NATURAL LANGUAGE PROCESSING - RESTAURANT REVIEW CLASSIFICATION

A PROJECT REPORT
SUBMITTED
IN THE PARTIAL FULFILLMENT OF
“MACHINE LEARNING WITH PYTHON INTERNSHIP”

By

Abhishek Lalwani
Akshitha Theretupally
Roopesh Kumar
Sunil Kumar

Under the esteemed guidance of
Gurvansh Singh
M.Tech
Knowledge Solutions India

KNOWLEDGE SOLUTIONS INDIA
2nd Floor Flat No, Ghanshyam park society, 5, Dhole Patil Rd, Pune, Maharashtra 411001

ABSTRACT

Sentiment analysis is a machine learning technique that detects polarity (e.g. a *positive* or *negative* opinion) within text, whether a whole document, paragraph, sentence, or clause. Understanding people's emotions is essential for businesses since customers are able to express their thoughts and feelings more openly than ever before. By automatically analyzing customer feedback, from survey responses to social media conversations, brands are able to listen attentively to their customers, and tailor products and services to meet their needs. For example, using sentiment analysis to automatically analyze 4,000+ reviews about your product could help you discover if customers are happy about your pricing plans and customer service.

In our project we used three machine learning models (SVM, KNN, and SVM-PCA) to perform sentiment analysis on the restaurant review dataset. The SVM and SVM-PCA model has the highest accuracy of 94% in classifying the reviews as positive and negative whereas, the KNN model gave us an accuracy of 90%. We used a bag-of-words model for representing text data for modeling text with machine learning algorithms. The bag-of-words model is simple to understand and implement and has seen great success in problems such as language modeling and document classification.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1: INTRODUCTION

Problem Statement

Our problem statement is classifying the restaurant review dataset into positive and negative reviews.

Objective

The objective of our project is to achieve accuracy of 90% or above.

1.1 NATURAL LANGUAGE PROCESSING (NLP)

Natural Language Processing, usually shortened as NLP, is a branch of artificial intelligence that deals with the interaction between computers and humans using the natural language. The ultimate objective of NLP is to read, decipher, understand, and make sense of the human languages in a manner that is valuable. Most NLP techniques rely on machine learning to derive meaning from human languages.

Natural Language Processing is the driving force behind the following common applications:

1. Language translation applications such as Google Translate
2. Word Processors such as Microsoft Word and Grammarly that employ NLP to check grammatical accuracy of texts.
3. Interactive Voice Response (IVR) applications used in call centers to respond to certain users' requests.
4. Personal assistant applications such as OK Google, Siri, Cortana, and Alexa.

1.2 Working of NLP

NLP entails applying algorithms to identify and extract the natural language rules such that the unstructured language data is converted into a form that computers can understand. When the text has been provided, the computer will utilize algorithms to extract meaning associated with every sentence and collect the essential data from them.

1.2.1 Sentiment Analysis

The most widely used technique in NLP is sentiment analysis. Sentiment analysis is most useful in cases such as customer surveys, reviews and social media comments where people express their opinions and feedback. The simplest output of sentiment analysis is a 3-point scale: positive/negative/neutral. In more complex cases the output can be a numeric score that can be bucketed into as many categories as required.

Sentiment Analysis can be done using supervised as well as unsupervised techniques. It requires a training corpus with sentiment labels, upon which a model is, trained which is then used to identify the sentiment.

Chapter 2: SOFTWARE LIBRARIES

A software library generally consists of pre-written code, classes, procedures, scripts, configuration data and more. Typically, a developer might manually add a software library to a program to achieve more functionality or to automate a process without writing code for it.

2.1 Libraries Used in Our Project

2.1.1 NumPy

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. The name is an acronym for "Numeric Python" or "Numerical Python". It is an extension module for Python, mostly written in C. This makes sure that the precompiled mathematical and numerical functions and functionalities of Numpy guarantee great execution speed.

Fig 2.1: NumPy library

NumPy enriches the programming language Python with powerful data structures, implementing multi-dimensional arrays and matrices. These data structures guarantee efficient calculations with matrices and arrays. The implementation is even aiming at huge matrices and arrays, better known under the heading of "big data". Besides that, the module supplies a large library of high-level mathematical functions to operate on these matrices and arrays.

