

**Amazon Fine Food Reviews**

**BUAN 6346.501 - BIG DATA PROJECT**

**Group members:**

**Anil BULUSU: AVB140030**

**Amey VANMALI: AKV170000**

**Sakshi.: sxx180001**

**Siril ALPURI: SXA180003**

**Siri chandana RedDy: SXR180008**

**Project mOTIVATION**

We got together as a group as we're interested in the technical area of analytics which deals with the analysis of data though tools and building the code logic. Also, the industry demands individuals with hands-on experience who have previously dealt with the huge dataset. Implementation will give us the understanding and prepare us beforehand for the challenges that are faced while handling extensive datasets.

When we were brainstorming dataset ideas for the project, we all happened to come across this dataset which we felt is something that everyone can associate themselves with; as all of us check reviews before we go to a restaurant. Everyone looks for whether a restaurant is rated as good or bad which is exactly what we aim to do using the knowledge gained in the classroom. We hope to learn & understand how real-world big data problems are and how our curricular knowledge can be applied practically through this project.

**CONTENTS**

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Topic** | **Page No.** |
| 1 | Project Overview | 4 |
| 2 | Problem Statement | 5 |
| 3 | Platform Used | 8 |
| 4 | Exploratory analysis | 10 |
| 5 | Data Preprocessing | 11 |
| 6 | Implementation | 13 |
| 7 | Model | 15 |
| 8 | Results | 19 |
| 10 | References | 21 |

**Project Overview**

Sentiment Analysis is a fundamental task in Natural Language Processing (NLP). Its uses are many: from analyzing political sentiment on social media, gathering insight from user-generated product reviews or even for financial purposes, such as developing trading strategies based on market sentiment. The goal of most sentiment classification tasks is to identify the overall sentiment polarity of the documents in question, i.e. is the sentiment of the document positive or negative.

Sentiment analysis is critical because it helps us to see what customers like and dislike about a brand. Customer feedback—from social media, websites, call center agents, or any other source—contains a treasure trove of useful business information. But, it isn’t enough to know what customers are talking about. We must also know how they feel. Sentiment analysis is one way to uncover those feelings.

Sometimes known as “opinion mining,” sentiment analysis can let us know if there has been a change in public opinion toward any aspect of our business. Peaks or valleys in sentiment scores give you a place to start if we want to make product improvements, train sales or customer care agents, or create new marketing campaigns.

Sentiment analysis is not a once and done effort. By reviewing customer’s feedback on a business regularly you can be more proactive regarding the changing dynamics in the market place. In this project we are doing the sentiment analysis of amazon fine food review dataset. Our task is to label the reviews given by customers as either positive or negative.

**Problem Statement**

This dataset consists of reviews of fine foods from amazon. The data span a period of more than 10 years, including all ~500,000 reviews up to October 2012. Reviews include product and user information, ratings, and a plain text review. It also includes reviews from all other Amazon categories.

Data Statistics

* 568,454 - Reviews
* 256,059 - Users
* 74,258 - Products
* 260 users with > 50 reviews
* 10 Columns

Our goal is to correctly label the amazon fine food reviews as either positive or negative. This is a supervised learning task so we will be employing various supervised learning techniques.

