Automatic Multi-Class Document Classification using AI Techniques

Graduation Project

Project report submitted in partial fulfillment of the requirement for the award of the Degree of Bachelor of Science in Computer Science

By

Ali A. Alrabeei 351105820

Ahmed O. Alomar 362105145

Under the Supervision of

Dr. Ahmed Abo-Bakr

Department of Computer Science

College of Computer and Information Sciences

Majmaah University

Al Majmaah

Kingdom of Saudi Arabia

(S2) 2019

Automatic Multi-Class Document Classification using AI Techniques

Graduation Project

Project report submitted in partial fulfillment of the Requirements for the Award of the Degree of BSc. In Computer Science

By

Ahmed O. Alomar

362105145

alomar1432@gmail.com

Ali A. Alrabeei

351105820

newaaa4@gmail.com

Department of Computer Science

College of Computer and Information Sciences

Majmaah University

Al Majmaah, Kingdom of Saudi Arabia

CERTIFICATE

This is to certify that the project report entitled Automatic Multi-Class Document Classification using AI Techniques being submitted by

Ahmed O. Alomar 362105145
Ali A. Alrabeei 351105820

in partial fulfillment for the award of the Degree of Bachelor of Science in Computer Science to the Majmaah University, AL Majmaah, Kingdom of Saudi Arabia during the academic year 2018/2019 is a record of authentic work carried out under my supervision and guidance and it has not formed the basis for the award of any Degree/Diploma/Associateship/Fellowship or other similar title to any candidate of any University

Signature of Supervisor
Dr. Ahmed Abo-Bakr

Countersigned by

Head of the Department

Seal of the College

DECLARATION

We, Ahmed Alomar & Ali Alrabeei, declare that the project report entitled Automatic Multi-Class Document Classification using AI Techniques submitted to Majmaah University, Al Majmaah, Kingdom of Saudi Arabia, in partial fulfillment for the award of the Degree of Bachelor of Science in Computer Science during the academic year 2018/2019 is a record of authentic work carried out under the supervision and guidance of Dr. Ahmed Abo-Bakr and it has not formed the basis for the award of any Degree/Diploma/Associateship/Fellowship or other similar title to any candidate of any University

Signature of Students

Ali Alrabeei

Ahmed Alomar

ACKNOWLEDGEMENT

The team would like to express their gratitude and appreciation to College of Computer and Information Sciences, Majmaah University for providing the permission and various hardware/software resources for completing the project successful.

Special thanks to our honorable rector **Dr. Khalid Bin Saad Al Muqrin**, Majmaah University for his constant help and for providing us with all facilities needed for this project work

Our sincere gratitude to **Dr. Mohammed Al Shehri,** Dean, College of Computer and Information Sciences, Majmaah University for supporting and encouraging the project work in every phase.

Sincere thanks to **Dr. Talal Alharbi, Vice Dean (Academic Affairs)** College of Computer and Information Sciences, Majmaah University for his continued support in completing this project successfully.

Our special thanks to our supervisor **Dr. Ahmed Abo-Bakr**, College of Computer and Information Sciences, Majmaah University. His inspiring guidance and valuable advice and continuous encouragement at every stage in the progress of this project work enabled us to complete the project on time

ABSTRACT

While the documents are increasing every day in the world, the classification become an important subject for those who have unclassified documents. To classify documents using machine learning there is many methods. In this report you will see a few methods that's we are worked on, like Support Vector Machine, K-Nearest Neighbor, Naïve Bayes, etc.

Each method has a different way to apply it and different accuracy depend on the data and the preprocessing of the data.

At the end of this project we will see which method is better and give high accuracy to apply in our website.

ACRONYMS AND KEY TERMINOLOGIES

Supervised learning:	the machine learning task of inferring a function from labeled training data
Classifier:	mathematical function, implemented by a classification algorithm that maps input data to a category
SVM:	Support Vector Machine
NB:	Naïve Bayes
K-NN:	K-Nearest Neighbors
RF:	Random Forest
Python:	a general-purpose, high-level programming language
Sckit-Learn:	an open source machine learning library for the Python programming language
NumPy:	the fundamental package for scientific computing with Python
SciPy:	an open source library of scientific tools
Matplotlib:	a python plotting library which produces publication quality figures
Precision:	the ratio of correct assignments by the system divided by the total number of the system's assignments. (Intuitively, precision is the ability of the classifier not to label as positive a sample that is negative)
RBF	Radial Basis Function
TF-iDF	Term Frequency inverse Document Frequency
MVC	Model-View-Controller
Flask	Python framework for building MVC projects
HTML	Hyper Text Markup Language for creating Web pages
CSS	Cascading Style Sheets is a style sheet language
JavaScript	High-level, interpreted programming language

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. EXECUTIVE SUMMARY

1.1 Project Overview

In the ’80s, the most popular approach for the creation of automatic document classifiers consisted in manually building, by means of knowledge engineering techniques. Such an expert system would typically consist of a set of manually defined logical rules, one per category, of type.

If (DNF formula) then (category).

A DNF (“disjunctive normal form”) formula is a disjunction of conjunctive clauses; the document is classified under (category) if it satisfies the formula, that is, if it satisfies at least one of the clauses.

The automated categorization (or classification) of texts into predefined categories has witnessed a booming interest in the last 20 years, due to the increased availability of documents in digital form and the ensuing need to organize them. In the research community, the dominant approach to this problem is based on machine learning techniques: a general inductive process automatically builds a classifier by learning, from a set of pre-classified documents, the characteristics of the categories. The advantages of this approach over the knowledge engineering approach (consisting in the manual definition of a classifier by domain experts) are a very good effectiveness, considerable savings in terms of expert labor power, and straightforward portability to different domains.

A lot of works had been studying classification of English and Latin texts very early. They studied it using different methods to see which one will fit and is this method will give high accuracy or not. So, they give a hint for new researchers to focus on a few methods that can help them.

Supervised learning

The supervised learning is where you have input data and output data and you use an algorithm to map from the input data to the output data.

The goal of Supervised learning is when you have new input data you can predict the output from the data. The name supervised learning comes from the following process. The learning algorithm is trained on a dataset can be thought as a teacher supervising the process.

Unsupervised learning

The unsupervised learning is when you have only the input data without the suitable output data.

The goal of unsupervised learning is to divide the import data based on structure or distribution of the data in order to learn from them. It was named unsupervised because there is no correct answer or teacher to guide the process.

1.2 Purpose and Scope of The Project

1.2.1 Purpose

The goal of this project is to develop a software that allows the average users to classify their own documents into different categories.

The documents classification software will help the users to select their document sand apply support vector machine classification on their documents

1.2.2 Scope

The text document classification software will allow the user to classify text documents into different categories depending on their pre-specification using machine learning methods, techniques and algorithms

1.2.3 SWOT Analysis

Strengths

- Using machine learning methods will allow the user to classify the text documents into different categories using pre-classified data for learning

Weaknesses

- Accuracy may vary depending on the documents, training data, number of features, features chosen and features representations
- Low number of training data will significantly affect the accuracy

Opportunities

- Allow users to classify documents to different categories depending on their pre-specifications with high accuracy using machine learning methods

Threats

- Wrong choose of method or features my give very bad accuracy
- The results directly deepened on training data

1.2.4 Scope statement:

prepared by Ali A. Alrabeei

Project scope statement Date: Oct 2, 2018

General Project information

Project Name:	Automatic Multi-Class Document Classification using AI Techniques
Sponsor:	Majmaah University
Project Manager:	Ahmed O. Alomar

Problem/Opportunity Statement:

The growth in the number of documents is enormous, with it comes the need to classify them to deferent categories depend on user categories.

Project Objectives:

Make a classification software for classifying documents into categories that are specified by the user.

Project Description:

Developing a software with graphical user interface to allow users to classify their documents.

Project Deliverables:

Text Documents Classification Software Using Machine Learning Algorithms.

Estimated Project Duration:

7 months

1.2 Feasibility study

1.3.1 Technical feasibility

The project is possible with the current technology; using python programming language version 3.6 or above. The technical risk is in the future the technology can be improve and no such system is using python programs. The project is not available, it can be obtained.

1.3.2 Economic feasibility

The development cost is 0 SAR.

1.3.3 Schedule feasibility

We are able to do the project in 6 months which is less than two semesters.

1.3.4 Operational feasibility

There is a similar software in the market and there is no social issue with that software. So, the software has no social issues and the we can labor on the software.

1.3.5 Legal feasibility

The software is legal because it does not break any terms or conditions of any law system.

