Sentiment Analysis Term Project Report

Edward (Qianfu) Tang, April 2022

Introduction

This report focuses on a project in which we implement a sentiment analyzer that automatically classifies input text as either positive or negative. The dataset is extracted from the movie review dataset from Cornell. In the polarity dataset v2.0, there are 1000 positive movie reviews and 1000 negative ones.

Problem Definition

We aim to build a sentiment analyzer, that is, build a system that can classify a given movie review as either positive or negative. There are only 2 possible labels, positive and negative, denoted by 0 and 1 during our training and testing. The challenge is to extract important text features so that the models will learn effectively from the preprocessed data, and achieve an ideal accuracy. We can define "ideal" accuracy to be over 85%.

Previous Work

There are plenty of previous works that focus on classification of sentiments. However, in this paper, we are more focused on how different ways for feature extraction can impact the learning outcome for a set of selected models. Asghar et al. have summarized commonly-used preprocessing and feature extraction techniques. For preprocessing, the approaches include:

- Part-of-speech (POS) tagging: assigning a task to each word in a text and classifying a word to a specific morphological category, such as noun, verb, adjective, etc.
- Stop Word Removal: removal of stop words, which are defined by common-frequency words, such as "a", "the", "of", "an" in English. There is stop word vocabulary in English available in the NLTK library.
- Stemming and Lemmatization: converts inflected words into their stem forms (e.g. loving->lov) or lemma (loving->love).

- Negation: when a word that indicates negation like “not”, “shouldn’t”, “no”, it is important to also note how many of the surrounding words are affected by it.

Approach

Machine Learning Models

1. K-Nearest-Neighbor [5]

The k-nearest-neighbor algorithm is based on the Euclidean distance between data points, k nearest neighbors are taken, then the category of each neighbor is noted. Then count the number of the data points in each category. Assign the new data points to that category for which the number of the neighbor is maximum.

2. Decision Tree

A decision tree is a flowchart-like structure in which each internal node represents a "test" on an attribute (e.g. whether a coin flip comes up heads or tails), each branch represents the outcome of the test, and each leaf node represents a class label (decision taken after computing all attributes). The paths from root to leaf represent classification rules.

3. Random Forest

In a random forest classifier, a decision tree is used as a preliminary model, and each tree in the ensemble is built from a sample drawn with replacement (i.e., a bootstrap sample) from the training set. A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.

4. MLP Neural Net

The MLP consists of three or more layers (an input and an output layer with one or more *hidden layers*) of nonlinearly-activating nodes. Since MLPs are fully

connected, each node in one layer connects with a certain weight to every node in the following layer. With error function given by

$$\mathcal{E}(n) = \frac{1}{2}\sum_j e_j^2(n).$$

and gradient descent

$$\Delta w_{ji}(n) = -\eta \frac{\partial \mathcal{E}(n)}{\partial v_j(n)} y_i(n)$$

(η is the learning rate and y_i is the output of the previous neuron)

is applied to ensure that the weights quickly converge to a response, without oscillations.

5. Linear SVM

Linear support vector machine (SVM) creates a line or a hyperplane that separates the data into classes. The linear SVM algorithm finds the hyperplane that has the maximum margin that separates the data into 2 classes. The hyperplane can be formalized as:

$$\mathbf{w}^T\mathbf{x} - b = 0,$$

6. RBF SVM

The support vector machine with radial basis function kernel, which is defined on two samples, x and x', as:

$$K(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\sigma^2}\right)$$

7. Gaussian Naïve Bayes

Naïve Bayes is a conditional probability model that given a feature vector $(x_1, x_2, ..., x_n)$ with length n, for each of the K possible outcomes, or C_k, a conditional probability is assigned, defined as:

$$p(C_k \mid x_1, \ldots, x_n)$$

By Bayes's Theorem, this probability can be decomposed into:

$$p(C_k \mid \mathbf{x}) = \frac{p(C_k)\, p(\mathbf{x} \mid C_k)}{p(\mathbf{x})}$$