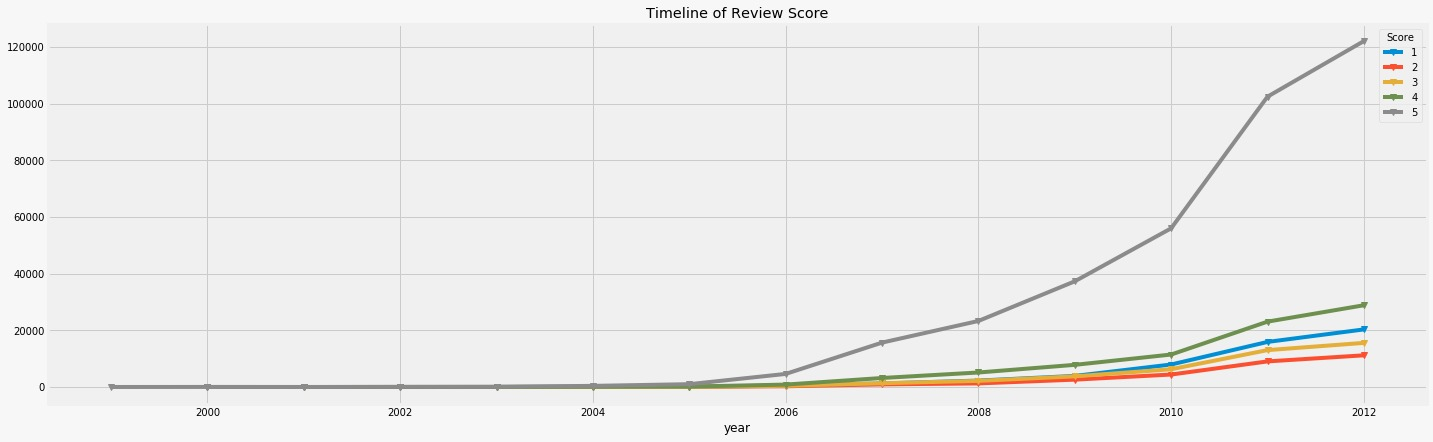


Fig 1: The timeline of reviews over the years

Each review has the following 10 features:

* Id
* ProductId - unique identifier for the product
* UserId - unique identifier for the user
* ProfileName
* HelpfulnessNumerator - number of users who found the review helpful
* HelpfulnessDenominator - number of users who indicated whether they found the review helpful
* Score - rating between 1 and 5
* Time - timestamp for the review
* Summary - brief summary of the review
* Text - text of the review

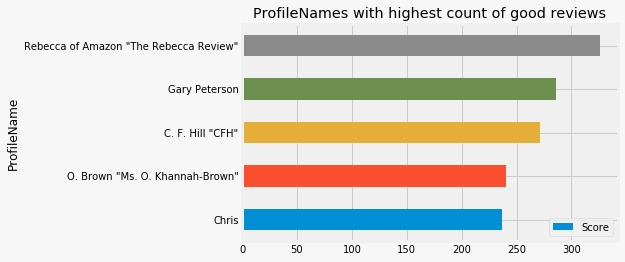
.

Fig 2: Profile Names with highest count of good reviews

## 

Fig 3: Profile Names with highest count of bad reviews

Metrics

The accuracy\_score function computes the accuracy, either the fraction (default) or the count (normalize=False) of correct predictions.

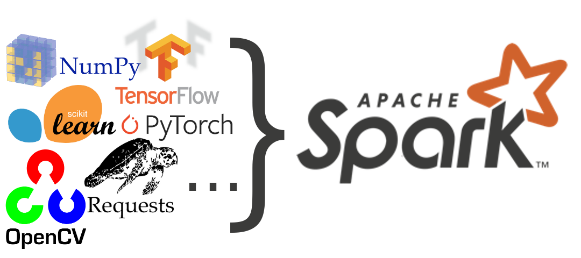
**Platform** **USed**

PySpark on Databricks

Our motivation behind the project was to apply the academic knowledge to solve the big data challenge, which in our case was the volume of data.

we used python on spark via databricks platform, which gave us the freedom to run the entire dataset on the cluster.

PySpark

[Apache Spark](https://www.edureka.co/blog/spark-tutorial/) is one the most widely used frameworks when it comes to handling and working with Big Data and Python is one of the most widely used programming languages for Data Analysis, Machine Learning, and much more. So, why not use them together? This is where Spark with Python also known as PySpark comes into the picture.

PySpark SparkContext and Data Flow

Talking about Spark with Python, working with RDDs is made possible by the library Py4j. PySpark Shell links the Python API to Spark Core and initializes the Spark Context. Spark Context is at the heart of any Spark application.

* Spark Context sets up internal services and establishes a connection to a Spark execution environment.
* The Spark Context object in driver program coordinates all the distributed processes and allows for resource allocation.
* Cluster Managers provide Executors, which are JVM processes with logic.
* Spark Context objects send the application to executors.
* Spark Context executes tasks in each executor.

