Predicting Star Ratings from User Reviews
CSE 142 Fall 2019

Group #2: Angelina Agabin (aagabin - 1590090)
Celine Seghbossian (cseghbos - 1595557)
Kaleen Shrestha (kashrest - 1615749)

December 6, 2019

1 Tools Used (can be reduced if we go over the 3 page limit)

We extensively made use of packages in NLTK for data pre-processing, Scikit-learn for feature extraction, and ML algorithms, specifically:

sklearn.feature_extraction.text import TfidfVectorizer: This tool made use of the bag of word model by taking documents and extracting n-gram features. The hyper parameters were tuned to make use of the term-frequency - inverse document frequency (TF-IDF) to pick weights for features, favoring features/words that occur frequently within a single document but not frequently across all documents.

nltk.stem import PorterStemmer: This tool was used to pre-process the text data by stemming words. An example would be that texts such as run, running, runs, runned would all be transformed to run.

sklearn.preprocessing import StandardScaler: This was used to normalize feature vectors so that no feature (like the word "the") would be unfairly favored due to scale imbalance.

sklearn.model_selection import train_test_split: This class was used to do cross-validation to gather a better accuracy/evaluation of current model to help with tuning the hyper-parameters of the models.

sklearn.linear_model import Perceptron, LogisticRegression, svm.LinearSVC, from sklearn.neighbors import NearestCentroid: These were imported models, we used three linear models and one variante of knn from sklearn, and experimented tuning the hyperparameters to get good preforming instances of the models.

2 Diversity

Our group acknowledges and appreciates the diversity that each member contributes to the project. Our group differs from most other groups because we are three women in STEM. Having an all-female group gave us a different method of communication and collaboration that is not often experienced in a traditionally male-dominated field. Within our group, each member's ethnic background and coding experience provided useful insight regarding the project. As children of immigrants from Nepal, China, the Philippines and Iran, our varying experiences and mindsets allowed us to approach the project from multiple directions. Additionally, two out of three of our members were raised by parents who code which undoubtedly affects their coding practices. Lastly, we found it useful that each one of us has experience with different coding languages and development tools. Our diversity, in combination with our willingness to learn from each other contributed to our success as a group.

3 Abstract

In this project, preprocessing, feature extraction, and label prediction were implemented using various sklearn libraries. Text from the reviews were stemmed and simplified using a bag-of-words model from which features for training were then extracted. Data was then normalized to account for the uneven distribution of data among the five labels. Predictions were then made using Logistic Regression, SVM, Perceptron, and Nearest-Centroid. Models were trained and tested using cross validation or splitting data into train and dev sets. Tuning was done to various models to increase accuracy while avoiding over or underfitting. The final prediction is made with ensemble voting.

4 Data Pre-processing

In order to reduce redundancy in the data, we made use of the stemming method, particularly the Porter Stemmer class from NLTK. Words then like running, runs, and run would be all transformed into the word run. This way, we would not get different features during feature extraction for the same English meaning. For this problem, running, run, and runs all signify the meaning of run and that is good enough for our purposes.

5 Feature Extraction

Feature extraction was done using Sklearn's TfidfVectorizer class, on the text. This is a bag-of-words approach where words are features. The hyper-parameters for TfidfVectorizer could be tuned to include a range of n-grams (groups of n words) and there are thresholds for the minimum and maximum frequency of a potential feature across documents/text reviews.

6 Approaches

For Logistic Regression, we tried a default hyperparameter setting for the model, used 3-fold cross evaluation to evaluate preformance, tweaked the hyperparameter to include some class balancing, and elastic net regularization (a combination of L2 and L1 regularization) and evaluated performance with 3-fold cross evaluation again, which yielded better accuracy. This was done with with the other models as well.

For example, we found that on test data like this, perceptron did not perform as well as other models. We had to add a regularizer to account for some classes having more instances than others, and had the algorithm balance the class weights. We found that increasing the number of iterations improved the accuracy, but also increased the run time of the program. Ultimately, perceptron still performed the worst out of all the algorithms.

Tuning SVM:
We found that changing most of the parameters didn't make a big difference, but we still had the algorithm add weights to balance the classes. While testing, we also found that SVM did not converge if the number of max iterations wasn't high enough. For the training dataset, a max iteration of 2000 allowed the algorithm to converge.

7 Results

We tested 4 models. And these are the accuracy measurements when the models were trained on 80% of the training data and tested on the remaining 20%:
Logistic Regression: 64.21%
SVM: 58.93%
Perceptron: 52.05%
Nearest-Centroid: 57.57%
And a voting ensemble method combining all four models: 64.12%

We ended up using voting for our final model, which was a mistake on our part. We did not have enough time to check accuracy. A better choice would have been to use the logistic regression model.

8 Conclusion

During this project, we experimented with feature extraction and model parameters to fine tune our models. We learned that feature extraction is a very experimental step, especially for such large dimension feature vectors in bag of words approaches. Cross-validation proved to be a valuable tool for judging the performance of our models, but we also discovered it increased running time even on lower folds like 3-fold. While tuning our models, we learned the parameters specific to each model and how tweaking them could possibly affect the accuracy or even the run time of the models.

9 Ideas for Future Work

In the future, we could try more ensemble methods like Ada-boost or XG-boost for the weak-performing models. Also if we could spend more time, we would have tried out lemma-ization as another data-preprocessing approach and experiment with adding the other data that came with the reviews like the number of "cool", "funny", "useful" votes a review got, as those could be helpful features as well.