2. PROJECT MANAGEMENT AND PLANNING

2.1 Work Breakdown Structure (WBS)

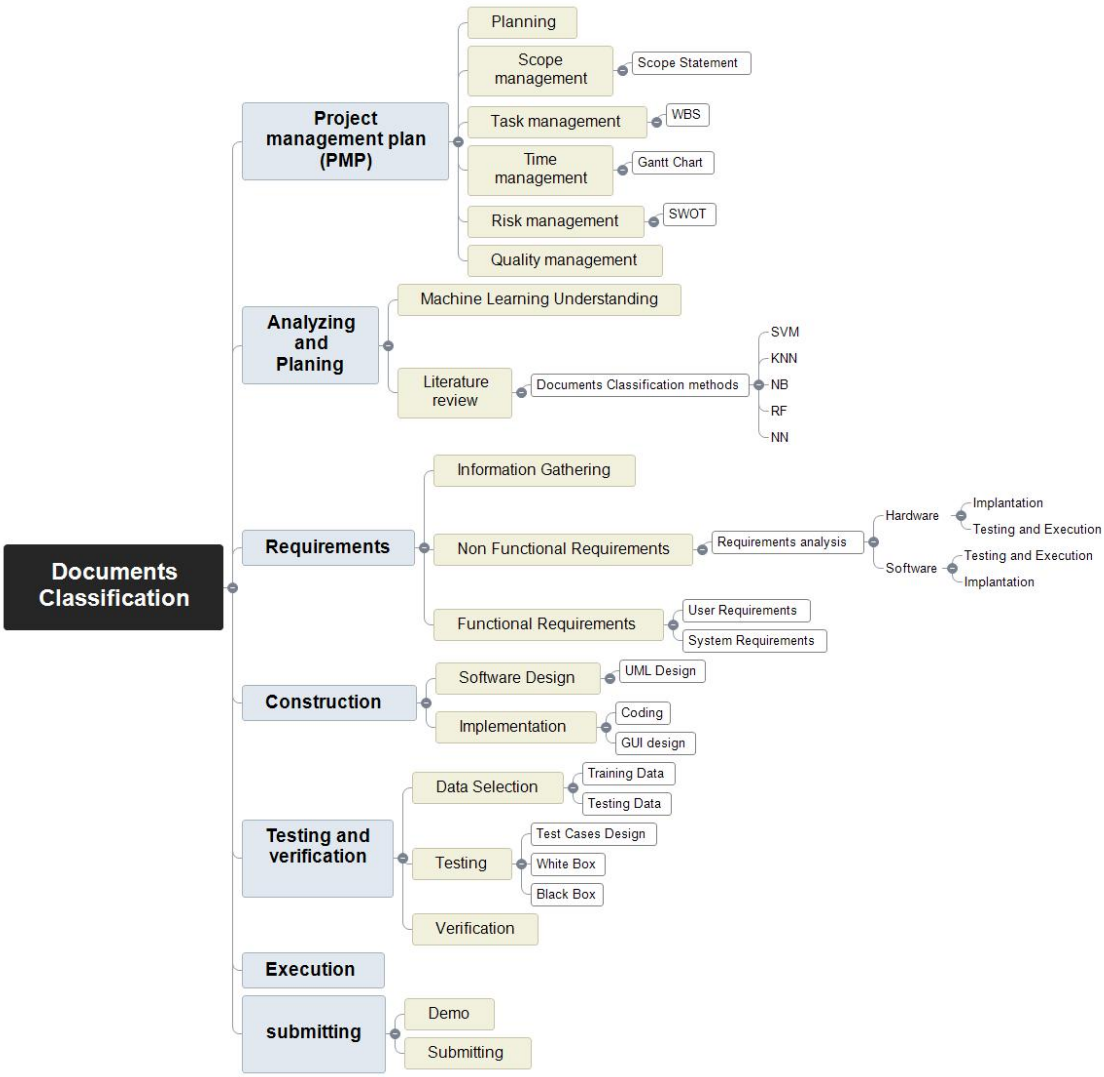
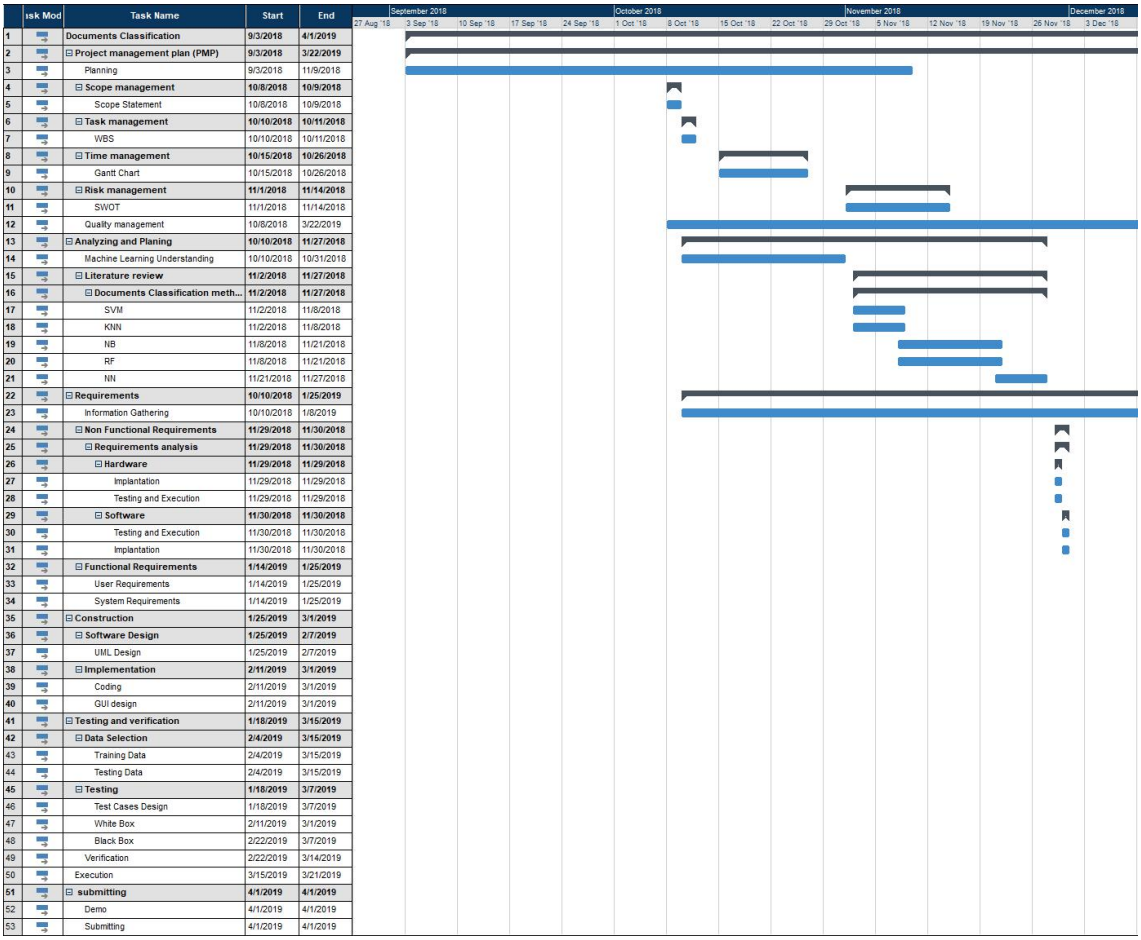


Figure 1 WBS

2.2 Gantt Charts

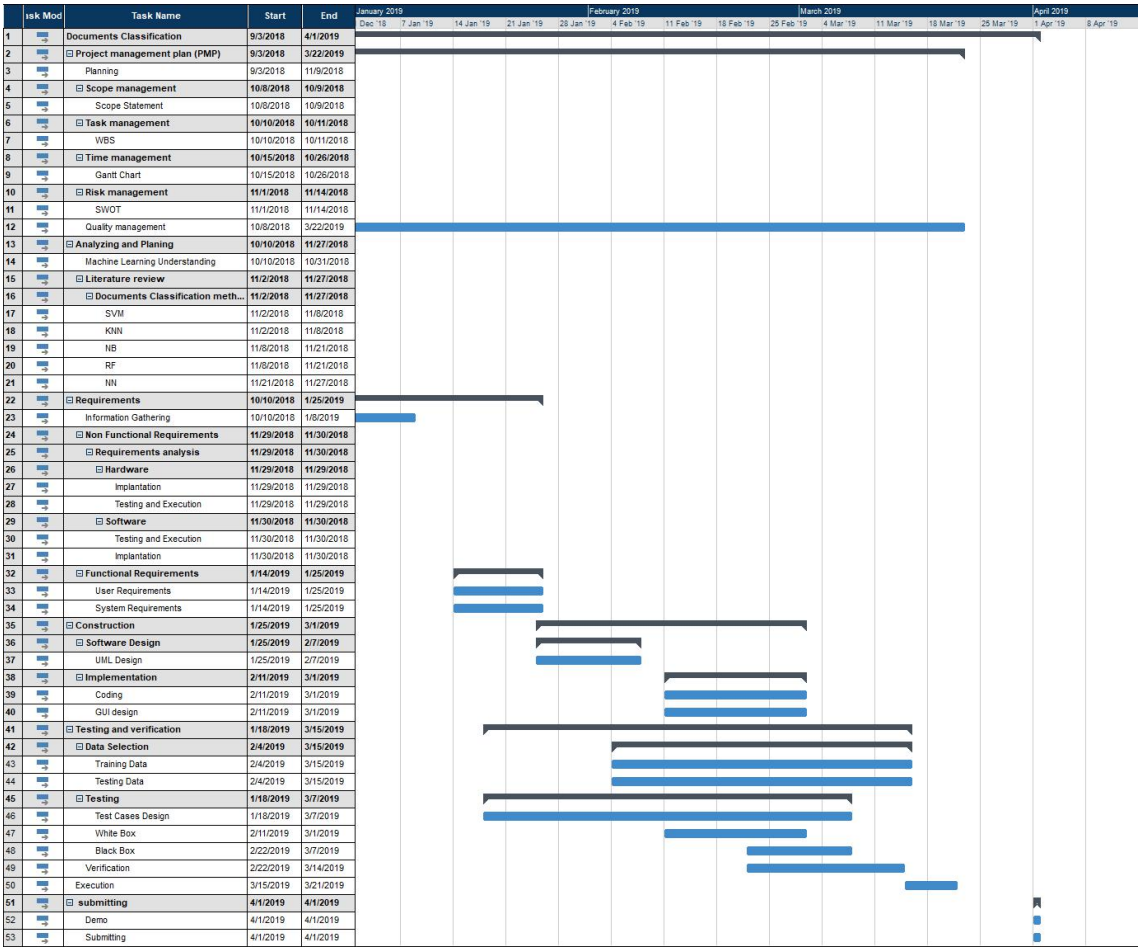
2.2.1 Gantt Chart (S1)