Under independent assumptions, this probability over the class variable C can be represented as:

$$p(C_k \mid x_1, \ldots, x_n) = \frac{1}{Z} p(C_k) \prod_{i=1}^{n} p(x_i \mid C_k)$$

where

$$Z = p(\mathbf{x}) = \sum_k p(C_k)\, p(\mathbf{x} \mid C_k)$$

8. Quadratic Discriminant Analysis

Different than the Bayes probability model, QDA models the distribution of X given the predictor's class (label) P(X=x|Y=k). Additionally, for each of the class y the covariance matrix is given by:

$$\Sigma_y = \frac{1}{N_y - 1} \sum_{y_i = y} (x_i - \mu_y)(x_i - \mu_y)^\top$$

The quadratic discriminant function is given by:

$$\delta_k(x) = \log \pi_k - \frac{1}{2} \mu_k^T \Sigma_k^{-1} \mu_k + x^T \Sigma_k^{-1} \mu_k - \frac{1}{2} x^T \Sigma_k^{-1} x - \frac{1}{2} \log |\Sigma_k|$$

Baseline Model

The dataset is split into 750:250 for both positive and negative reviews, for training and validation, respectively. During preprocessing, stop words are removed and vocabulary is extracted from training data. As a baseline system that we use to get our initial accuracy, I have implemented a bag-of-words(BoW) model, where we extract vocabulary from the training set.

The selected machine learning models include k-nearest-neighbors, decision tree, random forest, multilayer perceptron (neural network), linear support vector machine(SVM), Radial Basis Function(RBF) SVM, Naive Bayes and quadratic discriminant analysis (QDA).

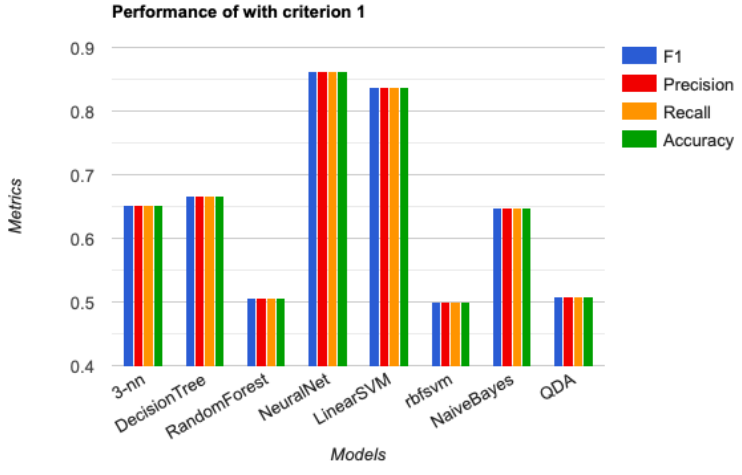
Improved System: Feature Extraction

There are other ways to do feature extraction other than, or on top of, the BoW model. These are:

