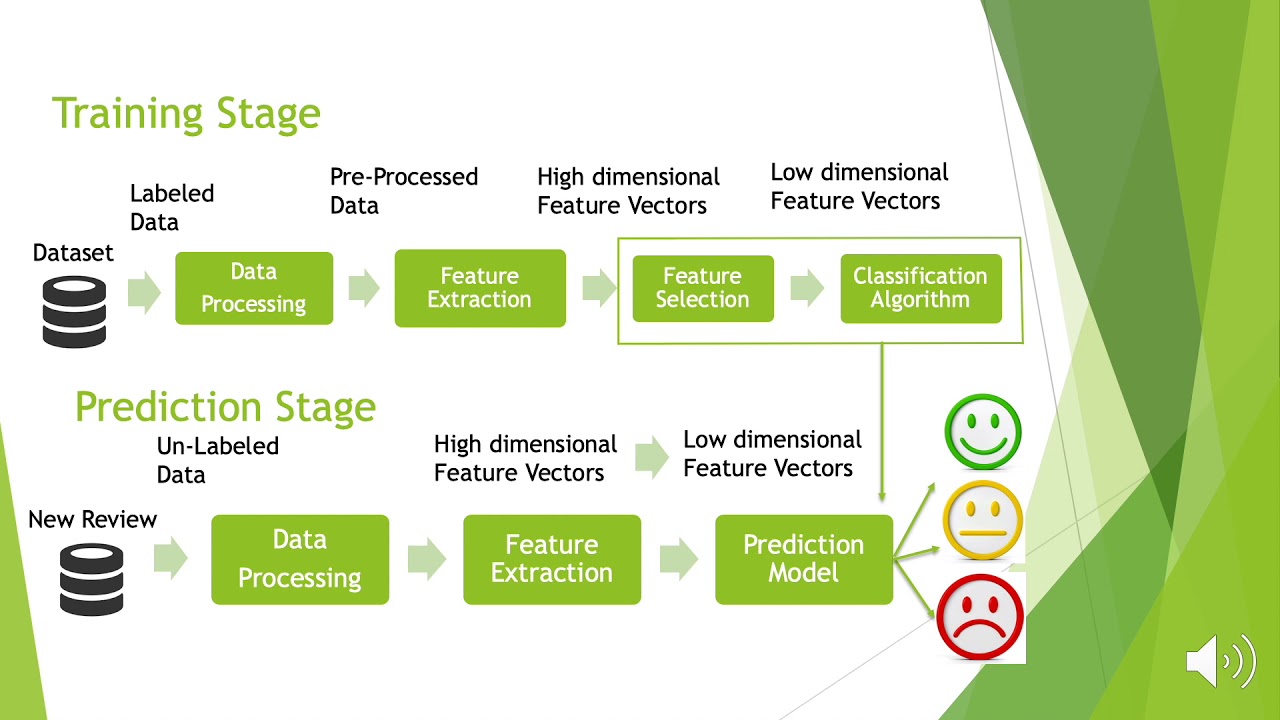
**SENTIMENT CLASSIFICATION AND OPINION MINING ON AIRLINE REVIEWS**







Chapter 1 Introduction

Contents

Page No

* 1. [Overview 1](#_TOC_250006)
  2. [Purpose](#_TOC_250005)

Chapter 2 Literature Survey

* 1. [Existing Problem](#_TOC_250004)  2
  2. [Proposed Solution](#_TOC_250003)

Chapter 3 Theoretical Analysis

* 1. [Block diagram](#_TOC_250002) 3
  2. Hardware/Software Designing

[Chapter 4 System Development](#_TOC_250001)

* 1. [Introduction 4](#_TOC_250000)
  2. Machine Learning
     1. Machine Learning Methods 5
     2. Code Implementation 8
  3. Data Cleansing
     1. Introduction 13
  4. Deployment Diagram 17

#### Chapter 5 Algorithms

* 1. Decision Tree 18
  2. Random Forest
  3. Logistic Regression 19
  4. SVM

#### Chapter 6 Implementation

* 1. Installation and Execution Steps 20
     1. Keras
     2. Tensorflow 21
     3. OpenCV 23
     4. Downloading and Installing OpenCV
     5. Scipy
  2. Conclusion

#### Chapter 7 Program Code

* 1. Code.py 24
  2. App.py 30
  3. Html code 31

Chapter 8 Flow Chart 35

#### Chapter 9 Output 36

#### Chapter 10 Advantages and Disadvantages 42

#### Chapter 11 Applications 43

#### Chapter 12 Conclusion and Future Scope 44

#### Chapter 13 References 45

# List of Figures

|  |  |  |
| --- | --- | --- |
|  | Page No | |
| Figure 3.1.1 | Block Diagram | 3 |
| Figure 4.2.1 | Applications of Machine Learning | 7 |
| Figure 4.2.2.1 | Machine Learning Process | 8 |
| Figure 4.2.2.2 | Figure | 8 |
| Figure 4.2.2.3 | Data Processing | 10 |
| Figure 4.2.2.4 | Figure | 12 |
| Figure 4.3.1 | Figure | 14 |
| Figure 4.4.1 | Deployment Diagram | 17 |
| Figure 5.1.1 | Decision Tree | 18 |
| Figure 5.1.2 | Random Forest | 18 |
| Figure 5.1.3 | Logistic Regression | 19 |
| Figure 5.1.4 | SVM | 19 |

## Abstract

Opinion Mining or Sentiment Analysis can be defined as the task of detecting, extracting and classifying opinions on something. It is a type of the processing of the natural language (NLP) to track the public mood to a certain law, policy, or marketing, etc. It involves a way that development for the collection and examination of comments and opinions about legislation, laws, policies, etc., which are posted on the social media. The process of information extraction is very important because it is a very useful technique but also a challenging task. That mean, to extract sentiment from an object in the world-wide, need to automate opinion mining systems to do it. The existing techniques for sentiment analysis include machine learning (supervised and unsupervised), and lexical-based approaches. As, sentiment analysis is gaining importance in the research study of text mining and natural language processing (NLP). There has been a rise in accessibility of online applications and a surge in social platforms for opinion sharing, online review websites, and personal blogs, which have captured the attention of stakeholders such as customers, organizations, and governments to analyze and explore these opinions. Therefore, the main aim of sentiment classification is to analyze an online document such as a blog, comment, review and new items as a comprehensive statements and categories it as positive, negative and neutral.

#### Overview

## Chapter 1

### Introduction

Twitter has great popularity as an online social networking service in recent years, tweets on twitter become a valuable source of information for companies to get feedback s from their customers. However, extracting such information from seemingly random comments and reviews is not an easy task due to the scale of data size and the difficulty in semantic analysis. For example, someone tweets: “@Virgin America Hey first time flyer next week – excited! But I’m having a hard time getting my flights added to my Elevate account. Help?” Is she happy with the flight experience or not? If not, what makes her unhappy? Having machines answer these types of questions and learn the sentiment of each tweet automatically will be beneficial for service providers to understand the strength and weakness of their products and to improve them in the future.

In this project we focus on reviews on twitter for major U.S. airlines and try to extract sentiment and opinions from these reviews. We explored Natural Language processing to predict the sentiment of these tweets – positive, neutral or negative (referred as sentiment task).

#### Purpose

These days, sentiment analysis is gaining importance in the research study of text mining and natural language processing (NLP). There has been a rise in accessibility of online applications and a surge in social platforms for opinion sharing, online review websites, and personal blogs, which have captured the attention of stakeholders such as customers, organizations, and governments to analyze and explore these opinions. Therefore, the major role of sentiment classification is to analyze an online document such as a blog, comment, review and new items as a comprehensive sentiment and categories it as positive, negative, or neutral. Analyse and discuss the related topic in the field of sentiment analysis, Natural Language Processing techniques and lexicon resource creating and implementation approaches. Review state-of-art sentiment classification approaches. Investigate the advantages and disadvantages of these approaches. Investigate the existing text mining techniques in the field of sentiment analysis. Design and implement approaches reviewed from previous objectives. Evaluating the classification result obtained from baseline classifier using various measurement techniques investigated. Design and construct proposed classification method using equivalent configurations implemented in baseline classification approach. Evaluation and analysis results obtained from baseline classifier and proposed classification strategy. Critical investigation on obtained results of classification and error of misclassification for the proposed ensemble classification approaches.

#### Existing Problem

## Chapter 2

### Literature Survey

Opinion Mining (OM) or Sentiment Analysis (SA) can be defined as the task of detecting extracting and classifying opinions on something. It is a type of the processing of the natural language (NLP) to track the public mood to a certain law, policy, or marketing, etc. It involves a way that development for the collection and examination of comments and opinions about legislation, laws, policies, etc., which are posted on the social media. The process of information extraction is very important because it is a very useful technique but also a challenging task. That means to extract sentiment from an object in the web wide, need to automate opinion-mining systems to do it. The existing techniques for sentiment analysis include machine learning (supervised and unsupervised), and lexical-based approaches. Hence, the main aim of this project presents a survey of sentiment analysis (SA) and opinion mining (OM) approaches, various techniques used that related ion this field.