	Task Mod	Task Name	Start	End
1		Documents Classification	9/3/2018	4/1/2019
2		Project management plan (PMP)	9/3/2018	3/22/2019
3		Planning	9/3/2018	11/9/2018
4		Scope management	10/8/2018	10/9/2018
5		Scope Statement	10/8/2018	10/9/2018
6		Task management	10/10/2018	10/11/2018
7		WBS	10/10/2018	10/11/2018
8		Time management	10/15/2018	10/26/2018
9		Gantt Chart	10/15/2018	10/26/2018
10		Risk management	11/1/2018	11/14/2018
11		SWOT	11/1/2018	11/14/2018
12		Quality management	10/8/2018	3/22/2019
13		Analyzing and Planing	10/10/2018	11/27/2018
14		Machine Learning Understanding	10/10/2018	10/31/2018
15		Literature review	11/2/2018	11/27/2018
16		Documents Classification meth...	11/2/2018	11/27/2018
17		SVM	11/2/2018	11/8/2018
18		KNN	11/2/2018	11/8/2018
19		NB	11/8/2018	11/21/2018
20		RF	11/8/2018	11/21/2018
21		NN	11/21/2018	11/27/2018
22		Requirements	10/10/2018	1/25/2019
23		Information Gathering	10/10/2018	1/8/2019
24		Non Functional Requirements	11/29/2018	11/30/2018
25		Requirements analysis	11/29/2018	11/30/2018
26		Hardware	11/29/2018	11/29/2018
27		Implantation	11/29/2018	11/29/2018
28		Testing and Execution	11/29/2018	11/29/2018
29		Software	11/30/2018	11/30/2018
30		Testing and Execution	11/30/2018	11/30/2018
31		Implantation	11/30/2018	11/30/2018
32		Functional Requirements	1/14/2019	1/25/2019
33		User Requirements	1/14/2019	1/25/2019
34		System Requirements	1/14/2019	1/25/2019
35		Construction	1/25/2019	3/1/2019
36		Software Design	1/25/2019	2/7/2019
37		UML Design	1/25/2019	2/7/2019
38		Implementation	2/11/2019	3/1/2019
39		Coding	2/11/2019	3/1/2019
40		GUI design	2/11/2019	3/1/2019
41		Testing and verification	1/18/2019	3/15/2019
42		Data Selection	2/4/2019	3/15/2019
43		Training Data	2/4/2019	3/15/2019
44		Testing Data	2/4/2019	3/15/2019
45		Testing	1/18/2019	3/7/2019
46		Test Cases Design	1/18/2019	3/7/2019
47		White Box	2/11/2019	3/1/2019
48		Black Box	2/22/2019	3/7/2019
49		Verification	2/22/2019	3/14/2019
50		Execution	3/15/2019	3/21/2019
51		submitting	4/1/2019	4/1/2019
52		Demo	4/1/2019	4/1/2019
53		Submitting	4/1/2019	4/1/2019

Figure 2 Gantt Chart (S1)

2.2.2 Gantt Chart (S2)



	Task Mod	Task Name	Start	End
1		**Documents Classification**	**9/3/2018**	**4/1/2019**
2		**Project management plan (PMP)**	**9/3/2018**	**3/22/2019**
3		Planning	9/3/2018	11/9/2018
4		**Scope management**	**10/8/2018**	**10/9/2018**
5		Scope Statement	10/8/2018	10/9/2018
6		**Task management**	**10/10/2018**	**10/11/2018**
7		WBS	10/10/2018	10/11/2018
8		**Time management**	**10/15/2018**	**10/26/2018**
9		Gantt Chart	10/15/2018	10/26/2018
10		**Risk management**	**11/1/2018**	**11/14/2018**
11		SWOT	11/1/2018	11/14/2018
12		Quality management	10/8/2018	3/22/2019
13		**Analyzing and Planing**	**10/10/2018**	**11/27/2018**
14		Machine Learning Understanding	10/10/2018	10/31/2018
15		**Literature review**	**11/2/2018**	**11/27/2018**
16		**Documents Classification meth...**	**11/2/2018**	**11/27/2018**
17		SVM	11/2/2018	11/8/2018
18		KNN	11/2/2018	11/8/2018
19		NB	11/8/2018	11/21/2018
20		RF	11/8/2018	11/21/2018
21		NN	11/21/2018	11/27/2018
22		**Requirements**	**10/10/2018**	**1/25/2019**
23		Information Gathering	10/10/2018	1/8/2019
24		**Non Functional Requirements**	**11/29/2018**	**11/30/2018**
25		**Requirements analysis**	**11/29/2018**	**11/30/2018**
26		**Hardware**	**11/29/2018**	**11/29/2018**
27		Implantation	11/29/2018	11/29/2018
28		Testing and Execution	11/29/2018	11/29/2018
29		**Software**	**11/30/2018**	**11/30/2018**
30		Testing and Execution	11/30/2018	11/30/2018
31		Implantation	11/30/2018	11/30/2018
32		**Functional Requirements**	**1/14/2019**	**1/25/2019**
33		User Requirements	1/14/2019	1/25/2019
34		System Requirements	1/14/2019	1/25/2019
35		**Construction**	**1/25/2019**	**3/1/2019**
36		**Software Design**	**1/25/2019**	**2/7/2019**
37		UML Design	1/25/2019	2/7/2019
38		**Implementation**	**2/11/2019**	**3/1/2019**
39		Coding	2/11/2019	3/1/2019
40		GUI design	2/11/2019	3/1/2019
41		**Testing and verification**	**1/18/2019**	**3/15/2019**
42		**Data Selection**	**2/4/2019**	**3/15/2019**
43		Training Data	2/4/2019	3/15/2019
44		Testing Data	2/4/2019	3/15/2019
45		**Testing**	**1/18/2019**	**3/7/2019**
46		Test Cases Design	1/18/2019	3/7/2019
47		White Box	2/11/2019	3/1/2019
48		Black Box	2/22/2019	3/7/2019
49		Verification	2/22/2019	3/14/2019
50		Execution	3/15/2019	3/21/2019
51		**submitting**	**4/1/2019**	**4/1/2019**
52		Demo	4/1/2019	4/1/2019
53		Submitting	4/1/2019	4/1/2019

Figure 3 Gantt Chart (S2)

2.2.3 Gantt Chart (Full Project)

Task Name	Start	End
Documents Classification	9/3/2018	4/1/2019
Project management plan (PMP)	9/3/2018	3/22/2019
Planning	9/3/2018	11/6/2018
Scope management	10/8/2018	10/9/2018
Scope Statement	10/8/2018	10/8/2018
Task management	10/10/2018	10/11/2018
WBS	10/10/2018	10/11/2018
Time management	10/15/2018	10/26/2018
Gantt Chart	10/15/2018	10/26/2018
Risk management	11/1/2018	11/14/2018
SWOT	11/1/2018	11/14/2018
Quality mangement	10/8/2018	3/22/2019
Analysing and Planing	10/10/2018	11/27/2018
Machine Learning Understanding	10/10/2018	10/31/2018
Literature review	11/2/2018	11/27/2018
Documents Classification meth...	11/2/2018	11/8/2018
SVM	11/2/2018	11/8/2018
KNN	11/5/2018	11/9/2018
NB	11/8/2018	11/21/2018
RF	11/8/2018	11/21/2018
NN	11/21/2018	11/27/2018
Requirements	10/10/2018	1/25/2019
Information Cathering	10/10/2018	1/8/2019
Functional Requirements	1/14/2019	1/25/2019
User Requirements	1/14/2019	1/25/2019
System Requirements	1/14/2019	1/25/2019
Construction	1/25/2019	3/1/2019
Software Design	1/25/2019	2/7/2019
UML Design	1/25/2019	2/7/2019
Implementation	2/11/2019	3/1/2019
Coding	2/11/2019	3/1/2019
GUI design	2/11/2019	3/1/2019
Testing and verification	1/18/2019	3/15/2019
Data Selection	2/4/2019	3/15/2019
Training Data	2/4/2019	3/15/2019
Testing Data	2/4/2019	3/15/2019
Testing	1/18/2019	3/7/2019
Test Cases Design	1/18/2019	3/7/2019
White Box	2/11/2019	3/1/2019
Black Box	2/22/2019	3/7/2019
Verification	2/22/2019	3/14/2019
Execution	3/15/2019	3/21/2019

Figure 4 Gantt Chart (Full Project)

3. REQUIREMENTS

3.1 Hardware and Software Requirements

3.1.1 Hardware

- **Implementation**

  i86 or amd64 one core 2.5GHz processor or higher

  2GB DDR3 RAM 1333Hz or more

  50GB HDD or more

- **Testing and execution**

  i86 or amd64 2 core 3GHz processor or higher

  8GB DDR3 RAM 1333Hz or more

  50GB HDD or more

3.1.1 Software

- **Implementation**

  Python 3.6 on higher

  pip

  python3-numpy

  python3-panda

  python3-matplotlib

  python3-scikit

  python3-gtq

  python3-flask

  Visual Studio Code editor

  Any OS that can run python3.6 and VS Code (latest version)

- **Testing and execution**

  Windows, mac or Linux OS

3.2 Non-Functional Requirements

- The result should have 70% accuracy or higher
- The program should be platform independent
- The program should by developed with python language
- The program should use less than 8GB of RAM
- The program should have availability rate of 99.9% or higher
- The program should be developed using Visual Studio Code editor
- The application should be easy to use.
- The application should be user friendly
- The application should be easy to remember
- The transformations and animations in GUI should be smooth
- The application should be fast in loading

3.2 Functional Requirements

- As a user I want to be able to classify documents
- As a user I want to be able to create categories

4. PROJECT BACKGROUND

4.1 Document Classification

In text classification there is multiple number of steps starting from starting with text collection and data processing till building the classification model using the training data and applying the model on the test data

These steps can be summarized in the following

1. Data collection

   Collecting data is the first step in text classification studies. The required data are

   samples of texts that belong to the area of interest.