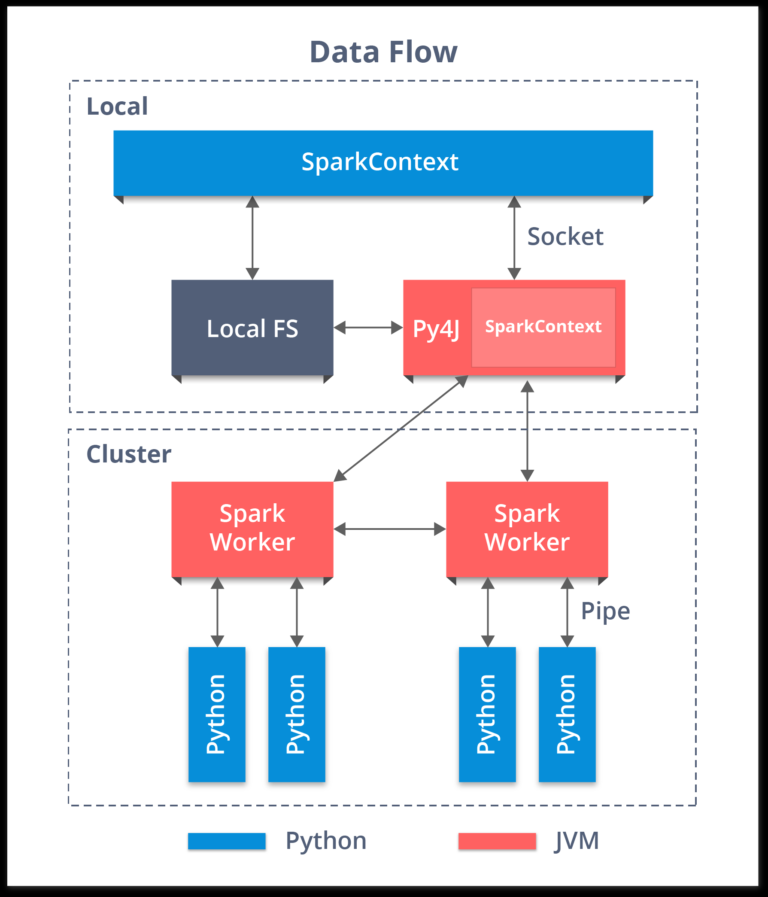


Fig 4: Data flow

Databricks:



Databricks is a company founded by the creators of [Apache Spark](https://en.wikipedia.org/wiki/Apache_Spark), that aims to help clients with cloud-based big data processing using Spark. Databricks grew out of the [AMPLab](https://en.wikipedia.org/wiki/AMPLab) project at [University of California, Berkeley](https://en.wikipedia.org/wiki/University_of_California,_Berkeley) that was involved in making [Apache Spark](https://en.wikipedia.org/wiki/Apache_Spark), a distributed computing framework built atop [Scala](https://en.wikipedia.org/wiki/Scala_(programming_language)). Databricks develops a web-based platform for working with Spark, that provides automated cluster management and [IPython](https://en.wikipedia.org/wiki/IPython)-style [notebooks](https://en.wikipedia.org/wiki/Notebook_interface). In addition to building the Databricks platform, the company is co-organizing [massive open online courses](https://en.wikipedia.org/wiki/Mooc) about Spark[[4]](https://en.wikipedia.org/wiki/Databricks#cite_note-4) and runs the largest conference about Spark - Spark Summit.

*For this Project we have used Databricks Community Edition*.

**exploratory Analysis**

Data Exploration

We have five-star rating system. We will treat rating 3, 4 and 5 as positive reviews (1) and 1 and 2 as negative reviews (0) (Ordinal to categorical conversion).

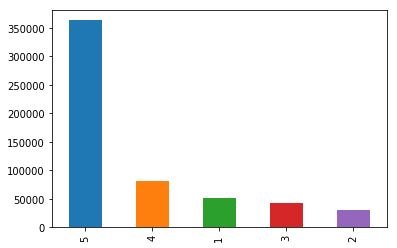


Fig 5: No of Reviews vs 5-star rating

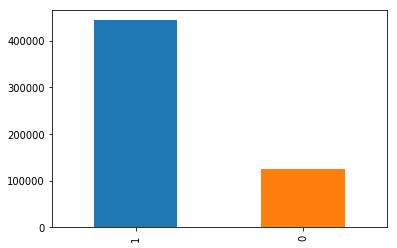


Fig 6: Positive and Negative reviews (1=Positive, 0=Negative) count

**Data Preprocessing**

We cannot feed the raw text data directly to any classification model. We need to convert the raw text data in to some sort of numbers so that they can be used as an input or output features for the classification model.

Further we need to clean our text data as it will reduce the size of the text and removes the features that are unnecessary and does not contain any valuable information.