#### Proposed Solution

These days ,sentiment analysis is gaining importance in the research study of text mining and natural language processing (NLP).There has been a rise in accessibility of online applications and a surge in social platforms for opinion sharing ,online review websites ,and personal blogs ,which have captured the attention of shake holders such as customers, organizations and governments to analyze and explore these opinions Therefore, the major role of sentiment classification is to analyze an online document such as a blog, comment, review and new items as a comprehensive sentiment and categories it as positives, negative, or neutral.

Research performed on Airline industry particularly by employing sentiment analysis based on aspect is minimal in the part. Significant contribution has been provided by this research to the customers in deciding the airlines and also helps in bridging the gap between the carriers and the customers. This research greatly aids the US airlines to look for the areas of improvement and could easily make comparison of their performances with their competitors for obtaining a better competitive edge in the market. In this research, there are short comings too. Only for English language, the model used for this research can be applied. Since there will be different grammatical structure for different languages. Thus for other language this model will not work for other languages. Further, it requires users input for adding or changing the existing feature.

#### Block Diagram

## Chapter 3

### Theoretical Analysis

The following block diagram describes the flow of processes involves the steps of extraction of sentiments by the stakeholders through natural language processing to analyze the tweets whether the tweets are negative, positive or neutral.

As shown below in the block diagram, natural language processing involves following steps:

1. Data Collection
2. Text processing
3. Model Building
4. Application Building

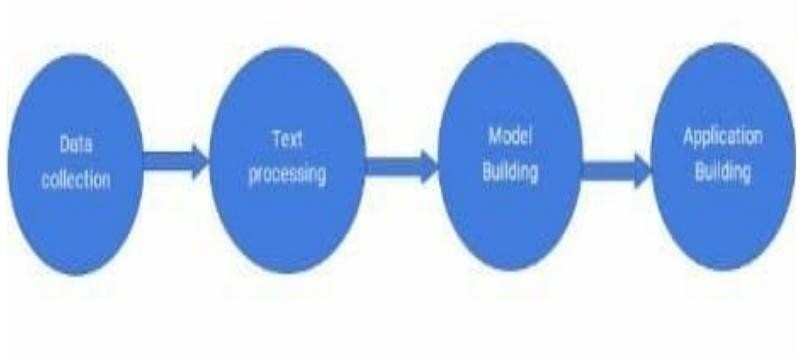


Figure 3.1.1 NLP Process

#### Hardware/Software Designing Hardware Design:

* + - Laptop
    - RAM-8GB
    - Hard Disk

#### Software Design:

* + - Anaconda
    - Python 3.7
    - Spyder
    - HTML and CSS

## Chapter 4 System Development

### Introduction

In this chapter we will discuss about the existing system and the proposed systems in detail.

Images are captured by digital camera mobile and processed using image growing technology.

### Machine Learning

Machine learning focuses on applications that learn from experience and improve their decision- making or predictive accuracy over time.

Machine learning is a branch of artificial intelligence (AI) focused on building applications that learn from data and improve their accuracy over time without being programmed to do so.

In data science, an algorithm is a sequence of statistical processing steps. In machine learning, algorithms are 'trained' to find patterns and features in massive amounts of data in order to make decisions and predictions based on new data. The better the algorithm, the more accurate the decisions and predictions will become as it processes more data.

Today, examples of machine learning are all around us. Digital assistants search the web and play music in response to our voice commands. Websites recommend products and movies and songs based on what we bought, watched, or listened to before. Robots vacuum our floors while we do something better with our time. Spam detectors stop unwanted emails from reaching our inboxes. Medical image analysis systems help doctors spot tumors they might have missed. And the first self-driving cars are hitting the road.

We can expect more. As big data keeps getting bigger, as computing becomes more powerful and affordable, and as data scientists keep developing more capable algorithms, machine learning will drive greater and greater efficiency in our personal and work lives.

There are four basic steps for building a machine learning application (or model). These are typically performed by data scientists working closely with the business professionals for whom the model is being developed.

**Step 1**: Select and prepare a training data set

Training data is a data set representative of the data the machine learning model will ingest to solve the problem it’s designed to solve. In some cases, the training data is labeled data ‘tagged’ to call out

features and classifications the model will need to identify. Other data is unlabeled, and the model will need to extract those features and assign classifications on its own.

In either case, the training data needs to be properly prepared—randomized, de-duped, and checked for imbalances or biases that could impact the training. It should also be divided into two subsets: the training subset, which will be used to train the application, and the evaluation subset, used to test and refine it.

**Step 2**: Choose an algorithm to run on the training data set

Again, an algorithm is a set of statistical processing steps. The type of algorithm depends on the type (labeled or unlabeled) and amount of data in the training data set and on the type of problem to be solved.

Common types of machine learning algorithms for use with labeled data include the following: Regression algorithms: Linear and logistic regression are examples of regression algorithms used to understand relationships in data. Linear regression is used to predict the value of a dependent variable based on the value of an independent variable. Logistic regression can be used when the dependent variable is binary in nature: A or B

For example, a linear regression algorithm could be trained to predict a salesperson’s annual sales (the dependent variable) based on its relationship to the salesperson’s education or years of experience (the independent variables.) Another type of regression algorithm called a support vector machine is useful when dependent variables are more difficult to classify.

Decision trees: Decision trees use classified data to make recommendations based on a set of decision rules. For example, a decision tree that recommends betting on a particular horse to win, place, or show could use data about the horse (e.g., age, winning percentage, pedigree) and apply rules to those factors to recommend an action or decision.

Instance-based algorithms: A good example of an instance-based algorithm is KNearest Neighbor or k-nn. It uses classification to estimate how likely a data point is to be a member of one group or another based on its proximity to other data points.

Algorithms for use with unlabeled data include the following:

Clustering algorithms: Think of clusters as groups. Clustering focuses on identifying groups of similar records and labeling the records according to the group to which they belong. This is done without prior knowledge about the groups and their characteristics. Types of clustering algorithms include the K-means, Two Step, and Kohonen clustering. Association algorithms: Association algorithms find patterns and relationships in data and identify frequent ‘if-then’ relationships called association rules. These are similar to the rules used in data mining.

Neural networks: A neural network is an algorithm that defines a layered network of calculations featuring an input layer, where data is ingested; at least one hidden layer, where calculations are performed make different conclusions about input; and an output layer. where each conclusion is assigned a probability. A deep neural network defines a network with multiple hidden layers, each of which successively refines the results of the previous layer. (For more, see the “Deep learning” section below.)

**Step 3**: Training the algorithm to create the model

Training the algorithm is an iterative process–it involves running variables through the algorithm, comparing the output with the results it should have produced, adjusting weights and biases within the algorithm that might yield a more accurate result, and running the variables again until the algorithm returns the correct result most of the time. The resulting trained, accurate algorithm is the machine learning model—an important distinction to note, because 'algorithm' and 'model' are incorrectly used interchangeably, even by machine learning mavens.

**Step 4**: Using and improving the model

The final step is to use the model with new data and, in the best case, for it to improve in accuracy and effectiveness over time. Where the new data comes from will depend on the problem being solved. For example, a machine learning model designed to identify spam will ingest email messages, whereas a machine learning model that drives a robot vacuum cleaner will ingest data resulting from real-world interaction with moved furniture or new objects in the room.

#### Machine learning methods

Machine learning methods (also called machine learning styles) fall into three primary categories. Supervised machine learning:

Supervised machine learning trains itself on a labeled data set. That is, the data is labeled with information that the machine learning model is being built to determine and that may even be classified in ways the model is supposed to classify data. For example, a computer vision model designed to identify purebred German Shepherd dogs might be trained on a data set of various labeled dog images. Supervised machine learning requires less training data than other machine learning methods and makes training easier because the results of the model can be compared to actual labeled results. But, properly labeled data is expensive to prepare, and there's the danger of overfitting, or creating a model so closely tied and biased to the training data that it doesn't handle variations in new data accurately. Unsupervised machine learning:

Unsupervised machine learning ingests unlabeled data lots and lots of it and uses algorithms to extract meaningful features needed to label, sort, and classify the data in real-time, without human intervention. Unsupervised learning is less about automating decisions and predictions, and more about identifying patterns and relationships in data that humans would miss. Take spam detection, for example people generate more email than a team of data scientists could ever hope to label or classify in their lifetimes. An unsupervised learning algorithm can analyze huge volumes of emails and uncover the features and patterns that indicate spam (and keep getting better at flagging spam over time).

Semi-supervised learning:

Semi-supervised learning offers a happy medium between supervised and unsupervised learning. During training, it uses a smaller labeled data set to guide classification and feature extraction from a larger, unlabeled data set. Semi-supervised learning can solve the problem of having not enough labeled data (or not being able to afford to label enough data) to train a supervised learning algorithm.