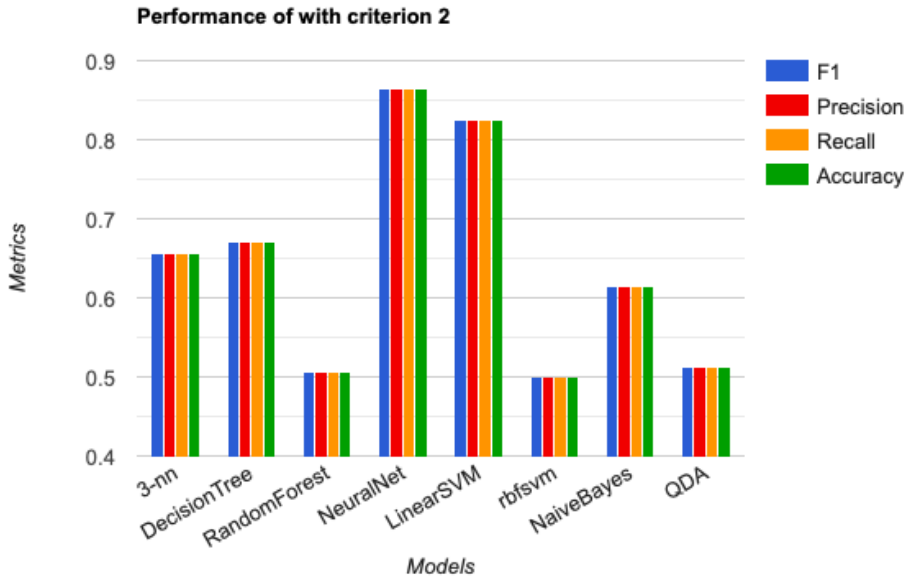
- Stop word removal: remove stop words using NLTK features
- Stemming: NLTK has implemented Porter Stemmer[4] in Python. Adding stemmer to the preprocessing allows us to reduce the size of vocabulary.
- Negation: include negation during preprocessing, using NLTK library
- POS tagging: include only adjectives and adverbs in the data in order to extract sentiment from text.

Results

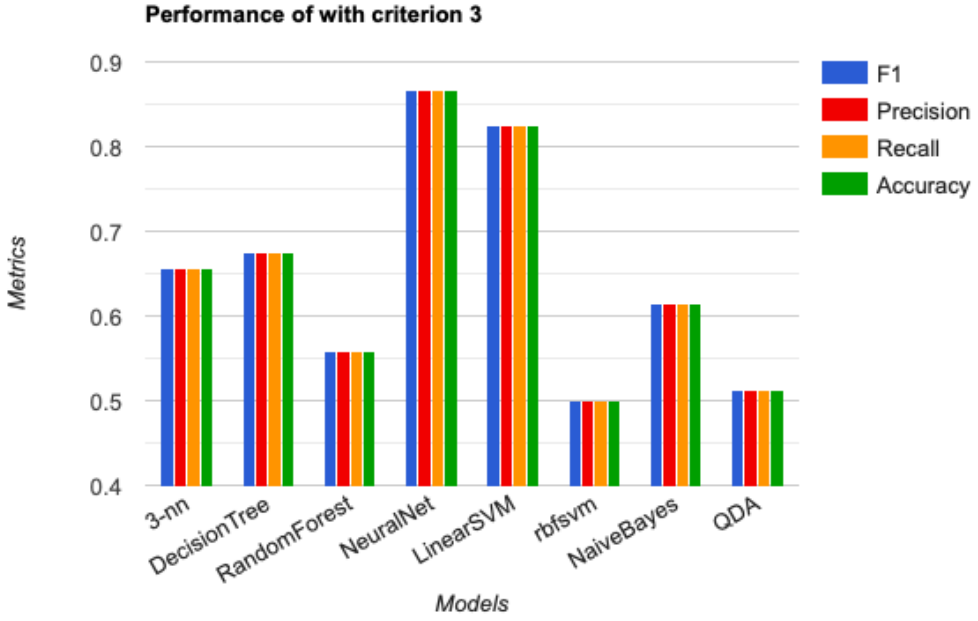
Criterion 1: Baseline system(BoW) with removing stop words:



