not closer to 100%) because we tested on a perfectly balanced sample in contrary to our previous sample which was quite imbalanced.