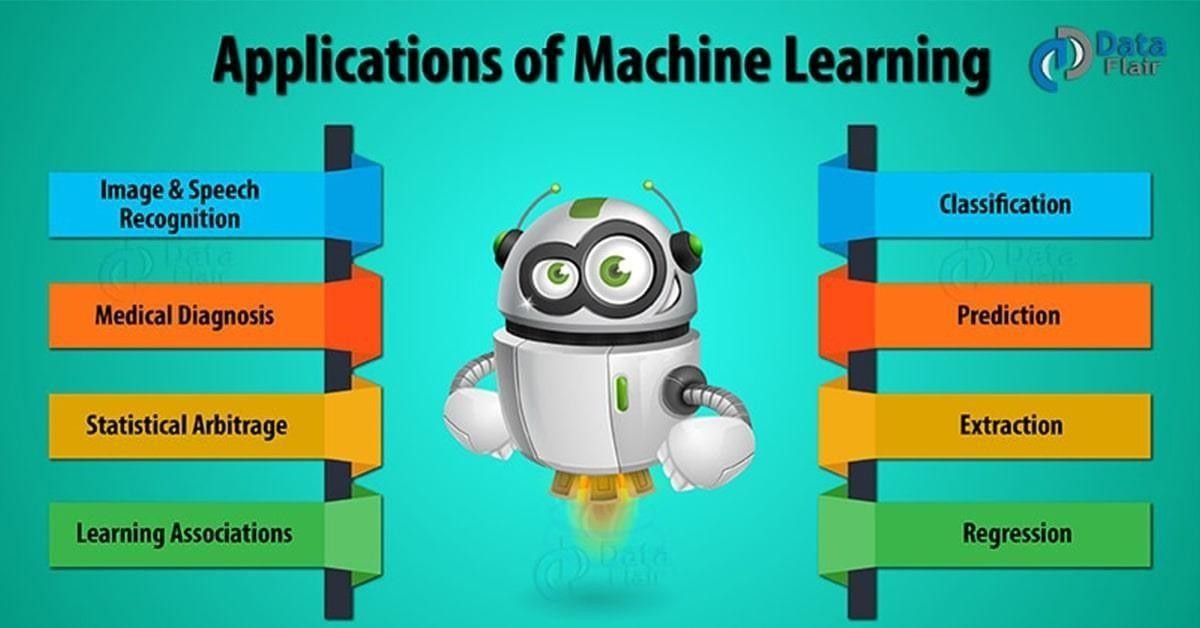


Figure 4.2.1 Applications of Machine Learning

#### Coding Implementation:

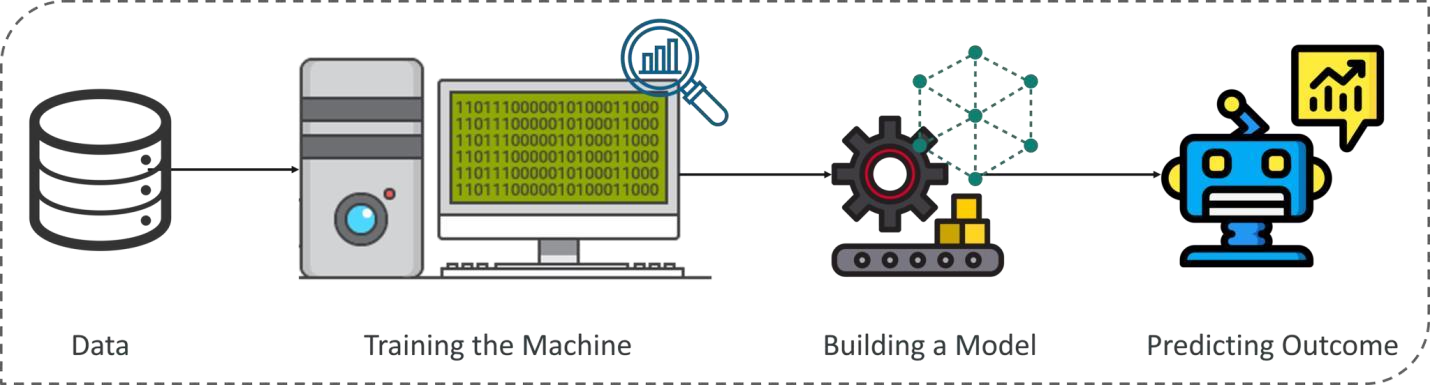


Figure 4.2.2.1 Machine Learning Process How does Machine Learning Work?

Machine Learning algorithm is trained using a training data set to create a model. When new input data is introduced to the ML algorithm, it makes a prediction on the basis of the model.

The prediction is evaluated for accuracy and if the accuracy is acceptable, the Machine Learning algorithm is deployed. If the accuracy is not acceptable, the Machine Learning algorithm is trained again and again with an augmented training data set.

The Machine Learning process involves building a Predictive model that can be used to find a solution for a Problem Statement. To understand the Machine Learning process let’s assume that you have been given a problem that needs to be solved by using Machine Learning.

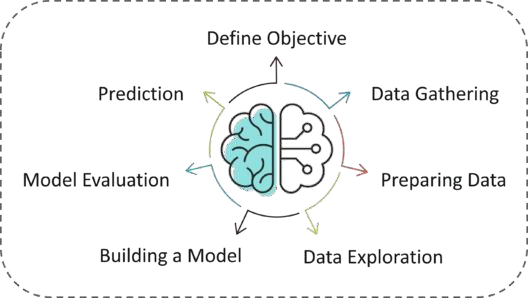


Figure 4.2.2.2

The below steps are followed in a Machine Learning process:

**Step 1:** Define the objective of the Problem Statement

At this step, we must understand what exactly needs to be predicted. In our case, the objective is to predict the possibility of rain by studying weather conditions. At this stage, it is also essential to take mental notes on what kind of data can be used to solve this problem or the type of approach you must follow to get to the solution.

**Step 2:** Data Gathering

At this stage, you must be asking questions such as,

* What kind of data is needed to solve this problem?
* Is the data available?
* How can I get the data?

Once you know the types of data that is required, you must understand how you can derive this data. Data collection can be done manually or by web scraping. However, if you’re a beginner and you’re just looking to learn Machine Learning you don’t have to worry about getting the data. There are 1000s of data resources on the web, you can just download the data set and get going.

Coming back to the problem at hand, the data needed for weather forecasting includes measures such as humidity level, temperature, pressure, locality, whether or not you live in a hill station, etc. Such data must be collected and stored for analysis.

**Step 3**: Data Preparation

The data you collected is almost never in the right format. You will encounter a lot of inconsistencies in the data set such as missing values, redundant variables, duplicate values, etc. Removing such inconsistencies is very essential because they might lead to wrongful computations and predictions. Therefore, at this stage, you scan the data set for any inconsistencies and you fix them then and there.

Data Processing:

Data Processing is a task of converting data from a given form to a much more usable and desired form i.e. making it more meaningful and informative. Using Machine Learning algorithms, mathematical modeling and statistical knowledge, this entire process can be automated. The output of

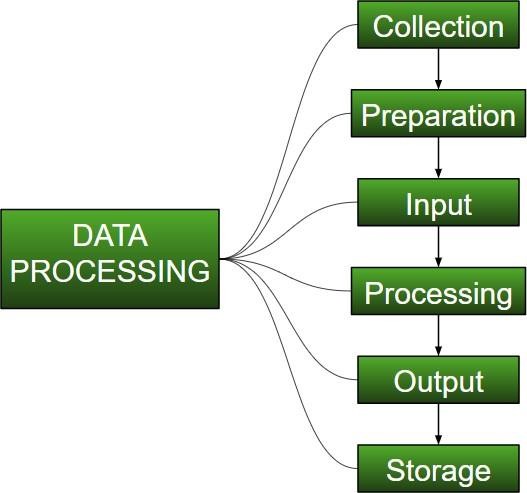
this complete process can be in any desired form like graphs, videos, charts, tables, images and many more, depending on the task we are performing and the requirements of the machine. This might seem to be simple but when it comes to really big organizations like Twitter, Facebook, Administrative bodies like Paliament, UNESCO and health sector organizations, this entire process needs to be performed in a very structured manner.

Figure 4.2.2.3 Data Processing

Collection:

The most crucial step when starting with ML is to have data of good quality and accuracy. Data can be collected from any authenticated source like [data.gov.in](https://data.gov.in/), [Kaggle](https://www.kaggle.com/) or [UCI dataset repository.](https://archive.ics.uci.edu/ml/datasets.html)

For example, while preparing for a competitive exam, student’s study from the best study material that they can access so that they learn the best to obtain the best results. In the same way, high-quality and accurate data will make the learning process of the model easier and better and at the time of testing, the model would yield state of the art results. A huge amount of capital, time and resources are consumed in collecting data. Organizations or researchers have to decide what kind of data they need to execute their tasks or research. Example: Working on the Facial Expression Recognizer, needs a large number of images having a

variety of human expressions. Good data ensures that the results of the model are valid and can be trusted upon.