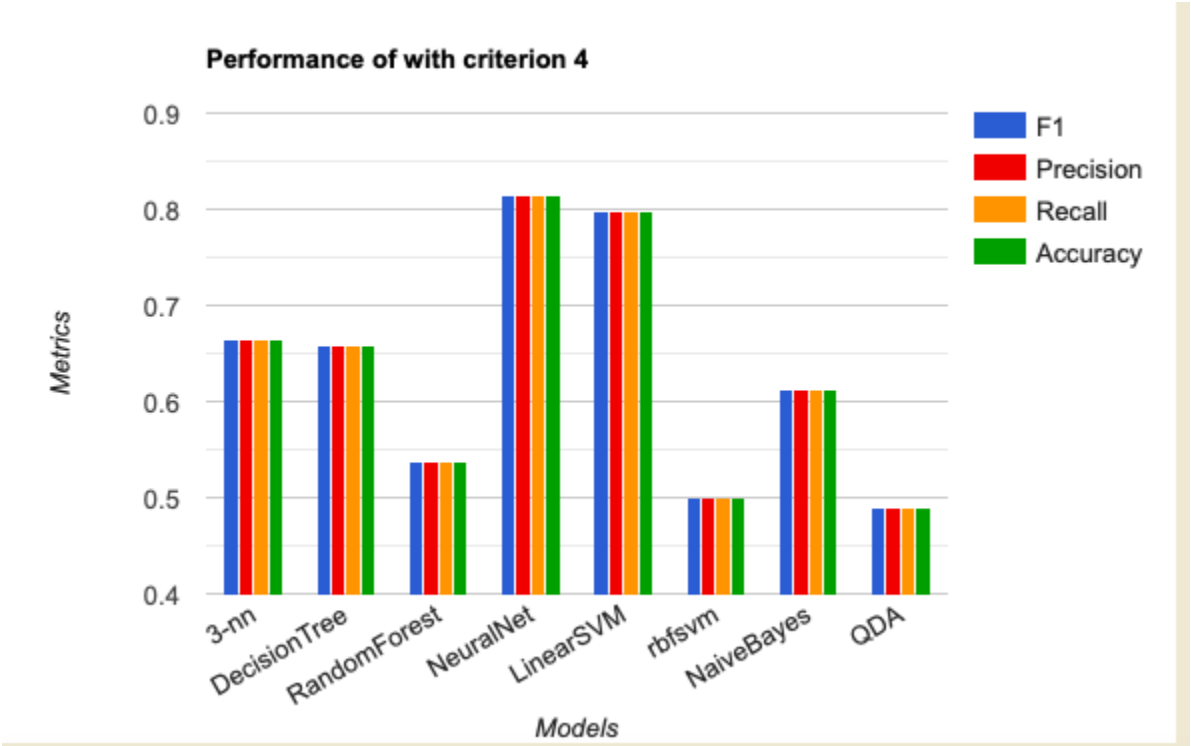
Criterion 2: BoW with removing stop words and Porter Stemmer:



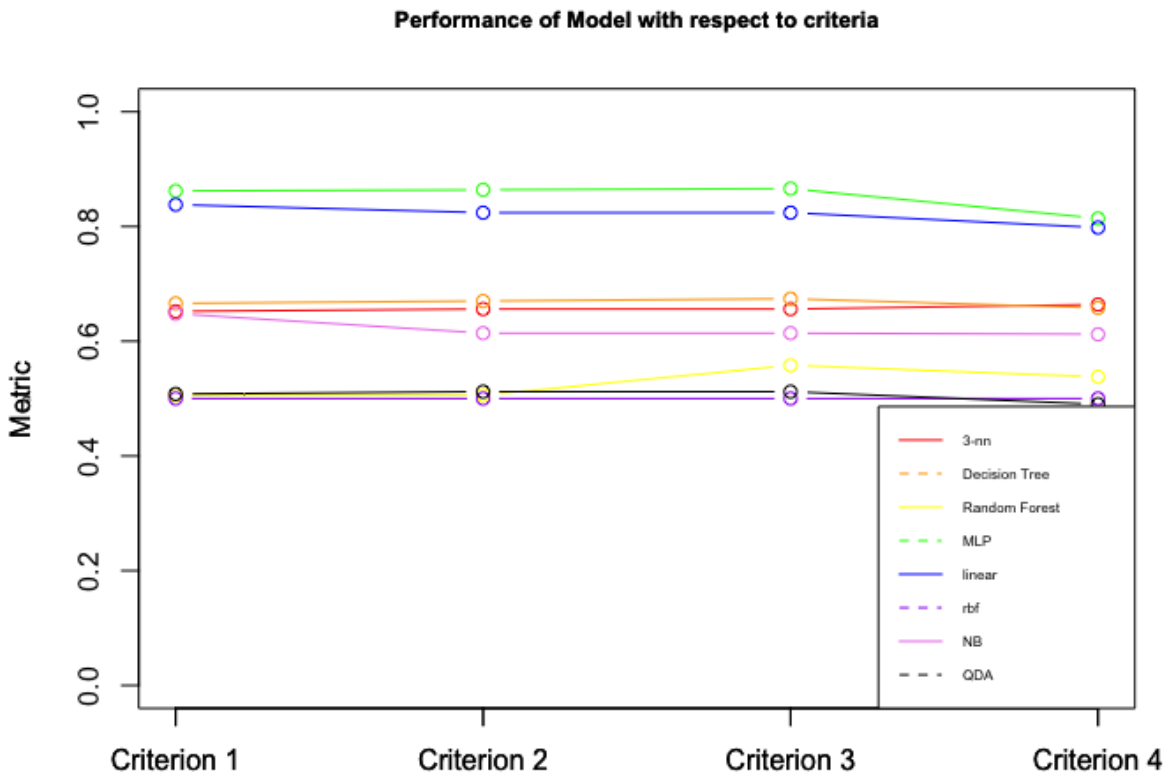
Criterion 3: BoW+stop word removal+stemming+negation