2. Text preprocessing

   Preprocessing is actually a trial to improve text classification by removing worthless information. It may include removal of numbers, punctuation (such as hyphens), and stop words, which are words that can be found in any text like prepositions and pronouns.

3. Data division

   After removing unwanted words and characters, the data are divided into two parts, training data and testing data.

4. Feature extraction

   Texts are characterized by two types of features, external and internal. External features are not related to the content of the text, such as author name, publication date, author gender, and so on. Internal features reflect the text content and are mostly linguistics features, such as lexical items and grammatical categories.

5. Feature selection

   The output of the feature extraction step is a long list of features, ranging from several thousand to hundreds of thousands.

6. Data processing

   In this step, the selected features from the previous step are formatted in a stable way to be represented to the classification algorithm.

7. Classification algorithm training

   In this step, the training matrix that contains the selected features and their corresponding weights in each text of the training data are used to train the classification algorithm.

8. Apply classification model

   The training process yields a classification model that will be tested by means of the testing data. The same features that were extracted from the training data and the same weighting methods will be used to test the classification model.

9. Performance measurement

   The ability of the classification model to classify texts into the correct classes results from all the previously described steps.

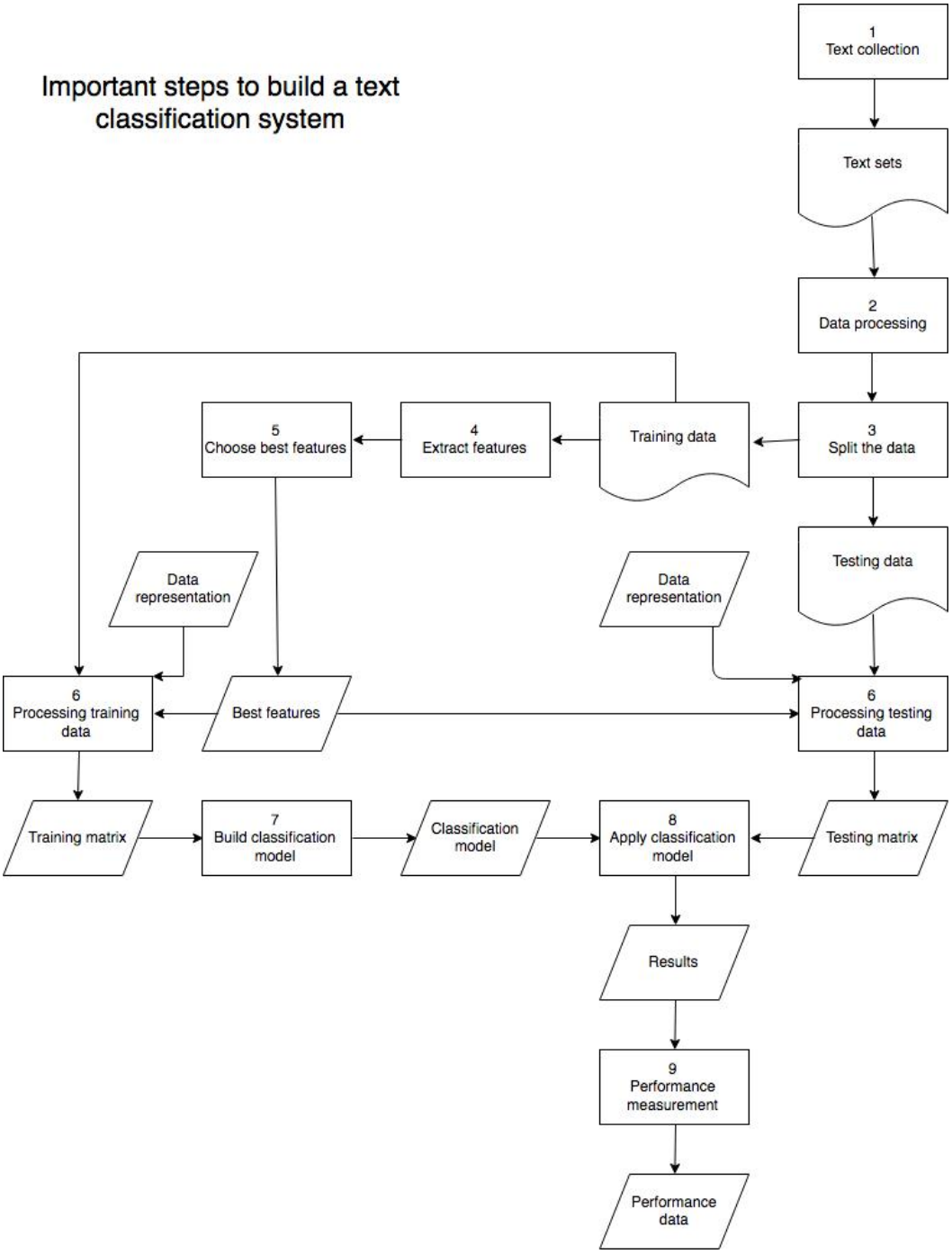


Figure 5 Steps of building text classification system

4.2 Machine Learning Algorithms

Machine learning is an application of artificial intelligence that gives systems the ability to learn and improve from experience automatically without being explicitly programmed

Machine Learning algorithms can be classified into two branches

1- Unsupervised Machine Learning
2- Supervised Machine Learning

Unsupervised learning is a branch of machine learning that learns from data that has not been labeled, unsupervised learning identifies commonalities in the data and reacts based on the presence or absence of such commonalities in each new piece of data and its usually used in Clustering and Anomaly Detection

Supervised Machine Learning is a branch of machine learning that learns from labeled data, supervised learning algorithm analyzes the training data and produces an inferred function, which can be used for mapping new examples

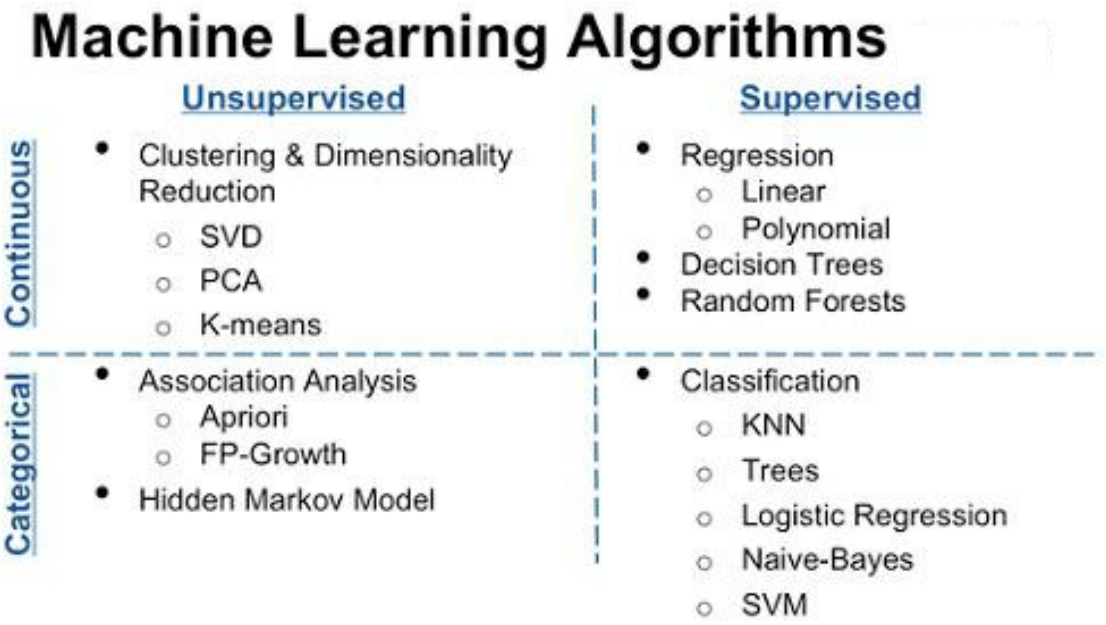


Table 1

4.3 Abstract view

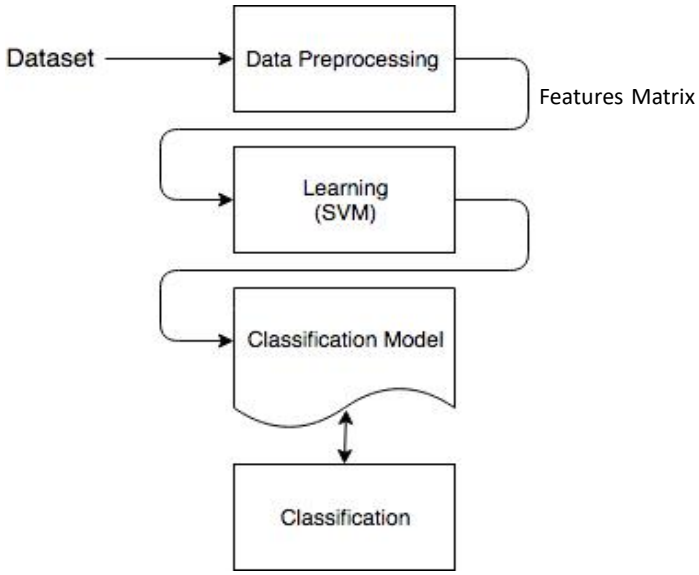


Figure 6 Abstract view

Data Preprocessing block contain all preprocessing steps which we are going to see in detail how the data will go from a normally data to a data that can be used in a machine learning algorithm.

Learning block contain the machine learning steps which we are going to see in detail how the machine will learn.

Classification block this will be used by the user to classify his/her article and we will see in detail how it will be.