Preparation:

The collected data can be in a raw form which can’t be directly fed to the machine. So, this is a process of collecting datasets from different sources, analyzing these datasets and then constructing a new dataset for further processing and exploration. This preparation can be performed either manually or from the automatic approach. Data can also be prepared in numeric forms also which would fasten the model’slearning.

**Example:** An image can be converted to a matrix of N X N dimensions, the value of each cell will indicate image pixel.

##### Input:

Now the prepared data can be in the form that may not be machine-readable, so to convert this data to readable form, some conversion algorithms are needed. For this task to be executed, high computation and accuracy is needed. Example: Data can be collected through the sources like MNIST Digit data (images), twitter comments, audio files, video clips.

##### Processing:

This is the stage where algorithms and ML techniques are required to perform the instructions provided over a large volume of data with accuracy and optimal computation.

##### Output:

In this stage, results are procured by the machine in a meaningful manner which can be inferred easily by the user. Output can be in the form of reports, graphs, videos, etc

##### Storage:

This is the final step in which the obtained output and the data model data and all the useful information are saved for the future use.

Data Preprocessing for Machine learning in Python **4**

* + Pre-processing refers to the transformations applied to our data before feeding it to the algorithm.
  + Data Preprocessing is a technique that is used to convert the raw data into a clean data set. In other words, whenever the data is gathered from different sources it is collected in raw format which is not feasible for the analysis.

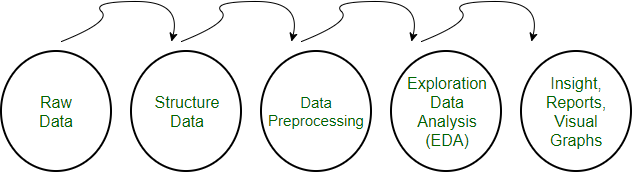


Figure 4.2.2.4

Need of Data Preprocessing

* + For achieving better results from the applied model in Machine Learning projects the format of the data has to be in a proper manner. Some specified Machine Learning model needs information in a specified format, for example, Random Forest algorithm does not support null values, therefore to execute random forest algorithm null values have to be managed from the original raw data set.
  + Another aspect is that data set should be formatted in such a way that more than one Machine Learning and Deep Learning algorithms are executed in one data set, and best out of them is chosen.

Rescale Data

* When our data is comprised of attributes with varying scales, many machine learning algorithms can benefit from rescaling the attributes to all have the same scale.
* This is useful for optimization algorithms in used in the core of machine learning algorithms like gradient descent.
* It is also useful for algorithms that weight inputs like regression and neural networks and algorithms that use distance measures like K-Nearest Neighbors.
* We can rescale your data using scikit-learn using the MinMaxScaler class.

Binarize Data (Make Binary)

* We can transform our data using a binary threshold. All values above the threshold are marked 1 and all equal to or below are marked as 0.
* This is called binarizing your data or threshold your data. It can be useful when you have probabilities that you want to make crisp values. It is also useful when feature engineering and you want to add new features that indicate something meaningful.
* We can create new binary attributes in Python using scikit-learn with the Binarizer class.

Standardize Data

* Standardization is a useful technique to transform attributes with a Gaussian distribution and differing means and standard deviations to a standard Gaussian distribution with a mean of 0 and a standard deviation of 1.
* We can standardize data using scikit-learn with the Standard Scaler class.

### Data Cleansing

#### Introduction:

Data cleaning is one of the important parts of machine learning. It plays a significant part in building a model. Data Cleaning is one of those things that everyone does but no one really talks about. It surely isn’t the fanciest part of machine learning and at the same time, there aren’t any hidden tricks or secrets to uncover. However, proper data cleaning can make or break your project. Professional data scientists usually spend a very large portion of their time on this step.

Because of the belief that, “Better data beats fancier algorithms”.

If we have a well-cleaned dataset, we can get desired results even with a very simple algorithm, which can prove very beneficial at times.

Obviously, different types of data will require different types of cleaning. However, this systematic approach can always serve as a good starting point.

#### Steps involved in Data Cleaning

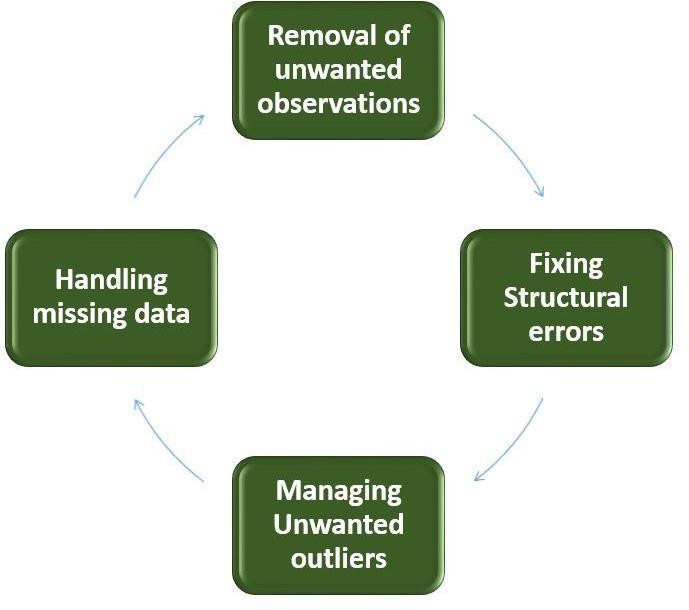


Figure 4.3.1

##### Removal of unwanted observations

This includes deleting duplicate/ redundant or irrelevant values from your dataset. Duplicate observations most frequently arise during data collection and irrelevant observations are those that don’t actually fit the specific problem that you’re trying to solve.

* Redundant observations alter the efficiency by a great extent as the data repeats and may add towards the correct side or towards the incorrect side, thereby producing unfaithful results.
* Irrelevant observations are any type of data that is of no use to us and can be removed directly.

1. Fixing Structural errors

The errors that arise during measurement transfer of data or other similar situations are called structural errors. Structural errors include typos in the name of features, same attribute with different name, mislabeled classes, i.e. separate classes that should really be the same or inconsistent capitalization.

* For example, the model will treat America and america as different classes or values, though they represent the same value or red, yellow and red-yellow as different classes or attributes, though one class can be included in other two classes. So, these are some structural errors that make our model inefficient and gives poor quality results.

1. Managing Unwanted outliers

Outliers can cause problems with certain types of models. For example, linear regression models are less robust to outliers than decision tree models. Generally, we should not remove outliers until we have a legitimate reason to remove them. Sometimes, removing them improves performance,

sometimes not. So, one must have a good reason to remove the outlier, such as suspicious measurements that are unlikely to be the part of real data.

1. Handling missing data

Missing data is a deceptively tricky issue in machine learning. We cannot just ignore or remove the missing observation. They must be handled carefully as they can be an indication of something important. The two most common ways to deal with missing data are:

1. Dropping observations with missing values.

Dropping missing values is sub-optimal because when you drop observations, you drop information.

* The fact that the value was missing may be informative in itself.
* Plus, in the real world, you often need to make predictions on new data even if some of the features are missing!

1. Imputing the missing values from past observations.

Imputing missing values is sub-optimal because the value was originally missing but you filled it in, which always leads to a loss in information, no matter how sophisticated your imputation method is.

* Again, “missingness” is almost always informative in itself, and you should tell your algorithm if a value was missing.
* Even if you build a model to impute your values, you’re not adding any real information. You’re just reinforcing the patterns already provided by other features.
* Both of these approaches are sub-optimal because dropping an observation means dropping information, thereby reducing data and imputing values also is sub-optimal as we fil the values that were not present in the actual dataset, which leads to a loss of information.
* Missing data is like missing a puzzle piece. If you drop it, that’s like pretending the puzzle slot isn’t there. If you impute it, that’s like trying to squeeze in a piece from somewhere else in the puzzle.
* So, missing data is always informative and indication of something important. And we must aware our algorithm of missing data by flagging it. By using this technique of flagging and filling, you are essentially allowing the algorithm to estimate the optimal constant for missingness, instead of just filling it in with the mean.

Some data cleansing tools

* + Openrefine
  + Trifacta Wrangler
  + TIBCO Clarity
  + Cloudingo
  + IBM Infosphere Quality Stage

So, we have discussed four different steps in data cleaning to make the data more reliable and to produce good results. After properly completing the Data Cleaning steps, we’ll have a robust dataset that avoids many of the most common pitfalls. This step should not be rushed as it proves very beneficial in the further process.

Feature Scaling is a technique to standardize the independent features present in the data in a fixed range. It is performed during the data pre-processing.

**Step 4:** Exploratory Data Analysis

Grab your detective glasses because this stage is all about diving deep into data and finding all the hidden data mysteries. EDA or Exploratory Data Analysis is the brainstorming stage of Machine Learning. Data Exploration involves understanding the patterns and trends in the data. At this stage, all the useful insights are drawn and correlations between the variables are understood.