Criterion 4: BoW+stop word removal+POS tagging



Overall performance:



Discussion

Amongst all 8 models that we have experimented with, the MLP neural network has the best accuracy overall, in which a 86% accuracy is achieved even with only a baseline model that only contains feature extractions like BoW and stop words. I compared the POS tagging approach with the stemming and negation approach, and it seems like in this case only keeping the adjectives and adverbs achieves suboptimal results in comparison to baseline models. The f1, precision, recall, and accuracy are all the same, which could happen due to our low-level complexity of tests and models. RBF SVM has the worst performance, that it stays at 50% across all the trials.

Conclusion

Neural network performs better than other decision-based and linearly separated models due to the multi-layer feature that learns patterns with several encoding and decoding. Only keeping the adjectives and adverbs does not necessarily contribute to the accuracy of data, but rather, stemming and negation could improve the efficiency of feature extraction.

Works Cited

"Movie Review Data." Data, www.cs.cornell.edu/people/pabo/movie-review-data/.

Porter Stemming Algorithm, tartarus.org/martin/PorterStemmer/.

"1.11. Ensemble Methods." Scikit, scikit-learn.org/stable/modules/ensemble.html#forest.

"Decision Tree." Wikipedia, Wikimedia Foundation, 15 Apr. 2022, en.wikipedia.org/wiki/Decision_tree.

Peterson, Leif E. "K-Nearest Neighbor." Scholarpedia, www.scholarpedia.org/article/K-nearest_neighbor.

"Sklearn.ensemble.RandomForestClassifier." *Scikit*, scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html.

"Multilayer Perceptron." *Wikipedia*, Wikimedia Foundation, 12 Apr. 2022, https://en.wikipedia.org/wiki/Multilayer_perceptron.

Pupale, Rushikesh. "Support Vector Machines(Svm) - an Overview." *Medium*, Towards

Data Science, 11 Feb. 2019,

https://towardsdatascience.com/https-medium-com-pupalerushikesh-svm-f4b42800e989.

"Naive Bayes Classifier." *Wikipedia*, 4 Mar. 2022, en.wikipedia.org/wiki/Naive_Bayes_classifier#Gaussian_naive_Bayes. Accessed 29 Apr. 2022.

"Quadratic Discriminant Analysis." *GeeksforGeeks*, 18 July 2021, www.geeksforgeeks.org/quadratic-discriminant-analysis/. Accessed 29 Apr. 2022.

"Radial Basis Function Kernel." *Wikipedia*, 23 Nov. 2019,

en.wikipedia.org/wiki/Radial_basis_function_kernel.