Data Cleaning and Text Preprocessing - Discard empty rows and columns

* Eliminate duplicate records
* Ordinal to categorical (formatting data)
* Stemming - Converting the words into their base word or stem word (Ex - tastefully, tasty, these words are converted to stem word called 'tasti'). This reduces the vector dimension because we don’t consider all similar words
* Removing Hashes or special characters, Punctuations, Html tags,
* Removing stop words – Stop words are the unnecessary words that even if they are removed the sentiment of the sentence doesn't change. Forex - This pasta is so tasty ==> pasta tasty (This, is, so are stop words so they are removed)

Our input and output feature is in textual form so we need to convert them into numbers. For our output we have simply assigned 1 to the scores greater than 3 and 0 otherwise but our input feature is a long text so to vectorize it we have used the below 2 methods:

[CountVectorizer](http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html)

The [CountVectorizer](http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html) provides a simple way to both tokenize a collection of text documents and build a vocabulary of known words, but also to encode new documents using that vocabulary.

An encoded vector is returned with a length of the entire vocabulary and an integer count for the number of times each word appeared in the document. This implementation produces a sparse representation of the counts using scipy.sparse.csr\_matrix. If you do not provide an a-priori dictionary and you do not use an analyzer that does feature selection, then the number of features will be equal to the vocabulary size found by analyzing the data.

TF – IDF (term frequency–inverse document frequency)

TF-IDF, short for term frequency–inverse document frequency, is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus It is often used as a weighting factor in searches of information retrieval, text mining, and user modeling.