For example, in the case of predicting rainfall, we know that there is a strong possibility of rain if the temperature has fallen low. Such correlations must be understood and mapped at this stage.

**Step 5:** Building a Machine Learning Model

All the insights and patterns derived during Data Exploration are used to build the Machine Learning Model. This stage always begins by splitting the data set into two parts, training data, and testing data. The training data will be used to build and analyze the model. The logic of the model is based on the Machine Learning Algorithm that is being implemented.

Choosing the right algorithm depends on the type of problem you’re trying to solve, the data set and the level of complexity of the problem. In the upcoming sections, we will discuss the different types of problems that can be solved by using Machine Learning.

**Step 6:** Model Evaluation & Optimization

After building a model by using the training data set, it is finally time to put the model to a test. The testing data set is used to check the efficiency of the model and how accurately it can predict the outcome. Once the accuracy is calculated, any further improvements in the model can be implemented

at this stage. Methods like parameter tuning and cross-validation can be used to improve the performance of the model.

**Step 7:** Predictions

Once the model is evaluated and improved, it is finally used to make predictions. The final output can be a Categorical variable (eg. True or False) or it can be a Continuous Quantity (eg. the predicted value of a stock).In our case, for predicting the occurrence of rainfall, the output will be a categorical variable.This step is also known as programming phase. The implementation of software design starts in terms of writing program code in the suitable programming language and developing error-free executable programs efficiently.

#### Deployment diagram

There may be more steps involved, depending on what specific requirements you have, but below are some of the main steps:

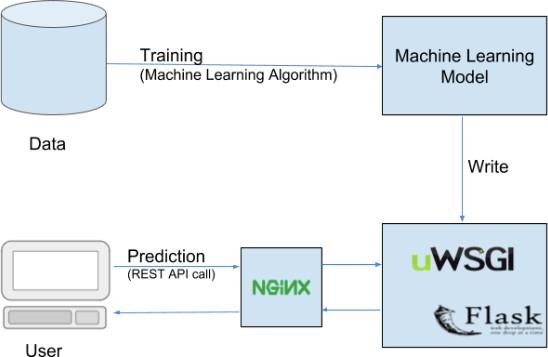


Figure 4.4.1

## Chapter 5

### Algorithms

#### Decision Tree

* Decision tree algorithm belongs to the family of supervised learning algorithms.
* The goal of using a Decision tree is to create a training model that can be used to predict the class or value of the target variable by learning simple decision rules inferred from prior data (training data).

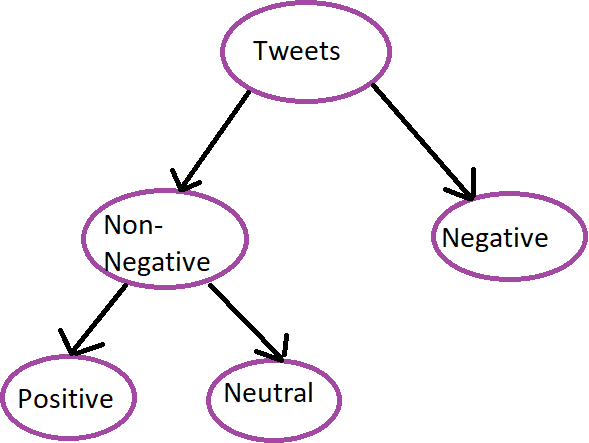


Figure 5.1.1

#### Random Forest

* Random forest algorithm is a supervised learning algorithms which is used for both classification and regression

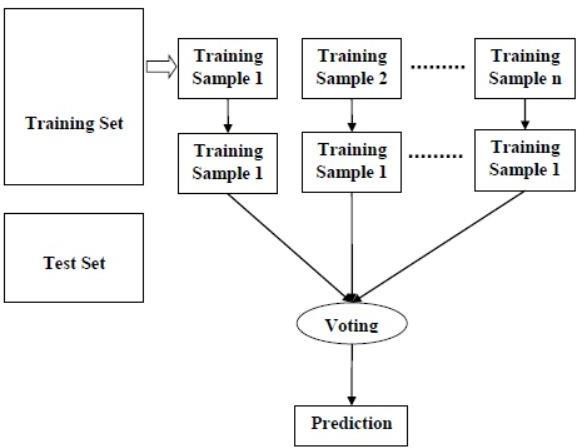


Figure 5.1.2

#### Logistic Regression

* Logistic regression is a supervised learning algorithms which is used to predict the probability of target variable
* It is one of the simplest ML algorithms that can be used for various classification problems such as spam detection, cancer detection etc.

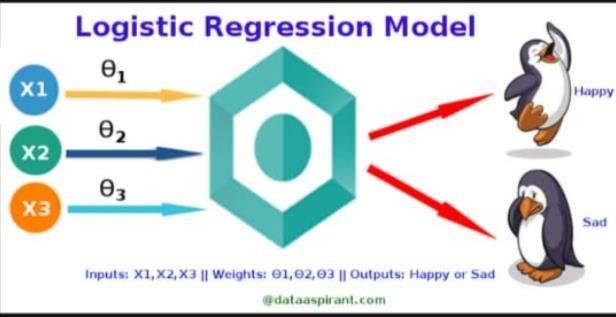


Figure 5.1.3

#### SVM

* SVM is nothing but the Support Vector Machine
* The SVM are supervised learning models with associated learning algorithms that analyze data for classification and regression analysis.

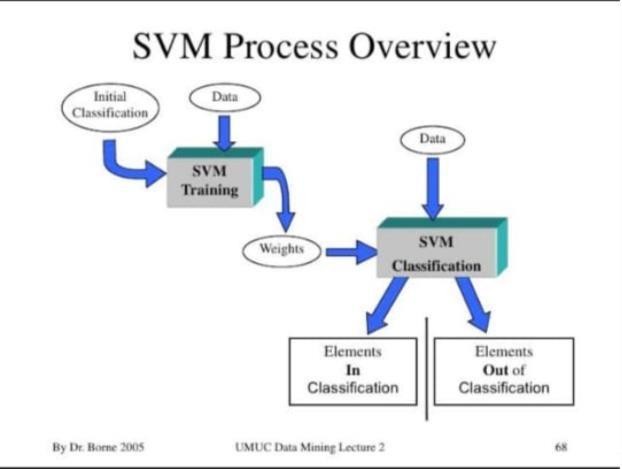


Figure 5.1.4

## Chapter 6

### Implementation

#### Installation and Execution Steps

First setup we have to setup the python environment with anaconda in windows and next install all required packages using pip command. These are the required packages for running the facial expression code. There are

1. Python 3.7.3
2. Keras 2.3.1
3. Tensorflow 1.14.0 4. Opencv 3.4.3.18

5. Scipy

After installing all required packages for facial expression recognition than move to the main code folder. Then run the python camera.py code through command prompt.

* + 1. Keras

Keras is a powerful and easy-to-use free open source Python library for developing and evaluating deep learning models. It wraps the efficient numerical computation libraries Theano and TensorFlow and allows you to define and train neural network models in just a few lines of code.

* Keras is a high-level interface and uses Theano or Tensorflow for its backend.
* It runs smoothly on both CPU and GPU.
* Keras, being modular in nature, is incredibly expressive, flexible, and apt for innovative research.
* Keras is a completely Python-based framework, which makes it easy to debug and explore. Keras with Deep Learning Frameworks:

Keras does not replace any of TensorFlow (by Google), CNTK (by Microsoft) or Theano but instead it works on top of them. Infact, Keras needs any of these backend deep-learning engines, but Keras officially recommends TensorFlow.

Keras & Python Version Compatibility:

Keras is compatible with Python2 (starting from v2.7) and Python3 (till version 3.6).

Features of Keras library

1. Keras is an user friendly API. It has consistent and simple APIs. For regular use cases, it requires very less of user effort.
2. Keras gives a very useful feedback about user actions in case of any error. It provides with the actionable feedback which helps developers to pinpoint the line or error and correct it.
3. Keras does not require separate configuration files for models. You can describe the model configuration in Python code itself.
4. Keras can run seamlessly on both CPU and GPU with required libraries installed.
5. Keras is extensible, which means you can add new modules as new classes and functions.
6. When it comes to support for development with Keras Library, Keras provides good number of examples for the existing models.
   * 1. Tensor Flow

Tensor Flow is an opensource library for numerical computation and large-scale machine learning. TensorFlow bundles together a slew of machine learning and deep learning (aka neural networking) models and algorithms and makes them useful by way of a common metaphor. It uses Python to provide a convenient front-end API for building applications with the framework, while executing those applications in high performance C++.

TensorFlow can train and run deep neural networks for handwritten digit classification, image recognition, word embeddings, recurrent neural networks, sequence-to-sequence models for machine translation, natural language processing, and PDE (partial differential equation) based simulations. Best of all, TensorFlow supports production prediction at scale, with the same models used for training. TensorFlow allows developers to create dataflow graphs structures that describe how data moves through a graph, or a series of processing nodes. Each node in the graph represents a mathematical operation, and each connection or edge between nodes is a multidimensional data array, or tensor.