2.1.2 Pandas

In computer programming, Pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. The name is derived from the term "panel data", an econometrics term for data sets that include observations over multiple time periods for the same individuals.

Fig 2.2: Pandas library

Some features of the Pandas library:

1. Data Frame object for data manipulation with integrated indexing.
2. Tools for reading and writing data between in-memory data structures and different file formats.
3. Data alignment and integrated handling of missing data.
4. Label-based slicing, fancy indexing, and sub setting of large data sets.
5. Data structure column insertion and deletion.

2.1.3. Re (Regular expression)

A regular expression is a special sequence of characters that helps you match or find other strings or sets of strings, using a specialized syntax held in a pattern. Regular expressions are widely used in the UNIX world.

The Python module provides full support for Perl-like regular expressions in Python. The re module raises the exception re.error if an error occurs while compiling or using a regular expression.

2.1.4 NLTK

The Natural Language Toolkit, or more commonly NLTK, is a suite of libraries and programs for symbolic and statistical natural language processing (NLP) for

English written in the Python programming language. It was developed by Steven Bird and Edward Loper in the Department of Computer and Information Science at the University of Pennsylvania. NLTK includes graphical demonstrations and sample data.

Features of NLTK

- Lexical analysis: Word and text tokenizer
- n-gram and collocations
- Part-of-speech tagger
- Tree model and Text chunker for capturing
- Named-entity recognition

2.1.5 SKLEARN

Scikit-learn (formerly scikit.learn and also known as sklearn) are a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, *k*-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

Chapter 3: MATHEMATICS

3.1 SVM

The main objective of SVM is to find the optimal linearly separating hyperplane which maximizes the margin.

Hyper-plane

It is plane that linearly divide the n-dimensional data points in two components. In case of 2D, hyperplane is line, in 3D it is plane. It is also called as *n-dimensional line*. *Fig.3* shows, a blue line(hyperplane) linearly separates the data point in two components.

Fig.3

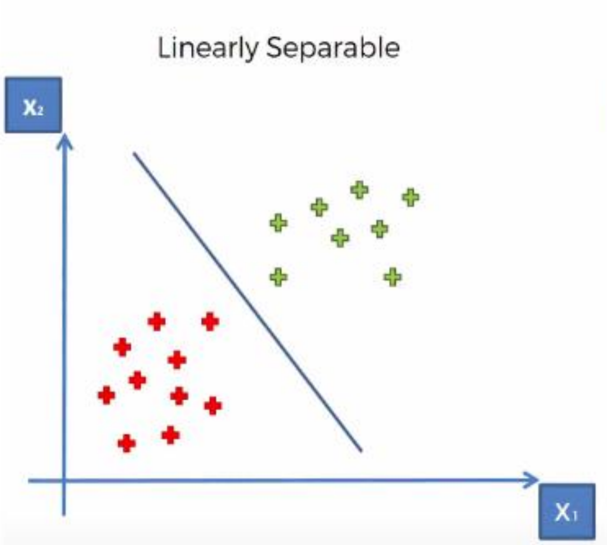
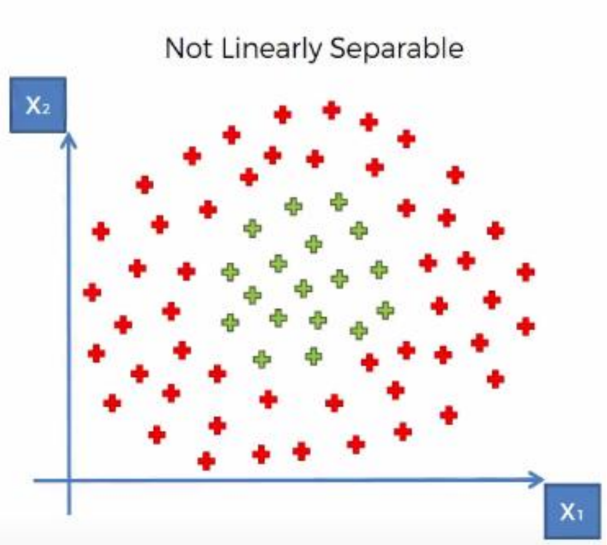


Fig.4



In the *Fig.3*, *hyperplane* is line divides data point into two classes (red & green), written as

$$\boxed{y = a * x + b}$$

$$\boxed{a * x + b - y = 0}$$

Let vector X=(x,y) and W=(a,-1) then in vector form hyperplane is

$$\boxed{W . X + b = 0}$$

3.2 KNN

In the classification problem, the K-nearest neighbor algorithm essentially said that for a given value of K algorithm will find the K nearest neighbor of unseen data point and then it will assign the class to unseen data point by having the class which has the highest number of data points out of all classes of K neighbors.