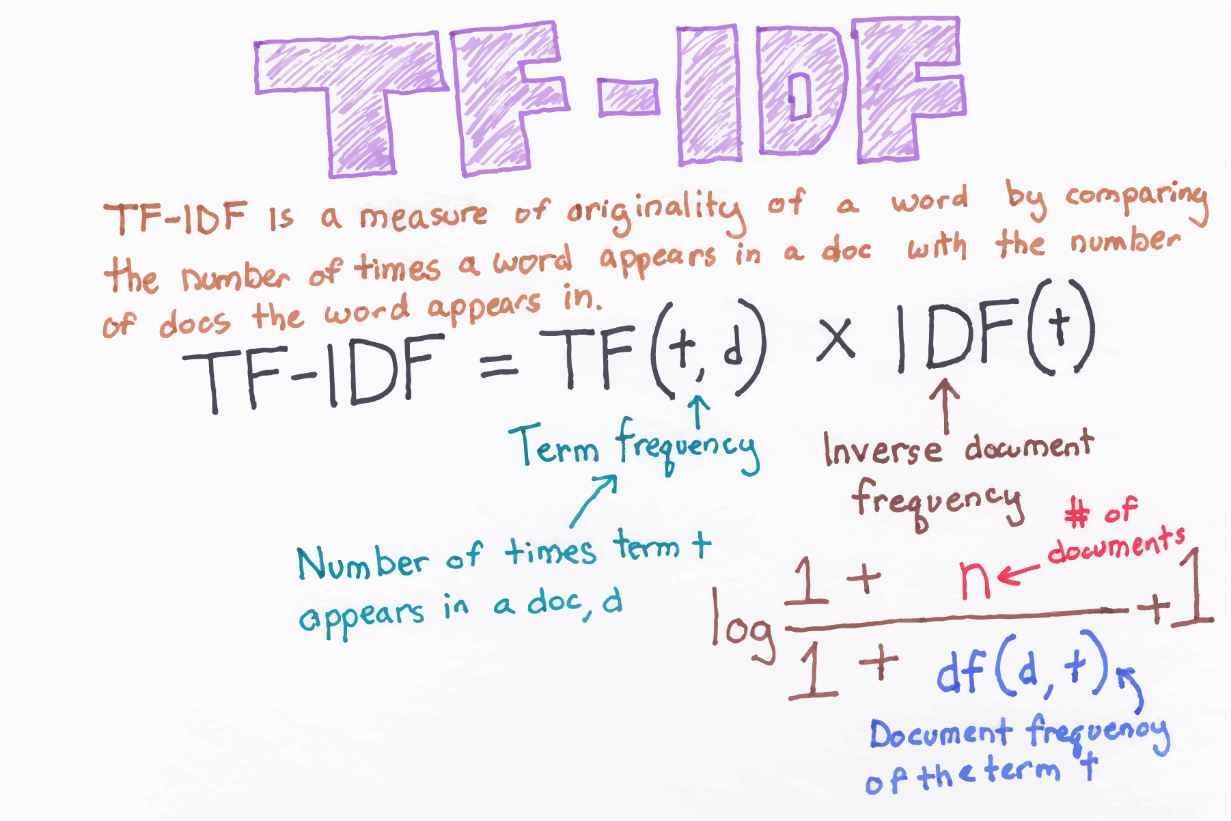


Fig 7: TF-IDF equation explanation

The tf-idf value increases proportionally to the number of times a word appears in the document and is offset by the frequency of the word in the corpus, which helps to adjust for the fact that some words appear more frequently in general the tf–idf is the product of two statistics, term frequency and inverse document frequency. Term frequency is the number of times the term occurs in document.  
The inverse document frequency is a measure of how much information the word provides, that is, whether the term is common or rare across all documents. It is the logarithmically scaled inverse fraction of the documents that contain the word, obtained by dividing the total number of documents by the number of documents containing the term, and then taking the logarithm of that quotient.

**Implementation**

After all the text cleaning steps we get the clean data of customer reviews which is our input feature. For the output feature we have labeled the ‘score’ feature from dataset as positive if score is greater than 3 and negative otherwise.

Word Clouds of Positive and Negative reviews after summary cleaning

A word cloud is an image composed of words used in a text or subject, in which the size of each word indicates its frequency or importance. The larger the text size the higher its importance and occurrence.

Below we can see the Word Cloud of Positive Reviews. We can see the words *great, good, candy, love, treat, natural, food* appear in large fonts, it means that in positive reviews these words are most common.



Fig 8: Word Cloud of Positive Reviews

Below we can see the Word Cloud of Negative Reviews. We can see the words *horrible, rip, bitter, warning, taste, chemical, garbage,* appear in large fonts, it means that in negative reviews these words are most common, which somewhat makes sense.

s

**Fig 9:** Word Cloud of Negative Reviews

Then we split the data into training and testing sets. After vectorizing the input feature, we can finally feed into supervised learning models.

**Models to predict good and bad reviews**

Text Classification Models

Since it’s a classification problem of good or bad reviews, we started with Logistic Regression.   
We must explore other text classification algorithms, to improve the accuracy.

1. Logistic Regression: -

Logistic regression falls under the category of supervised learning; it measures the relationship between the categorical dependent variable and one or more independent variables by estimating probabilities using a logistic/sigmoid function. In spite of the name ‘logistic regression’, this is not used for regression problem where the task is to predict the real-valued output. It is a classification problem which is used to predict a binary outcome (1/0, -1/1, True/False) given a set of independent variables.

**Confusion Matrix on test set**

Accuracy = 77.74%

[[ 1549 36512]

[ 1469 131130]]

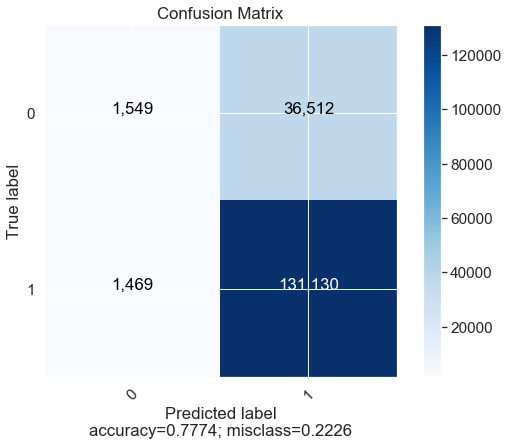


Fig 10: Confusion Matrix of Logistic Regression

1. Multinomial Naïve Bayes: -

Multinomial Naive Bayes is a specialized version of Naive Bayes that is designed more for text documents. Whereas simple naive Bayes would model a document as the presence and absence of particular words, multinomial naive Bayes explicitly models the word counts and adjusts the underlying calculations to deal with in. It estimates the conditional probability of a particular word given a class as the relative frequency of term t in documents belonging to class(c). The variation takes into account the number of occurrences of term t in training documents from class (c), including multiple occurrences.