4.2.1 Data Preprocessing

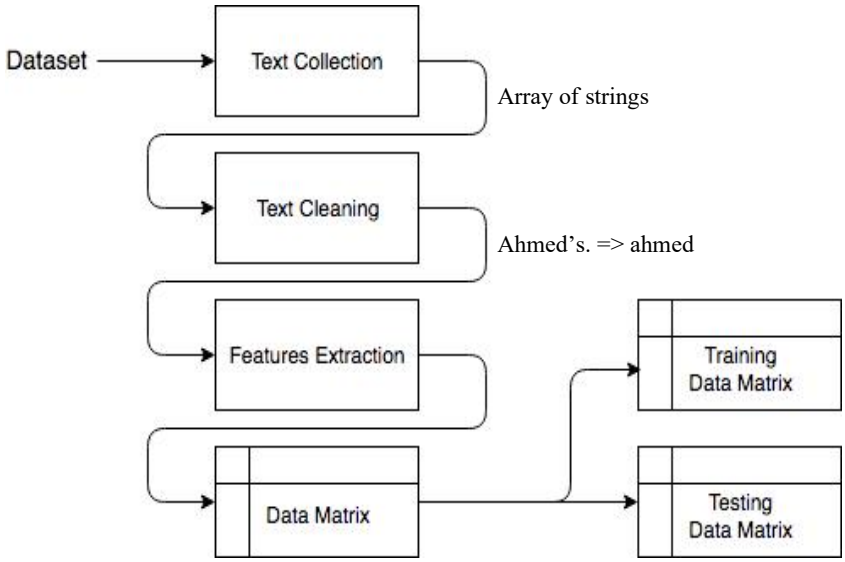


Figure 7 Data Preprocessing

1- Text Collection

In this step the text gets collected from the documents into an array of strings with the correct labels to be used later in learning and testing steps

2- Text Cleaning

In this step the text is cleaned form punctuation marks and short forms

‘s ‘t ‘re ‘d ‘ll ‘ve , ! ? ( )
“ ‘ . (0-9)

After the cleaning the text the is converted to lower case and after that the stop list word get removed

3- Features Extraction

In this step the text gets transformed into future matrix that represent the text in numeric values

4- Data Matrix

A 2D matrix of values where each row represent a document and each column represent a feature value

4.2.2 Learning

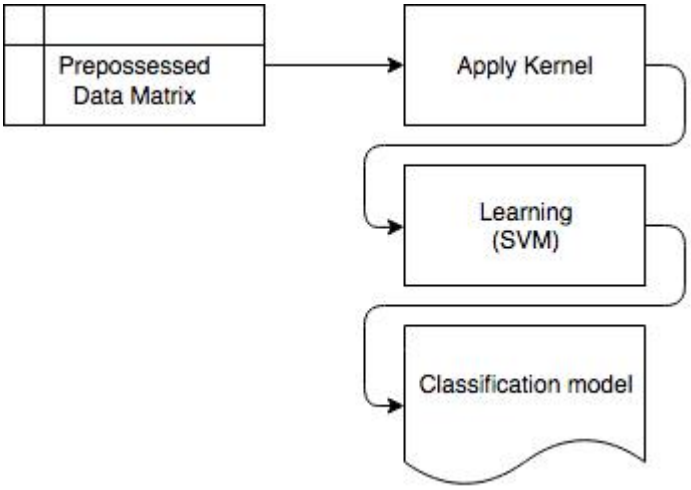


Figure 8 Learning

1- Apply Kernel block is where the kernel function is applied on the data. We have tested the following kernels

- linear
- Radial basis function (RBF)
- sigmoid kernel
- polynomial kernel

2- Learning block will take care of how the machine will learn. In our project the chosen algorithm is SVM based of the results we have got in GP1.

4.2.3 Testing

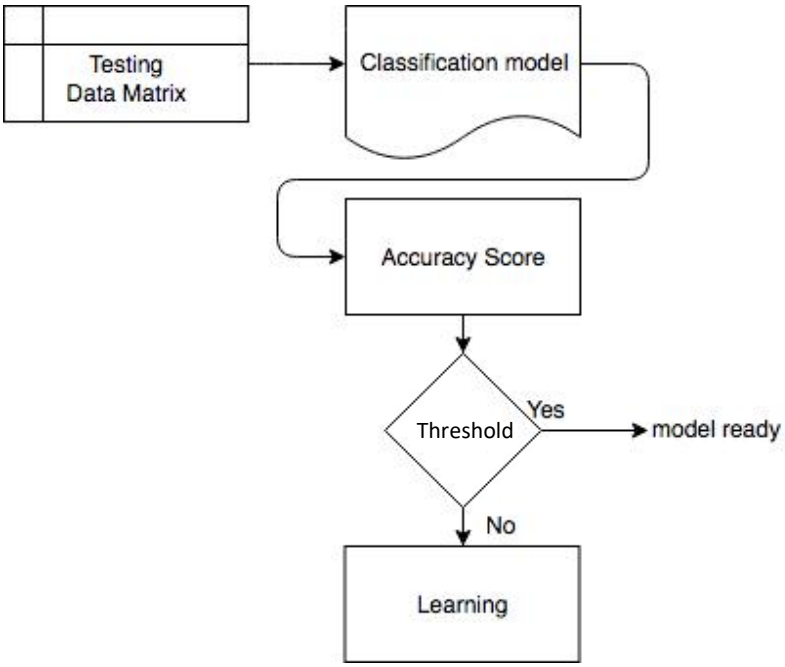


Figure 9 Testing

Accuracy Score block is the result of applying the Testing Data Matrix on the Classification model. based on the resulted accuracy the admin decides where to use the current model or to retrain the model with deferent parameters and kernel

if the accuracy is good enough then model ready to use

else the accuracy is not good enough go to learn step and use another method

4.2.4 Classification

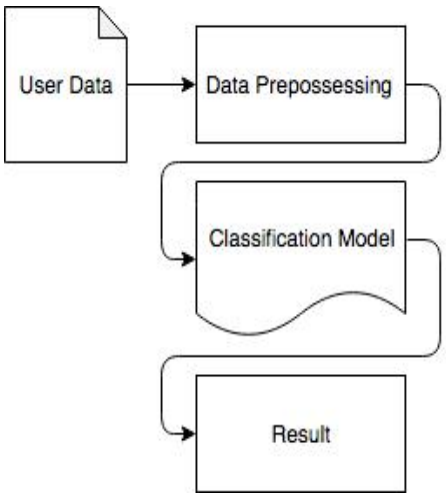


Figure 10 Classification

Data Preprocessing block we will do the same thing for what we did to our data in the first place in Data Preprocessing step.

Result block will show the category of the user data and the percentage of the relation between the user data and each category.