For distance metrics, we will use the Euclidean metric.

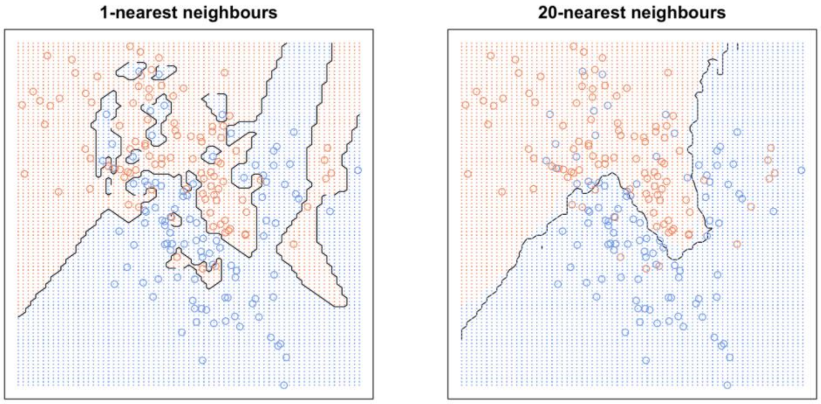
$$d(x, x') = \sqrt{\left(x_1 - x_1'\right)^2 + \ldots + \left(x_n - x_n'\right)^2}$$

Finally, the input x gets assigned to the class with the largest probability.

$$P(y = j | X = x) = \frac{1}{K} \sum_{i \in \mathcal{A}} I(y^{(i)} = j)$$

Now most probably, you are wondering how to decide the value for variable K and how it will affect your classifier. Well, like most machine learning algorithms, the K in KNN is a hyper parameter that we, must decide in place to get the most suitable fit for the data set.

When K is small, we are holding the region of a given prediction and pushing our classifier to be “more blind” to the overall distribution. A small value for K provides the most adjustable fit, which will have low bias but high variance. Graphically, our decision boundary will be more irregular. On the other hand, a higher K averages more voters in each prediction and hence is more flexible to outliers. Larger values of K will have a smoother decision boundary which means lower variance but increased bias.



1-nearest neighbours

20-nearest neighbours

Chapter 4: MACHINE LEARNING MODELS

In our project we used 3 models to classify the restaurant reviews dataset. They are as follows.

1. SVM
2. K-NN
3. SVM with PCA

General Steps followed for NLP models:

Step 1: Importing the libraries and dataset.

Step 2: Text Cleaning or Preprocessing

1. Remove Punctuations, Numbers: Punctuations, Numbers doesn’t help much in processing the given text, if included; they will just increase the size of the bag of words that we will create at the last step and decrease the efficiency of the algorithm.
2. Stemming: Take roots of the word
3. Convert each word into its lower case: For example, it is useless to have the same words in different cases (e.g. ‘good’ and ‘GOOD’).
4. Removing some words from ‘stop words’ increases the accuracy. For example “don't”, “not”, “but”, “won't”, “he”, “she” etc.
5. Appending the words to stop words which do not have the impact on the review also helps in increasing the accuracy.

Step 3: Making the bag of words

1. Take all the different words of reviews in the dataset without repeating the words.
2. One column for each word, therefore there are going to be many columns.

3. Rows are reviews
4. If word is there in a row of a dataset of reviews, then the count of word will be there in a row of bag of words under the column of the word.

Step 4:

We split the corpus into training and test sets. For this, we need class **train_test_split** from **sklearn.model_selection**. Test size chosen is 0.05 for all the models.

X is the bag of words; y is 0 or 1 (positive or negative).