**Confusion Matrix on test set**

Accuracy = 77.04%

Confusion Matrix

[[ 4578 33483]

[ 5695 126904]]

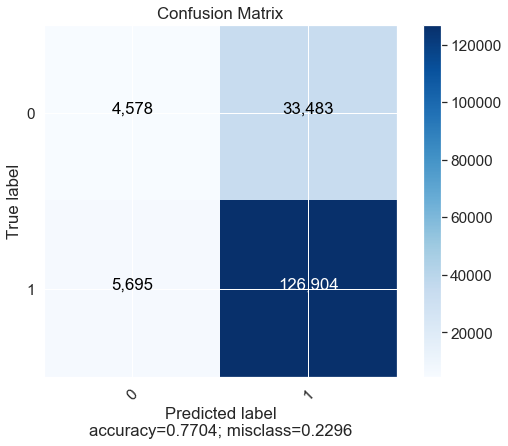


Fig 11: Confusion Matrix of Multinomial Naïve Bayes

1. Support Vector Machines: -

“Support Vector Machine” (SVM) is a supervised [machine learning algorithm](https://courses.analyticsvidhya.com/courses/introduction-to-data-science-2?utm_source=blog&utm_medium=understandingsupportvectormachinearticle) which can be used for both classification and regression challenges. However, it is mostly used in classification problems. In this algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiate the two classes very well.