5. DIAGRAMS

5.1 Sequence Diagram

5.1.1 Learning Sequence Diagram

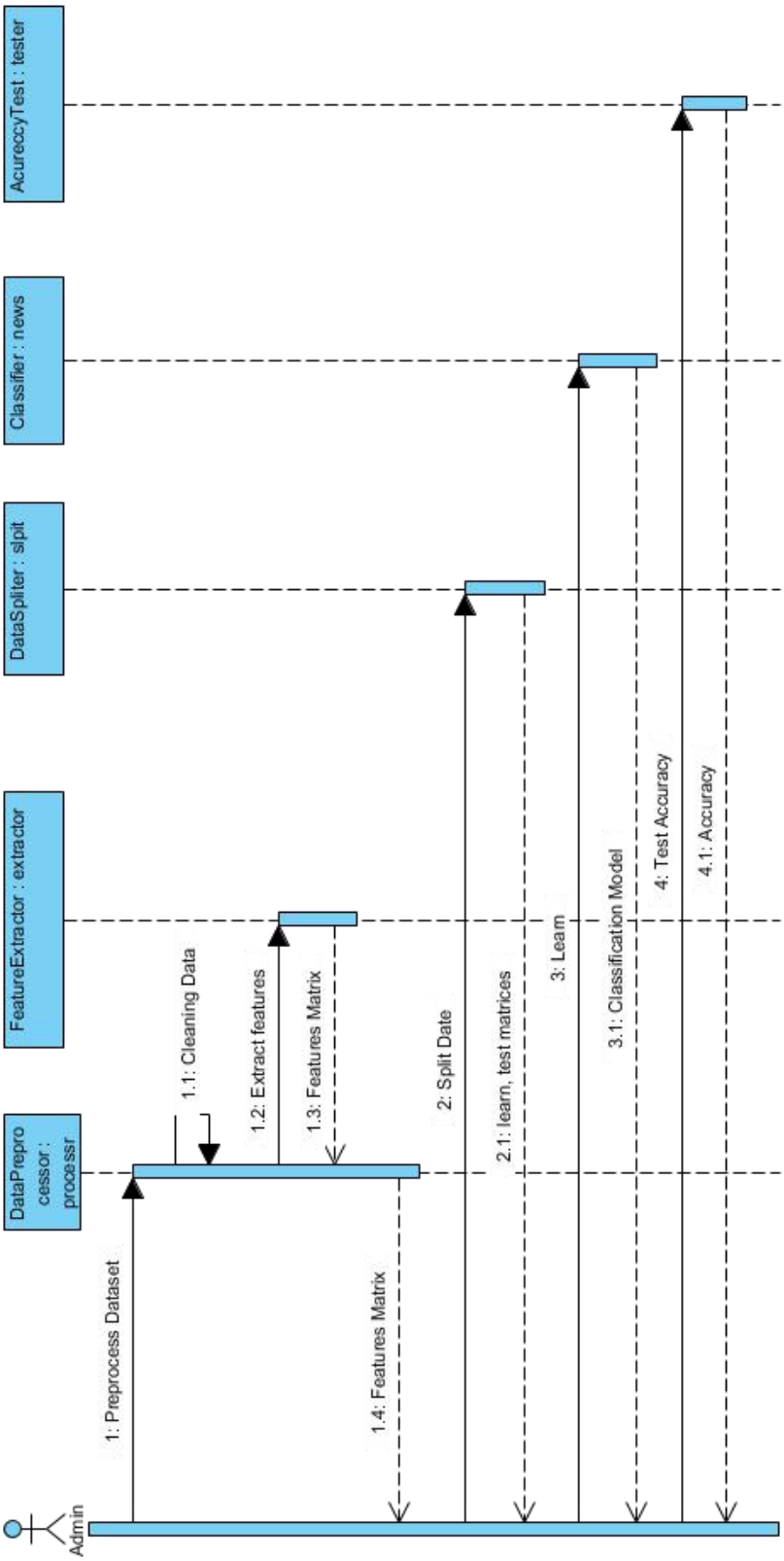


Figure 11 Learning Sequence Diagram

5.1.1 Classification Sequence Diagram

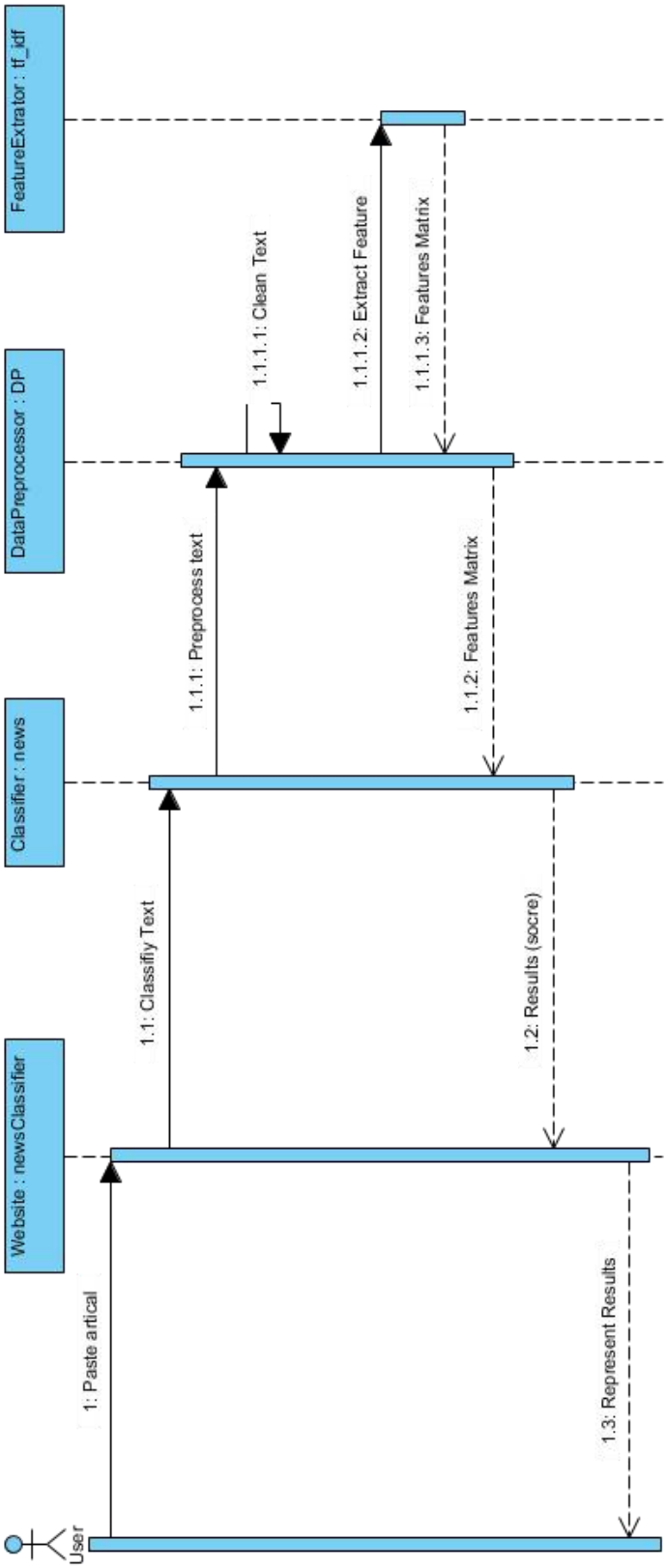


Figure 12 Classification Sequence Diagram

4.1.1 User Case Diagram

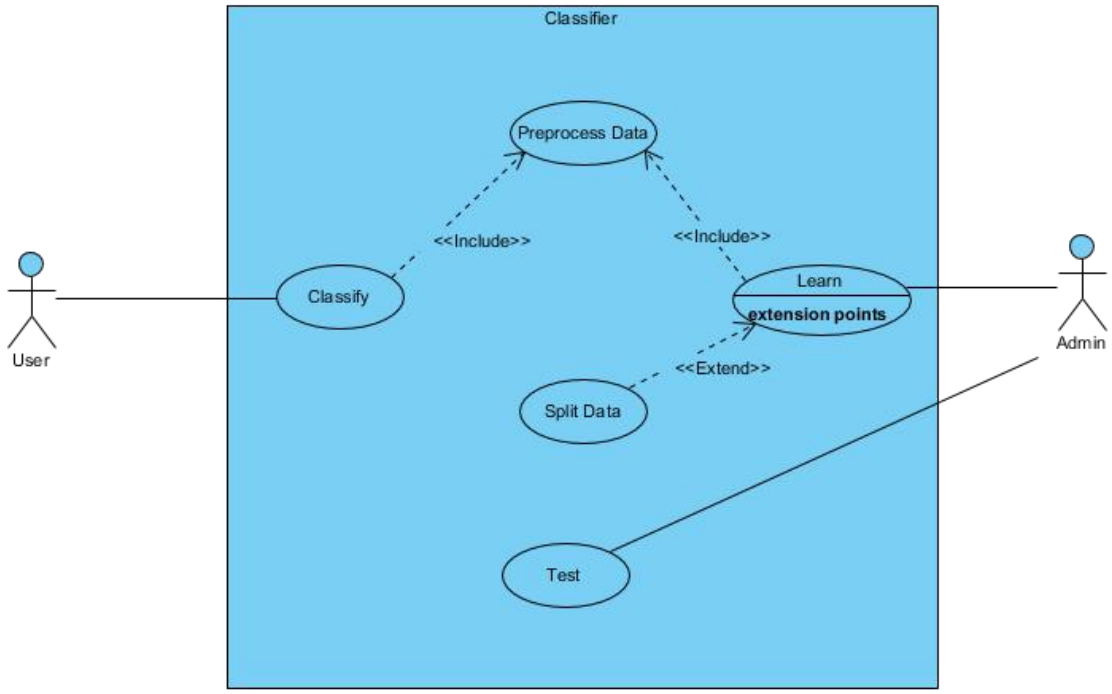


Figure 13 Use Case Diagram

6. LITERATURE REVIEW

6.1 Introduction

In this chapter we present a review of related literature. We proceed with an overview of the general classification space, follow that up with a detailed review of algorithms that can be used in text classification. Then present results from previous classifier comparison work, and close with a summary of the chapter.

6.2 Classification Overview

Classification is a supervised learning task that its goal is to infer a prediction model using a training dataset containing instances whose category membership is known, and then using the model to assign class labels to testing instances whose class labels are unknown. E.g. in spam filtering, already labelled mail as either spam or not spam is used to train a classifier, and the classifier is then used in the future to automatically place mail whose category is unknown, into either spam or not spam categories.

Training of a classifier progresses from gathering a training set that is representative of the real world, from thence, the input data is represented into a feature vector that contains the features that describe the object. With input features in place, a training algorithm e.g. SVM or Naïve Bayes is selected and run on the training set to come up with a predicting function. The function is run on the testing set and its prediction accuracy and performance are measured.

Text classification presents different challenges, this is because some of the words are much more likely to be correlated to the class distribution than others. As such, a wide array of methods has been proposed with a goal of determining the most important features for the purpose of classification. In the following sub sections, we review the algorithms that can be used in the task of text classification

- Random Forest (RF)
- Naïve Bayes (NB)
- K-Nearest neighbors (KNN)
- Neural Network (NN)
- Support Victor Machine (SVM)

6.3 Previous Studies

Thorsten Joachims (1997), comparing SVM, Naïve Bayes for multivariate Bernoulli models, C4.5, Rocchio algorithm, and K-NN. The results show how SVM performed better than all other methods.

Mohammad Khorsheed and Abdulmohsen Al-Thubaity (2013), comparing NB, SVM, C4.5, MLP, and KNN. The result shows that the SVM and Naïve Bayes give better results than the others.

6.4 Feature Extraction

6.4.1 TF-iDF

TF-iDF or 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

The tf–idf value increases proportionally to the number of times a word appears in the document and is offset by the number of documents in the corpus that contain the word, which helps to adjust for the fact that some words appear more frequently in general. Tf–idf is one of the most popular term-weighting schemes today

TF: Term Frequency, which measures how frequently a term occurs in a document. Since every document is different in length, it is possible that a term would appear much more times in long documents than shorter ones. Thus, the term frequency is often divided by the document length (aka. the total number of terms in the document) as a way of normalization

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{i,j}}$$

IDF: Inverse Document Frequency, which measures how important a term is. While computing TF, all terms are considered equally important. However, it is known that certain terms, such as "is", "of", and "that", may appear a lot of times but have little importance. Thus, we need to weigh down the frequent terms while scale up the rare ones, by computing the following

$$idf(w) = log(\frac{N}{df_t})$$

$$w_{i,j} = tf_{i,j} * idf_i$$

tf_{ij} = number of occurrences of i in j
df_i = number of documents containing i
N = total number of documents

6.5 Random Forest (RF)

"A random forest is a classifier consisting of a collection of tree structured classifiers {h(x, Θk),k=1,...} where the {Θk} are independent identically distributed random vectors and each tree casts a unit vote for the most popular class at input x."[1]

Random Forest builds multiple decision trees and merge them together to get more accurate. Most of the options depend on two data objects that generated by random forest. Random Forest does not overfit. Which mean you can run as many trees as you want. It is fast.