Step 5: Fitting a Predictive Model

- Object creation
- Fit the model via .fit() method with attributes X_train and y_train Predicting Final Results via using .predict() method

Step 6: EVALUATION

Precision: Precision is calculated as the number of correct positive predictions (TP) divided by the total number of positive predictions (True Positive + False Positive). It is also called positive predictive value (PPV). The best precision is 1.0, whereas the worst is 0.0

Recall: Recall is calculated as the number of correct positive predictions divided by the total number of positives. It is also called recall (REC) or true positive rate (TPR). The best sensitivity is 1.0, whereas the worst is 0.0.

F1 score: F-score is a harmonic-mean of precision and recall.

$$F_\beta = \frac{(1+\beta^2)(\mathrm{PREC}\cdot\mathrm{REC})}{(\beta^2\cdot\mathrm{PREC}+\mathrm{REC})}$$

β is commonly 0.5, 1, or 2.

Accuracy: Accuracy is calculated as the total number of two correct predictions (TP + TN) divided by the total number of a dataset (P + N). The best accuracy is 1.0, whereas the worst is 0.0. It can also be calculated by 1 – ERR.

3.1.1 SVM

A support vector machine (SVM) is a supervised machine learning model that uses classification algorithms for two-group classification problems. After giving SVM model sets of labeled training data for each category, they’re able to categorize new text. So we’re working on a text classification problem. **Support Vector Machines (SVM)**: a fast and dependable classification algorithm that performs very well with a limited amount of data to analyze. The idea behind the SVM algorithm is simple, and applying it to natural language classification doesn’t require most of the complicated stuff.

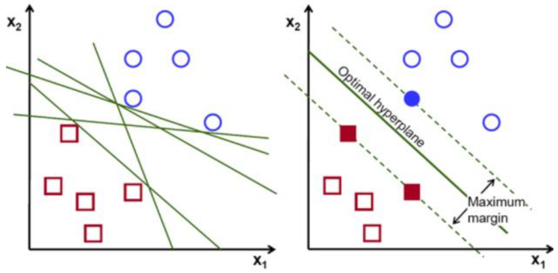


Fig 3.1: Figure illustrating SVM

Confusion Matrix for SVM model:

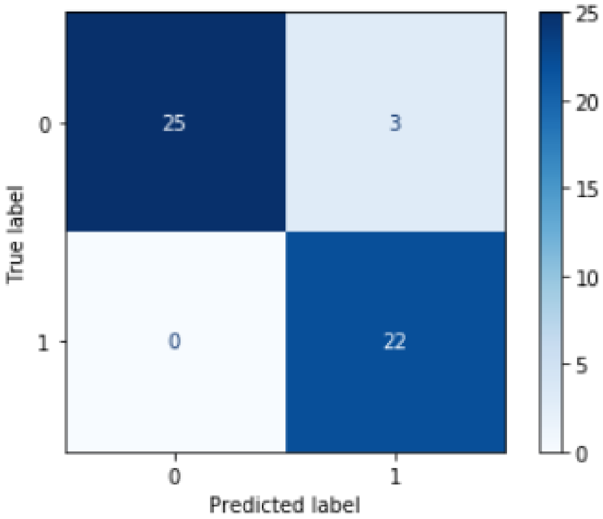


Fig 3.2: Confusion matrix of SVM

Classification report of SVM:

	Precision	Recall	F1-score
0	1.00	0.89	0.94
1	0.88	1.0	0.94

The accuracy of our SVM model is 94% and our test size is 0.05

Table 3.1: Classification report of SVM

3.1.2 KNN

In pattern recognition, the *k*-nearest neighbors algorithm (*k*-NN) is a non-parametric method proposed by Thomas Cover used for classification and regression. In both cases, the input consists of the *k* closest training examples in the feature space. The output depends on whether *k*-NN is used for classification or regression:

In *k-NN classification*, the output is a class membership. An object is classified by a plurality vote of its neighbors, with the object being assigned to the class most common among its *k* nearest neighbors (*k* is a positive integer, typically small). If *k* = 1, then the object is simply assigned to the class of that single nearest neighbor.

In *k-NN regression*, the output is the property value for the object. This value is the average of the values of *k* nearest neighbors.