TensorFlow provides all of this for the programmer by way of the Python language. Python is easy to learn and work with, and provides convenient ways to express how high-level abstractions can be coupled together. Nodes and tensors in TensorFlow are Python objects, and TensorFlow applications are themselves Python applications.

The actual math operations, however, are not performed in Python. The libraries of transformations that are available through TensorFlow are written as high-performance C++ binaries. Python just directs traffic between the pieces, and provides high-level programming abstractions to hook them together.

TensorFlow applications can be run on most any target that’s convenient: a local machine, a cluster in the cloud, iOS and Android devices, CPUs or GPUs. If you use

Google’s own cloud, you can run TensorFlow on Google’s custom TensorFlow

Processing Unit (TPU) silicon for further acceleration. The resulting models created by TensorFlow, though, can be deployed on most any device where they will be used to serve predictions.

TensorFlow 2.0, released in October 2019, revamped the framework in many ways based on user feedback, to make it easier to work with (e.g., by using the relatively simple Keras API for model training) and more performant. Distributed training is easier to run thanks to a new API, and support for TensorFlow Lite makes it possible to deploy models on a greater variety of platforms. However, code written for earlier versions of TensorFlow must be rewritten sometimes only slightly, sometimes significantly to take maximum advantage of new TensorFlow 2.0 features.

The single biggest benefit TensorFlow provides for machine learning development is abstraction. Instead of dealing with the nitty-gritty details of implementing algorithms, or figuring out proper ways to hitch the output of one function to the input of another, the developer can focus on the overall logic of the application. TensorFlow takes care of the details behind the scenes.

TensorFlow offers additional conveniences for developers who need to debug and gain introspection into TensorFlow apps. The eager execution mode lets you evaluate and modify each graph operation separately and transparently, instead of constructing the entire graph as a single opaque object and evaluating it all at once. The Tensor Board visualization suite lets you inspect and profile the way graphs run by way of an interactive, web-based dashboard.

TensorFlow also gains many advantages from the backing of an A-list commercial outfit in Google. Google has not only fueled the rapid pace of development behind the project, but created many significant offerings around TensorFlow that make it easier to deploy and easier to use: the above- mentioned TPU silicon for accelerated performance in Google’s cloud; an online hub for sharing models created with the framework; in-browser and mobile-friendly incarnations of the framework; and much more.

One caveat: Some details of TensorFlow’s implementation make it hard to obtain totally deterministic model-training results for some training jobs. Sometimes a model trained on one system will vary slightly from a model trained on another, even when they are fed the exact same data. The reasons for this are slippery e.g., how random numbers are seeded and where, or certain non-deterministic behaviors when using

GPUs). That said, it is possible to work around those issues, and TensorFlow’s team is considering more controls to affect determinism in a workflow.

* + 1. OpenCV

OpenCV is the huge open-source library for computer vision, machine learning, and image processing and now it plays a major role in real-time operation which is very important in today’s systems. By using it, one can process images and videos to identify objects, faces, or even the handwriting of a human. When it integrated with various libraries, such as Numpy, python is capable of processing the OpenCV array structure for analysis. To Identify image patterns and its various features we use vector space and perform mathematical operations on these features.

To install OpenCV, one must have Python and PIP, preinstalled on their system. **PIP** is a package management system used to install and manage software packages/libraries written in Python

* + 1. Downloading and Installing OpenCV

OpenCV can be directly downloaded and installed with the use of pip (package manager). To install OpenCV, just go to the command-line and type the following command:

pip install opencv-python

* + 1. Scipy

SciPy in Python is an open-source library used for solving mathematical, scientific, engineering, and technical problems. It allows users to manipulate the data and visualize the data using a wide range of high-level Python commands. SciPy is built on the Python NumPy extention. SciPy is also pronounced as "Sigh Pi." • SciPy is a scientific computation library that uses [NumPy](https://www.w3schools.com/python/numpy_intro.asp) underneath.

* SciPy stands for Scientific Python.
* It provides more utility functions for optimization, stats and signal processing.
* Like NumPy, SciPy is open source so we can use it freely.

#### Conclusion

In Chapter learnt about different libraries of python and its installation Steps.

## Chapter 7

### Program Code

* 1. Code.py

import numpy as np import pandas as pd import re

import nltk

import matplotlib.pyplot as plt

%matplotlib inline

airline\_tweets = pd.read\_csv(r'C:\Users\SMONIGAYATHRI\OneDrive\Desktop\New\Tweets.csv') airline\_tweets.head()

plot\_size=plt.rcParams["figure.figsize"] print(plot\_size[0])

print(plot\_size[1]) plot\_size[0]=8 plot\_size[1]=6

plt.rcParams["figure.figsize"]=plot\_size

airline\_tweets.airline.value\_counts().plot(kind='pie',autopct="%1.0f%%")

airline\_tweets.airline\_sentiment.value\_counts().plot(kind='pie', autopct='%1.0f%%', colors=["red","yellow","green"])

airline\_sentiment=airline\_tweets.groupby(['airline','airline\_sentiment']).airline\_sentiment.count().uns tack()

airline\_sentiment.plot(kind='bar') import seaborn as sns

sns.barplot(x='airline\_sentiment',y='airline\_sentiment\_confidence', data=airline\_tweets) features=airline\_tweets.iloc[:,10].values

labels=airline\_tweets.iloc[:,1].values processed\_features=[]

for sentence in range(0, len(features)): processed\_feature=re.sub(r'\W', ' ', str(features[sentence])) processed\_feature=re.sub(r'\s+[a-zA-Z]\s+', ' ', processed\_feature) processed\_feature=re.sub(r'\^[a-zA-Z]\s+', ' ', processed\_feature) processed\_feature=re.sub(r'\s+', ' ', processed\_feature,flags=re.I) processed\_feature=re.sub(r'^b\s+', '', processed\_feature) processed\_feature=processed\_feature.lower() processed\_features.append(processed\_feature)

import warnings warnings.filterwarnings("ignore")

print(airline\_tweets.negativereason.value\_counts()) plt.figure(figsize=(25,5)) sns.countplot(airline\_tweets.negativereason)

import warnings warnings.filterwarnings("ignore") import nltk

from wordcloud import WordCloud,STOPWORDS from nltk.corpus import stopwords

from nltk.stem import SnowballStemmer import re

nltk.download('stopwords')

stop\_words = stopwords.words('english')

new\_data=airline\_tweets[airline\_tweets['airline\_sentiment']=='negative'] words = ' '.join(new\_data['text'])

cleaned\_word=' '.join([word for word in words.split() if 'http' not in word

and not word.startswith('@') and word !='RT'])

wordcloud = WordCloud(stopwords=STOPWORDS, background\_color='black', width=3000,

height=2500

).generate(cleaned\_word) plt.figure(1,figsize=(10,10)) plt.imshow(wordcloud)

plt.axis('off') plt.show()

new\_data=airline\_tweets[airline\_tweets['airline\_sentiment']=='positive'] words = ' '.join(new\_data['text'])

cleaned\_word=' '.join([word for word in words.split()

if 'http' not in word

and not word.startswith('@') and word !='RT'])

wordcloud = WordCloud(stopwords=STOPWORDS, background\_color='black', width=3000,

height=2500

).generate(cleaned\_word) plt.figure(1,figsize=(10,10)) plt.imshow(wordcloud)

plt.axis('off') plt.show()

neutral\_text=airline\_tweets[airline\_tweets['airline\_sentiment']=='neutral'] words=''.join(neutral\_text['text'])

cleaned\_word=' '.join([word for word in words.split() if 'http' not in word

and not word.startswith('@') and word!='RT'])

wordcloud=WordCloud(stopwords=STOPWORDS, background\_color='black', width=3000, height=2500).generate(cleaned\_word)

plt.figure(1,figsize=(10,10)) plt.imshow(wordcloud)

plt.axis('off') plt.show()

text\_cleaning\_re = "@\S+|https?:\S+|http?:\S|[^A-Za-z0-9]+" def preprocess(x,stem=False):

x=re.sub(text\_cleaning\_re,' ',str(x).lower()).strip() tokens=[]

for token in x.split('\n'):

if token not in stop\_words: if stem:

tokens.append(stemmer.stem(token)) else:

tokens.append(token) return ' '.join(tokens)

airline\_tweets.text=airline\_tweets.text.apply(lambda x:preprocess(x))

airline\_tweets['sentiment']=airline\_tweets['airline\_sentiment'].apply(lambda x: 0 if x=='negative' else 1)

print(airline\_tweets.sentiment.value\_counts()) y=airline\_tweets.sentiment

from sklearn.feature\_extraction.text import TfidfVectorizer vectorizer = TfidfVectorizer(min\_df=10)