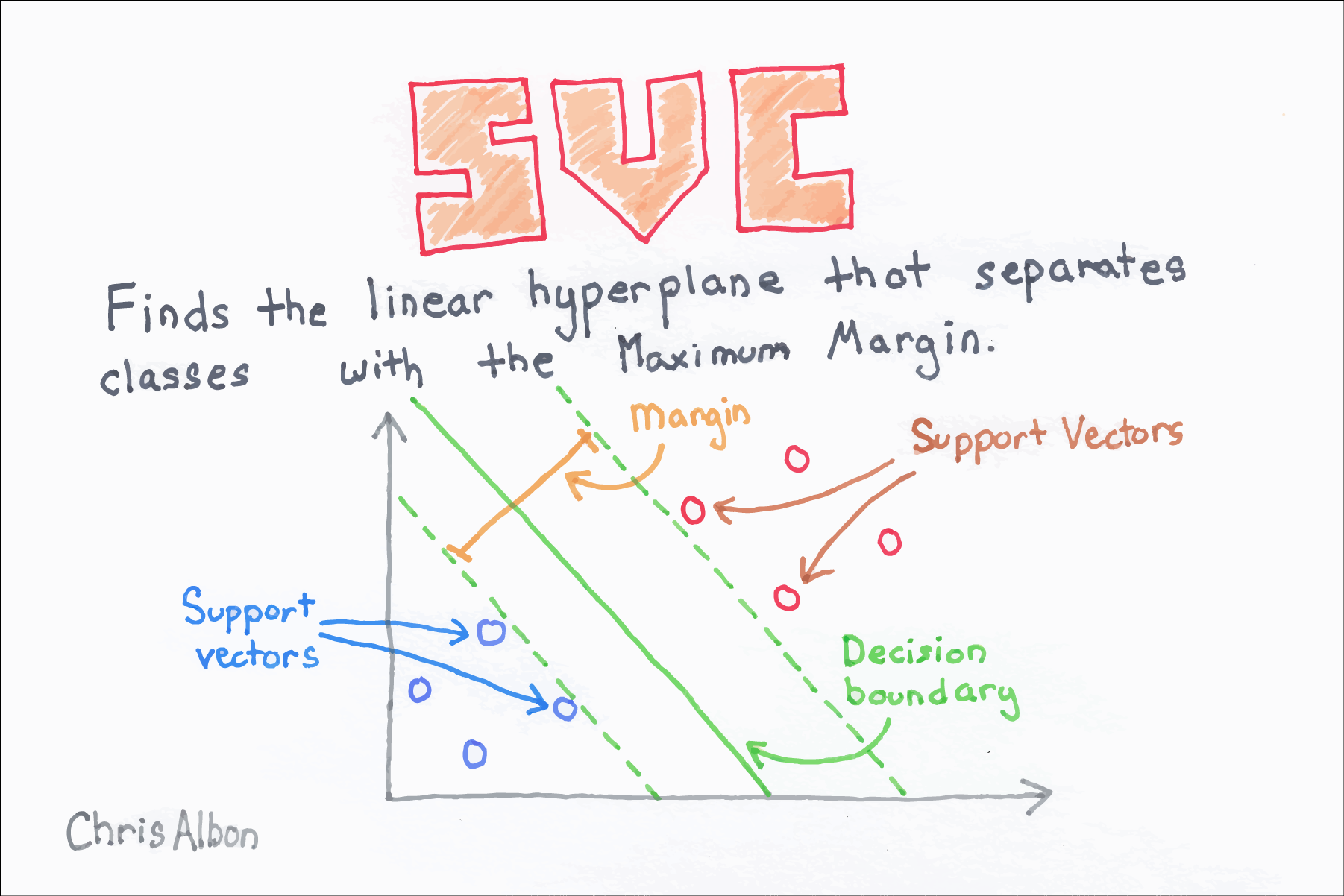


Fig 12: Support Vector Classifier explanation

There is a balance between SVC maximizing the margin of the hyperplane and minimizing the misclassification. In SVC, the latter is controlled with the hyperparameter CC, the penalty imposed on errors. C is a parameter of the SVC learner and is the penalty for misclassifying a data point. When C is small, the classifier is okay with misclassified data points (high bias but low variance). When C is large, the classifier is heavily penalized for misclassified data and therefore bends over backwards avoid any misclassified data points (low bias but high variance).

In scikit-learn, CC is determined by the parameter C and defaults to C=1.0. We should treat C has a hyperparameter of our learning algorithm which we tune using model selection techniques.

Support Vectors are simply the co-ordinates of individual observation. Support Vector Machine is a frontier which best segregates the two classes (hyper-plane/ line).

Here we have used Support Vector Classifier since it’s a classification problem.

**Confusion Matrix on test set**

Accuracy on test set: 83.52%.

Confusion Matrix

[[ 16571 21490]

[ 6637 125962]]

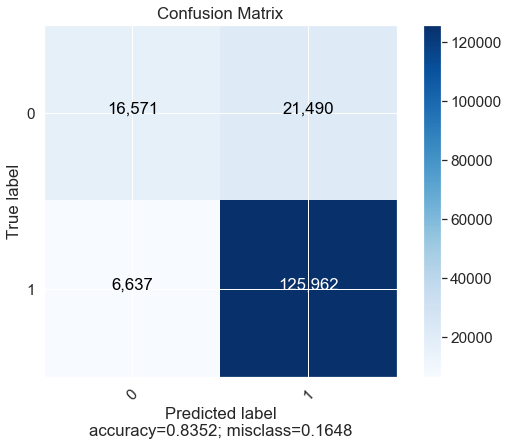


Fig 13: Confusion Matrix of Support Vector Machine

**Results**

Model evaluation and validation

Since the model is evaluated based on accuracy so the model with highest accuracy will be chosen. For our case SVM classifier has got highest accuracy, so we have chosen the SVM classifier for this project.

|  |  |
| --- | --- |
| **Algorithms** | **Accuracy** |
| Multinomial NB | 77.04% |
| Logistic Regression | 77.74% |
| SVM Classifier | 83.52% |

Table 1: Results Summary

**Refrences:**

Below are the list of sites, which helped us in completing this project

* <https://www.kaggle.com/snap/amazon-fine-food-reviews>
* <https://chrisalbon.com/machine_learning/support_vector_machines/support_vector_classifier/>
* [https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine- example-code/](https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-%20%20example-code/)
* <https://en.wikipedia.org/wiki/Tf%E2%80%93idf>
* <https://en.wikipedia.org/wiki/Logistic_regression>
* <https://machinelearningmastery.com/prepare-text-data-machine-learning-scikit-learn/>
* <https://www.clarabridge.com/sentiment-analysis/>
* <https://www.quora.com/How-does-multinomial-Naive-Bayes-work>
* <https://www.kaggle.com/grfiv4/plot-a-confusion-matrix>
* <http://ceur-ws.org/Vol-2086/AICS2017_paper_21.pdf>
* <https://en.wikipedia.org/wiki/Databricks>
* <https://dzone.com/articles/introduction-to-spark-with-python-pyspark-for-begi>
* [https://miro.medium.com/max/1080/1\*rOAZN14LvtiBgUhOcITbwg.png](https://miro.medium.com/max/1080/1*rOAZN14LvtiBgUhOcITbwg.png)
* <https://chrisalbon.com/images/machine_learning_flashcards/TF-IDF_print.png>