Confusion Matrix for KNN model:

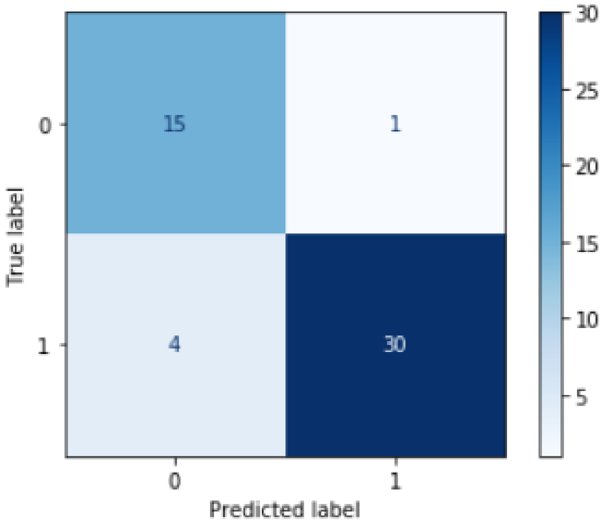


Fig 3.3: Confusion matrix for KNN model

Classification report of KNN:

	Precision	Recall	F1-score
0	0.79	0.94	0.86
1	0.97	0.8	0.92

The accuracy of our KNN model is 90% and our test size is 0.05

Table 3.2: Classification report for KNN

NOTE: n_neighbors=20, metric='euclidean', p=2, weights='uniform'

3.1.3 SVM with PCA

Principal Component Analysis is basically a statistical procedure to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables. Each of the principal components is chosen in such a way so that it would describe most of the still available variance and all these principal components are orthogonal to each other. In all principal components the first principal component has maximum variance.

PCA can be applied to both supervised and unsupervised learning algorithms.PCA is a dimension reduction tool, not a classifier. In scikit-learn all classifiers and estimators have a predict method which PCA does not. As the model requires classification, we need to put a classifier on the PCA transformed data. Here, in our model we are using SVM classifier to the PCA transformed data.SVM here is used in prediction.

Confusion Matrix for SVM with PCA model:

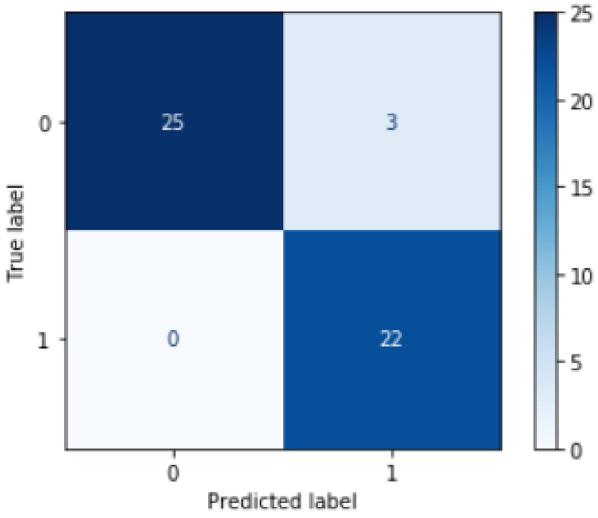


Fig 3.4: Confusion matrix for SVM-PCA

Classification report of SVM with PCA:

	Precision	Recall	F1-score
0	1.00	0.89	0.94
1	0.88	1.0	0.94

The accuracy of our SVM with PCA model is 94% and our test size is 0.05

Table 3.3: Classification report of SVM-PCA

CHAPTER 5: CONCLUSION

ML MODEL	**ACCURACY**
SVM	94%
KNN	90%
SVM-PCA	94%

Table 3.4: Accuracies of three models

The above table denotes the accuracies we achieved using three models - SVM, KNN and SVM-PCA. After obtaining, cleaning and preparing our data, we tried three approaches for generating feature sets for restaurants. We used a Bag of words model for all our approaches.

Using the bag of words model we found that SVM and SVM-PCA gave the highest accuracy of 94% compared to the KNN model which has accuracy of 90%.

Further these algorithms can be deployed using GUIs that helps the customers to find the best place to eat. It also helps the restaurant owners to improve their quality of their products based on the customer feedback.