X= vectorizer.fit\_transform(airline\_tweets.text) from sklearn.model\_selection import train\_test\_split

X\_train,X\_test,y\_train,y\_test = train\_test\_split(X,y,test\_size=0.2, random\_state=0) import sklearn

model = sklearn.linear\_model.LogisticRegression(penalty="l2", C=0.1)

model.fit(X\_train, y\_train)

score = model.score(X\_test, y\_test) score

from sklearn.metrics import classification\_report, confusion\_matrix sns.heatmap(confusion\_matrix(y\_test, model.predict(X\_test))) print(classification\_report(y\_test, model.predict(X\_test)))

from sklearn.ensemble import RandomForestClassifier models=RandomForestClassifier()

models.fit(X\_train, y\_train)

score = models.score(X\_test, y\_test) score

from sklearn.metrics import classification\_report, confusion\_matrix print(classification\_report(y\_test, model.predict(X\_test))) sns.heatmap(confusion\_matrix(y\_test, model.predict(X\_test))) from sklearn.svm import SVC

svm=SVC()

svm=SVC(kernel="rbf", C=0.025, probability=True) svm.fit(X\_train,y\_train)

score = svm.score(X\_test, y\_test) score

from sklearn.metrics import classification\_report, confusion\_matrix print(classification\_report(y\_test, model.predict(X\_test))) sns.heatmap(confusion\_matrix(y\_test, model.predict(X\_test))) from sklearn.tree import DecisionTreeClassifier

model=DecisionTreeClassifier(criterion="gini") model.fit(X\_train,y\_train)

score = model.score(X\_test, y\_test) score

from sklearn.metrics import classification\_report, confusion\_matrix print(classification\_report(y\_test, model.predict(X\_test))) sns.heatmap(confusion\_matrix(y\_test, model.predict(X\_test))) model.fit(X\_train,y\_train,epochs=20,batch\_size=32,verbose = 5) model.save("sentiment.h5")

#### App.py

from flask import render\_template, Flask, request,url\_for #from werkzeug.utils import secure\_filename

#from gevent.pywsgi import WSGIServer from keras.models import load\_model import pickle

import tensorflow as tf

graph = tf.get\_default\_graph()

with open(r'CountVectorizer','rb') as file: cv=pickle.load(file)

cla = load\_model('sentiment.h5')

app = Flask( name ) @app.route('/')

def index():

return render\_template('index.html')

@app.route('/tpredict')

@app.route('/', methods = ['GET','POST'])

def page2():

if request.method == 'GET':

img\_url = url\_for('static',filename = 'style/3.jpg') return render\_template('index.html',url=img\_url)

if request.method == 'POST': topic = request.form['tweet'] print("Hey " +topic) topic=cv.transform([topic]) print("\n"+str(topic.shape)+"\n") with graph.as\_default():

y\_pred = cla.predict\_classes(topic) print("pred is "+str(y\_pred))

if(y\_pred[0] == 2):

img\_url = url\_for('static',filename = 'style/1.jpg') topic = "Positive Tweet"

elif(y\_pred[0] == 0):

img\_url = url\_for('static',filename = 'style/2.jpg') topic = "Negative Tweet"

else:

img\_url = url\_for('static',filename = 'style/3.jpg') print(img\_url)

topic = "Neutral Tweet"

return render\_template('index.html',ypred = topic)

if name == ' main ':

app.run(host = 'localhost', debug = True , threaded = False)

#### Html code

<html>

<head>

<title>{{ title }} Sentiment Analysis</title>

<style> html {

background: #404550;

}

body {

background-color:white; background-

image:url("https://miro.medium.com/proxy/1\*NF6AdPk6sOMNNbQE5glvEQ.png"); background-repeat:no-repeat;

background-position:center bottom;

}

form, div{

border-bottom: 2px solid rgb(76, 67, 65); margin-bottom: 5em;

margin-left: auto; margin-right: auto; width: 50em

}

h1 {

font-family: Georgia, Times, "Times New Roman", serif; font-size: 1.8em;

color:red;

text-align:center;

border-bottom: 2px solid rgb(76, 67, 65); margin-bottom: 1.5em;

background: url(../\_images/icon\_sprites\_50.png) no-repeat

}

.example\_a { border: none;

background: #404040; color: #ffffff !important; font-weight: 100; padding: 20px;

text-transform: uppercase; border-radius: 6px; display: inline-block; transition: all 0.3s ease 0s;

float: right; margin-top: 2em;

}

.example\_a:hover {

color: #404040 !important;

font-weight: 700 !important; letter-spacing: 3px; background: none;

-webkit-box-shadow: 0px 5px 40px -10px rgba(0,0,0,0.57);

-moz-box-shadow: 0px 5px 40px -10px rgba(0,0,0,0.57); transition: all 0.3s ease 0s;

}

</style>

</head>

<body>

<h1>Opinion Mining on AirLine Reviews</h1>

<form method="POST">

<textarea name="text1" placeholder="Type your Tweet here: " rows="5" cols="50"></textarea><br><br>