In random forest any relation between two trees in the forest is matter because increasing the relation mean increasing the error rate. A tree with a low error rate is a strong classifier.

Increasing the strength of the individual trees decreasing the forest error rate.

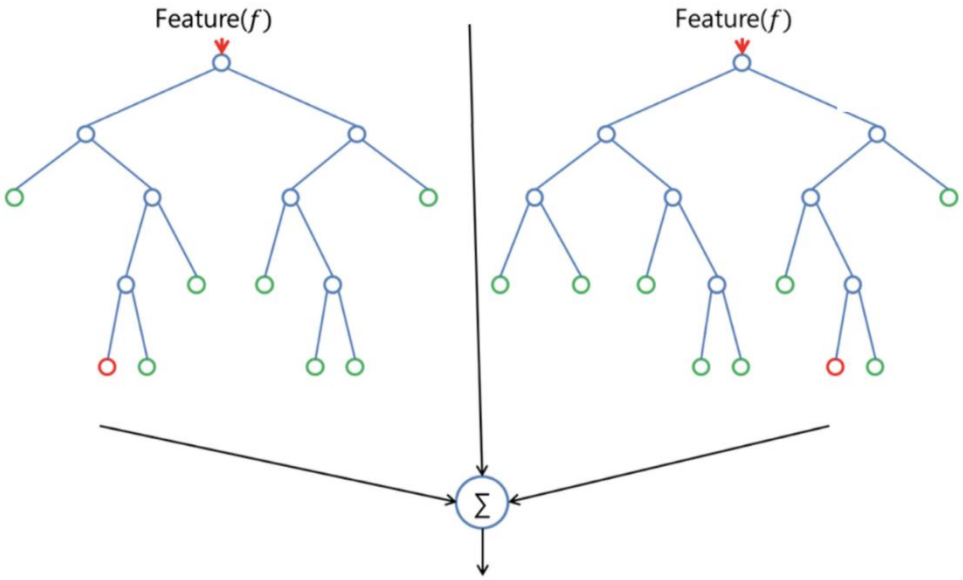


Figure 14 RF

figure 6 shows how the random forest would look like with two trees.

Why we should use Random Forest?

1. Widely used, it can be used in classification and regression.
2. RF is accurate.
3. Reduce overfitting.

Disadvantages of RF

1. Not easily interpretable.
2. Large number of trees may make the algorithm slow.

6.6 K-Nearest Neighbors

the k-nearest neighbors' algorithm (k-NN) is a non-parametric method used for classification and regression. In both cases, the input consists of the k closest training examples in the feature space

the output is a class membership. An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors. If k = 1, then the object is simply assigned to the class of that single nearest neighbor.

k-NN is a type of instance-based learning, or lazy learning, where the function is only approximated locally, and all computation is deferred until classification. The k-NN algorithm is among the simplest of all machine learning algorithms.

The best choice of k depends upon the data; generally, larger values of k reduces effect of the noise on the classification

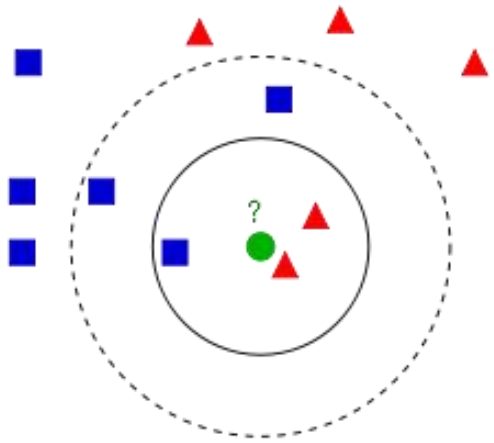


Figure 15 K-NN

Why we should use KNN?

1. The K-Nearest Neighbor (KNN) Classifier is a very simple classifier
2. The KNN algorithm does not have a learning stage (lazy learn)
3. Easy to implement
4. Does not take a lot of system resources

Disadvantages of KNN

1. does not learn anything from the training data and simply uses the training data itself for classification
2. the algorithm must compute the distance and sort all the training data at each prediction
3. High error rate in case of noisy data
4. It is computationally expensive to find the k nearest neighbors when the dataset is very large

6.7 Naïve Bayes

Naïve Bayes is a simple technique for constructing classifiers: models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set

all Naïve Bayes classifiers assume that the value of a particular feature is independent of the value of any other feature, given the class variable

$$\text{Posterior} = \frac{\text{Likelihood} * \text{Prior}}{\text{Evidence}}$$

$$P(C_j \mid A_1, A_2, ..., A_n) = \frac{\left(\prod_{i=1}^{n} P(A_i \mid C_j)\right) P(C_j)}{P(A_1, A_2, ..., A_n)}$$

The Naïve Bayes classifier has also been extended to modeling temporally aware training data, in which the importance of a document may decay with time. As in the case of other statistical classifiers, the naïve Bayes classifier can easily incorporate domain-specific knowledge into the classification process. The domain that the work addresses is that of filtering junk email. Thus, for such a problem, we often have a lot of additional domain knowledge which helps us determine whether an email message is junk or not. For example, some common characteristics of the email which would make an email to be more or less likely to be junk are as follows:

- The domain of the sender such as .edu or .com can make an email to be more or less likely to be junk.
- Phrases such as “Free Money” or over emphasized punctuation such as “!!!” can make an email more likely to be junk.

Why we should use NB?

1. Very simple, easy to implement and fast.
2. Highly scalable. It scales linearly with the number of predictors and data points.
3. Handles continuous and discrete data.
4. less sensitive to irrelevant features.

Disadvantages of NB

1. High error rate

6.8 Neural Network

The basic unit in a neural network is a neuron or unit. Each unit receives a set of inputs, which are denoted by the vector Xi, which in this case, correspond to the term frequencies in the *ith* document. Each neuron is also associated with a set of weights A, which are used in order to compute a function f(·) of its inputs. A typical function which is often used in the neural network is the linear function as follows: pi = A · Xi.

The goal of the neural network approach is to learn the set of weights A with the use of the training data. The idea is that we start off with random weights and gradually update them when a mistake is made by applying the current function on the training example. The magnitude of the update is regulated by a learning rate μ. This forms the core idea of the perceptron algorithm.

5.8 Support Vector Machine (SVM)

Support vector machine (SVM) method performs regression and classification tasks by constructing nonlinear decision boundaries. Because of the nature of the feature space in which these boundaries are found, Support Vector Machines can exhibit a large degree of flexibility in handling classification and regression tasks of varied complexities. There are several types of Support Vector models including linear, polynomial, radial basis function (RBF), and sigmoid.

Support Vector Machines are based on the concept of decision planes that define decision boundaries. A decision plane is one that separates between a set of objects having different class memberships.

In the next example, the objects belong either to class green or red. The separating line defines a boundary on the right side of which all objects are green and to the left of which all objects are red. Any new object (white circle) falling to the right is labeled, i.e., classified, as green or classified as red.

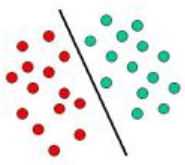


Figure 16 Linear SVM

In figure 8 a classic example of classic example of linear classifier. Classifier that separate the objects into their group with a line. Most classification tasks are not

simple and more complex structures are needed in order to make optimal separation. So, to classify new object correctly its basis on the examples are available.

To separate the objects in the next figure it is require a curve to separate it, which is more complex than a line. Classification tasks based on drawing separating lines to distinguish between objects of different class memberships are known as hyperplane classifiers. Support Vector Machines are particularly suited to handle such tasks.

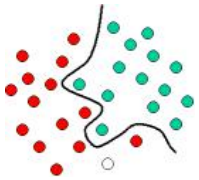


Figure 17 Non-Linear SVM

In the next figure the basic idea of support vector machine. The left side is the original objects mapped or rearranged using a set of mathematical functions known as kernels. The new setting is linearly separable. So, instead of using complex curve we can fine optimal line that can separate the objects.

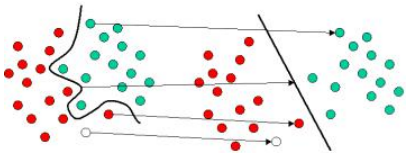


Figure 18 Kernel

To construct an optimal hyperplane, support vector machine uses iterative training algorithm which is used to minimize the error function.

Kernel functions that can be used in support vector machines models

$$K(\mathbf{X}_i, \mathbf{X}_j) = \begin{cases} \mathbf{X}_i \cdot \mathbf{X}_j & \text{Linear} \\ (\gamma \mathbf{X}_i \cdot \mathbf{X}_j + C)^d & \text{Polynomial} \\ \exp(-\gamma |\mathbf{X}_i - \mathbf{X}_j|^2) & \text{RBF} \\ \tanh(\gamma \mathbf{X}_i \cdot \mathbf{X}_j + C) & \text{Sigmoid} \end{cases}$$

Why we should use SVM?

1. Support Vector Machine in general give high accuracy.
2. Powerful, SVM can work even if there is a noise.
3. Most text categorization problems are linearly separable. The idea of SVM is to find such linear.

7. TESTING AND RESULTS

7.1 Introduction

In order to decide which classification model is accurately capturing a pattern, we must evaluate that model. The result of this evaluation is important for deciding how trustworthy the model is, and for what purposes we can use it. Evaluation can also be an effective tool for guiding us in making future improvements to the model.