<input class="example\_a" type="submit">

</form>

{% if final %}

<div>

<h2>The Sentiment of</h2> '{{ text1 }}' <h2>is {{ final }}% positive !</h2>

{% else %}

<p></p>

{% endif %}

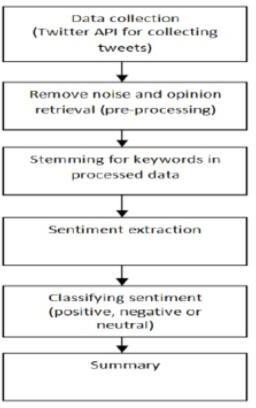
</div>

</body>

</html>

## Chapter 8

Flow Chart

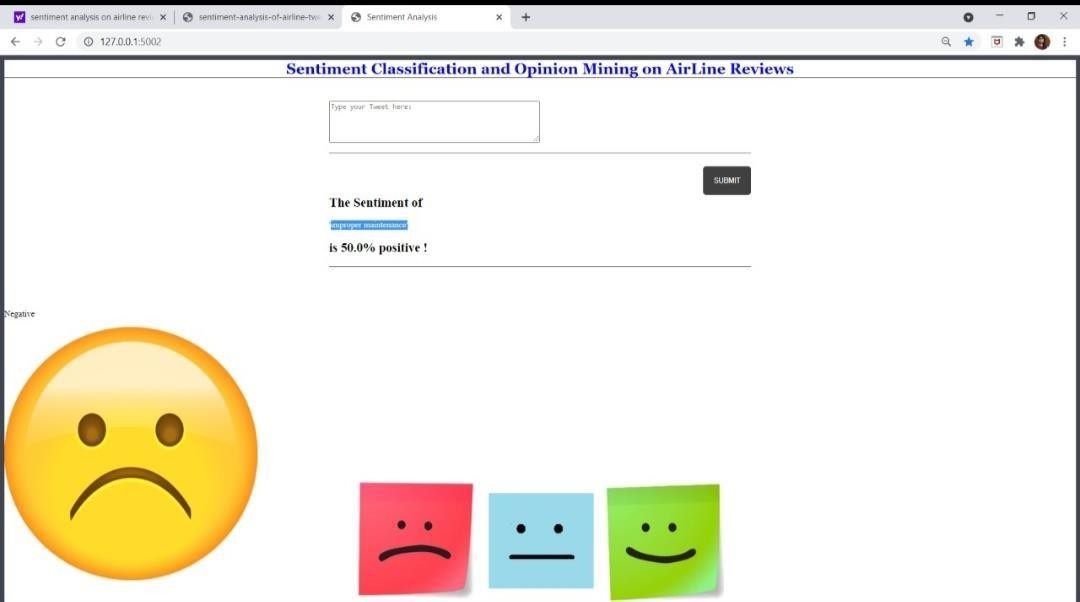


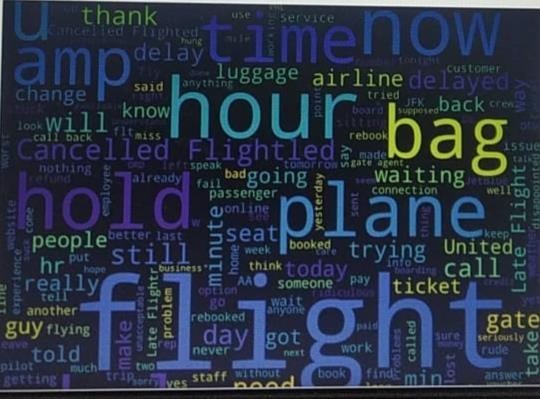
## Chapter 9

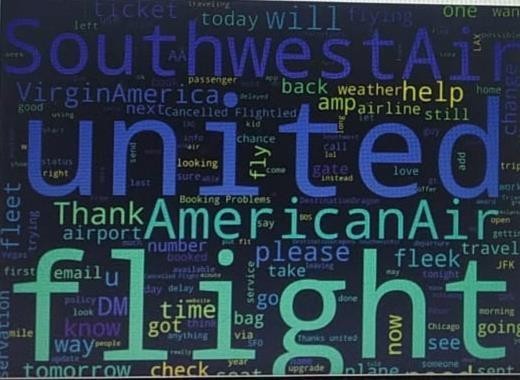
### Output



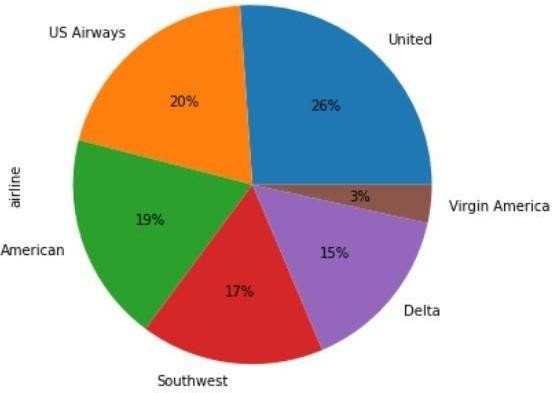


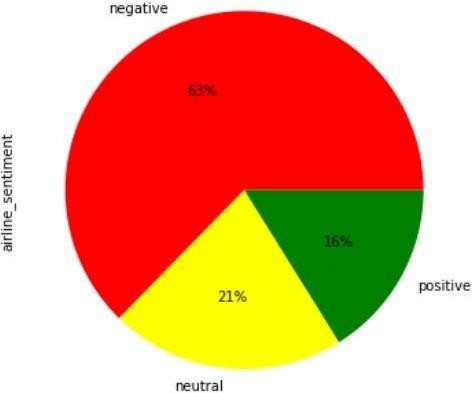


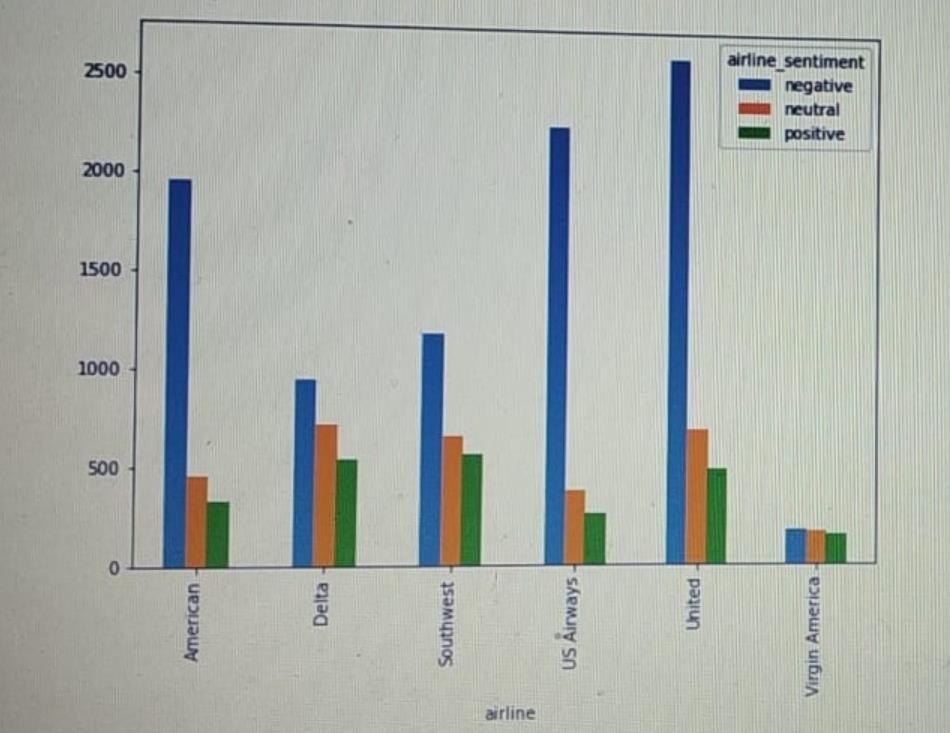


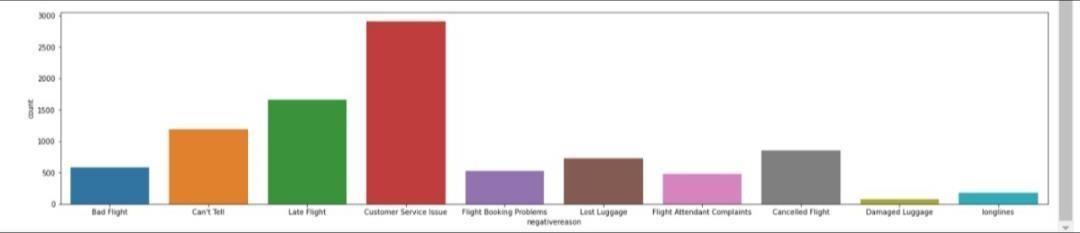












Advantages

## Chapter 10

### Advantages and Disadvantages

1. Gets Sentiments from the passengers as negative, neutral, positive. Negative - totally dissatisfied with the service, worst customer care ever. Neutral - good and satisfactory by the service.

Positive - brilliant and excellent job by the service.

1. Easy to access.
2. Can know the quality of airline by the sentiments.
3. Very faster way of understanding people’s feedback.

#### Disadvantages

1. Lack of ability to be spatially invariant to the input data.
2. Cannot learn temporal dependence.
3. Predicted values are not are always accurate, it exists in the ration of 7:10.

## Chapter 11

### Applications

#### APPLICATIONS

1. Sentiment analysis offers organizations the ability to monitor various social media sites in real time and act accordingly.
2. Aspect-level sentiment analysis is the most fine-grained analysis of review articles and social media snippets with respect to specific objects and their aspects.
3. Utilization of sentiment analysis techniques in stock picking and lead to superior returns.
4. Various business applications are Consumer voice, brand reputation, online commerce.
5. Politic applications are voting advice application.
6. Used in airlines for analysing reviews.

## Chapter 12

### Conclusion and Future Scope

#### Conclusion

Sentimental Analysis is a latest trend to understand the needs of the mass public; it’s an easier and cost effective way to understand how the people are feeling about a particular subject of matter and the brand impact of micro-blogging [3]. In this scenario we had consider the sentiment of the people towards the airline industry and tackled the recent issues of United Airlines and how the public feels about it.

The analysis confirmed our assumption on how effective an approach twitter sentiment analysis is. The procedure that we used, along with the software for better results depict clearly the sentiment of the mass crowd and thus the airlines could easily interpret the data and benefit from it by trying to improve on the aspects that seem negative or is disliked by the targeted audience.

#### Future Scope

There is still scope for improvement in this analysis since it is very new and yet has not been tested on many other classifying models. And the major setback is the limit in the number of tweets to be analyzed using AYLIEN in Rapid Miner being 1000 tweets a day for a free user otherwise one has to opt for plans.So in the future we are planning to further expand our research and analysis by gather a huge number of data and expanding the process of data mining involved in this analytical approach.

## Chapter 13

### References

#### References

1. B. J. Jansen, M. Zhang, K. Sobel, and A. Chowdary. Micro-blogging as online word of mouth branding. In CHI EA ’09: Proceedings of the 27th international conference extended abstracts on Human factors in computing systems, pages 3859–3864, New York, NY, USA, 2009. ACM.
2. T. Joachims. Making large-scale support vector machine learning practical. In B. Scholkopf,

C. J. C. Burges, and A. J. Smola, editors, Advances in kernel methods: support vector learning, pages 169–184. MIT Press, Cambridge, MA, USA, 1999.

1. C. D. Manning and H. Schutze. Foundations of statistical natural language processing. MIT Press, 1999.
2. Kouloumpis, Efthymios, Theresa Wilson, and Johanna D. Moore. "Twitter sentiment analysis: The good the bad and the omg!." Icwsm 11.538-541 (2011): 164.
3. M. Taboada, J. Brooke, M. Tofiloski, K. Voll, and M. Stede, “Lexicon-based methods for sentiment analysis,” Comput. Linguist., vol. 37, no. 2, pp. 267-307, 2011.
4. T. Mikolov, K. C. and, G. Corrado, and J. Dean, “Efficient Estimation of Word Representations in Vector Space,” CoRR, vol. abs/1301.3781, 2013.
5. B. Liu and L. Zhang, “A Survey of Opinion Mining and Sentiment Analysis,” in Mining Text Data, C. C. Aggarwal and C. Zhai, Eds. Boston, MA: Springer US, 2012, pp. 415-463. [8] B. Pang and

L. Lee, “Opinion Mining and Sentiment Analysis,” Found. Trends Inf. Retr., vol. 2, no. 1-2, pp. 1- 135, 2008.

[9] S. O. Orimaye, S. M. Alhashmi, and E.-G. Siew, “Performance and trends in recent opinion retrieval techniques,” The Knowledge Engineering Review, vol. 30, no. 1, pp. 76-105, 2015/001/001 2015