The dataset contains 7 classes, each class contain one topic. The classes are cultural news, sports news, social news, economic news, political news, general news and IT news. 100 texts for each class from each newspaper. 4,200 texts in total. One newspaper per day. We have used Boolean representation with no modification for terms selection.

Table 2

SVM	NB	RF	K-NN
94.58	61.06	90.11	40.32

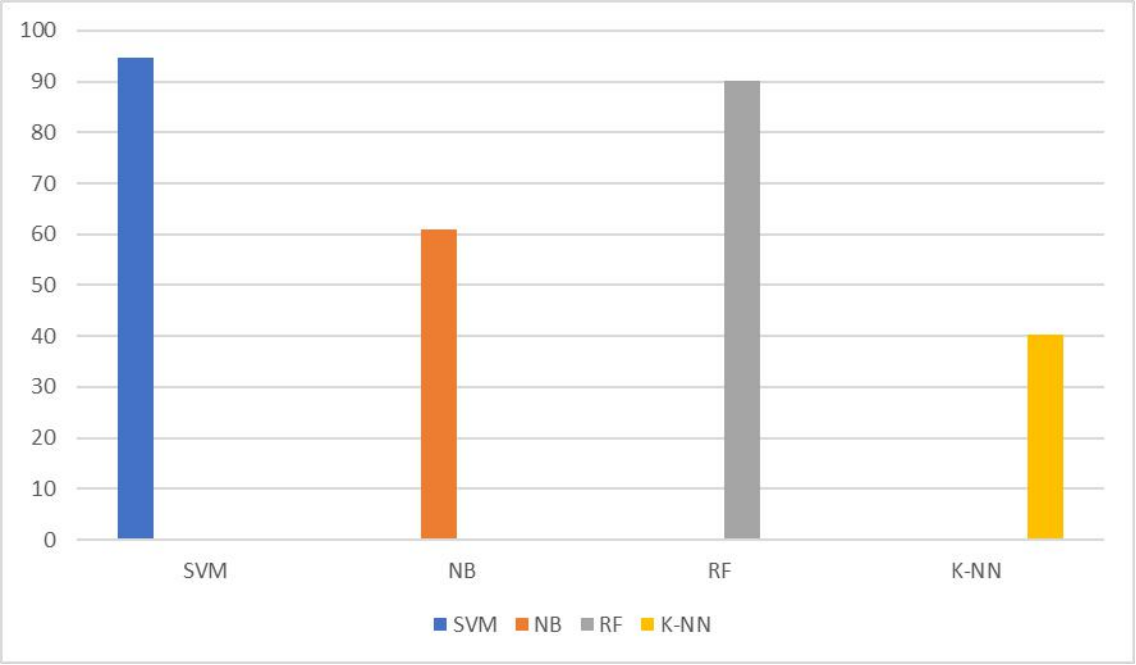


Figure 19 Accuracy Chart

7.2 Conclusion

From this study, we found that as previous studies have shown, SVM does very well in text classification. In short, given a text classification problem, SVM could be used without a big compromise on the classification accuracy. In addition, factoring in training time, in some cases it would be advised to use Naïve Bayes instead of SVM.

The best method for news articles is SVM using RBF as kernel and TF-iDF as features extraction method.

8. DEVELOPMENT TOOLS

8.1 RapidMiner

Figure 20 RapidMiner

RapidMiner is a data science software platform developed by the company of the same name that provides an integrated environment for data preparation, machine learning, deep learning, text mining, and predictive analytics.

RapidMiner brings artificial intelligence to the enterprise through an open and extensible data science platform.

Built for analytics teams, RapidMiner unifies the entire data science lifecycle from data prep to machine learning to predictive model deployment. More than 450,000 analytics professionals use RapidMiner products to drive revenue, reduce costs, and avoid risks. RapidMiner founded in 2007.

RapidMiner software used in this project to compare between the selected methods result in the same data to choose one to work in.

8.2 Python 3 (Programing Language)

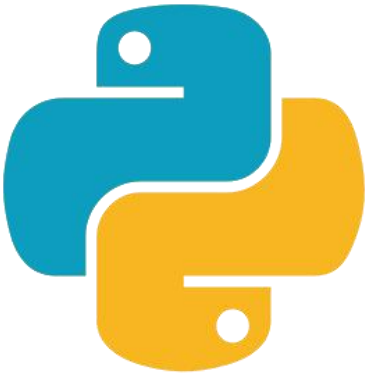


Figure 21 Python

Python is an interpreted, high-level programming language for general purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library. Python interpreters are available for many operating systems.

Since 2003, Python has consistently ranked in the top ten most popular programming languages in the TIOBE Programming Community Index where, as of January 2018, it is the fourth most popular language (behind Java, C, and C++). It was selected Programming Language of the Year in 2007 and 2010.

Advantages of using python

1- Python is easy to learn for even a novice developer
2- Supports multiple systems and platforms.
3- Object Oriented Programming-driven
4- Allows to scale even the most complex applications with ease
5- A large number of resources are available for Python

Disadvantages of Python

1- Python is slow
2- Python is not a good choice for memory intensive tasks

9.3 Visual Studio Code

Figure 22 Visual Studio Code

Visual Studio Code is a source code editor developed by Microsoft for Windows, Linux and macOS. It includes support for debugging, embedded Git control, syntax highlighting, intelligent code completion, snippets, and code refactoring. It is also customizable, so users can change the editor's theme, keyboard shortcuts, and preferences. It is free and open-source, although the official download is under a proprietary license.

Visual Studio Code is based on Electron, a framework which is used to deploy Node.js applications for the desktop running on the Blink layout engine. Although it uses the Electron framework, the software does not use Atom

In the Stack Overflow 2018 Developer Survey, Visual Studio Code was ranked the most popular developer environment tool, with 34.9% of 75,398 respondents claiming to use it

7.4 Libraries

Scikit-learn: an open source machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, logistic regression, Naïve Bayes, random forests, gradient boosting, k-means and DBSCAN. designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

NumPy: the fundamental package for scientific computing with Python.

SciPy: an open source library of scientific tools.

Matplotlib: a python 2D and 3D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms.

8.4 Code Stucture

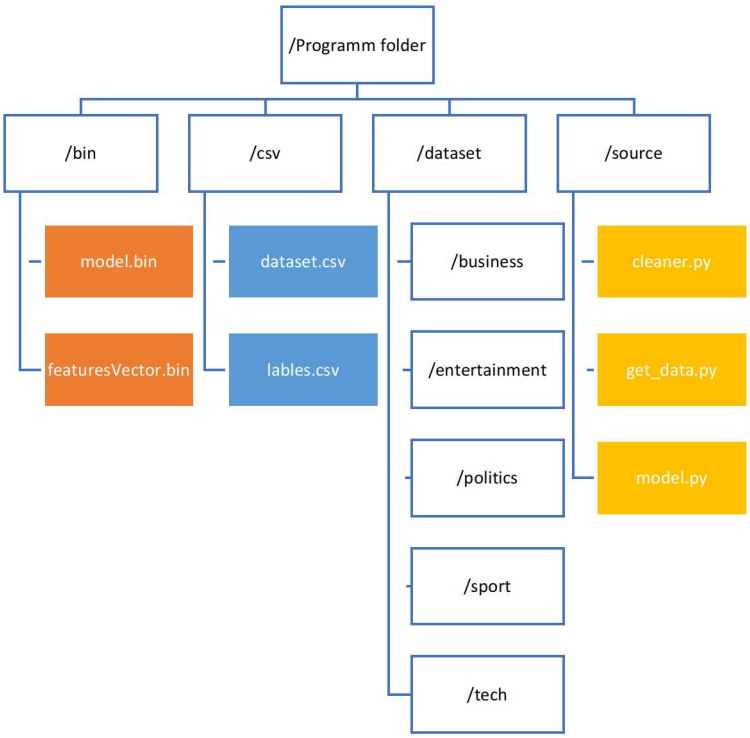


Figure 23 Code Structure

9. FRONTEND APPLICATION

9.1 MVC Architecture

The Model-View-Controller (MVC) is an architectural pattern that separates an application into three main logical components: the model, the view, and the controller. Each of these components are built to handle specific development aspects of an application. MVC is one of the most frequently used industry-standard web development frameworks to create scalable and extensible projects.

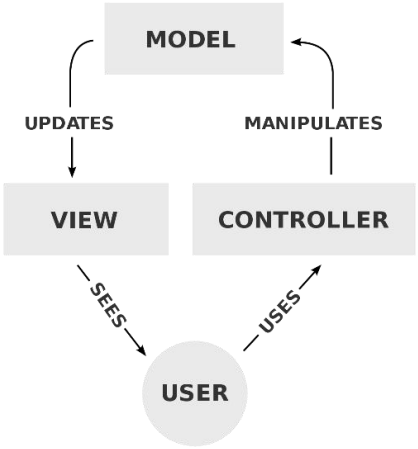


Figure 24 MVC Architecture

Model

The Model component corresponds to all the data-related logic that the user works with. This can represent either the data that is being transferred between the View and Controller components or any other business logic-related data. For example, a Customer object will retrieve the customer information from the database, manipulate it and update it data back to the database or use it to render data.

View

The View component is used for all the UI logic of the application. For example, the Customer view will include all the UI components such as text boxes, dropdowns, etc. that the final user interacts with.

Controller

Controllers act as an interface between Model and View components to process all the business logic and incoming requests, manipulate data using the Model component and interact with the Views to render the final output. For example, the Customer controller will handle all the interactions and inputs from the Customer View and update the database using the Customer Model. The same controller will be used to view the Customer data

9.2 Flask

Flask is a micro web framework written in Python. ... Extensions are updated far more regularly than the core Flask program. Applications that use the Flask framework include Pinterest, LinkedIn, and the community web page for Flask itself

Figure 25 Flask

9.3 Design Code Structure

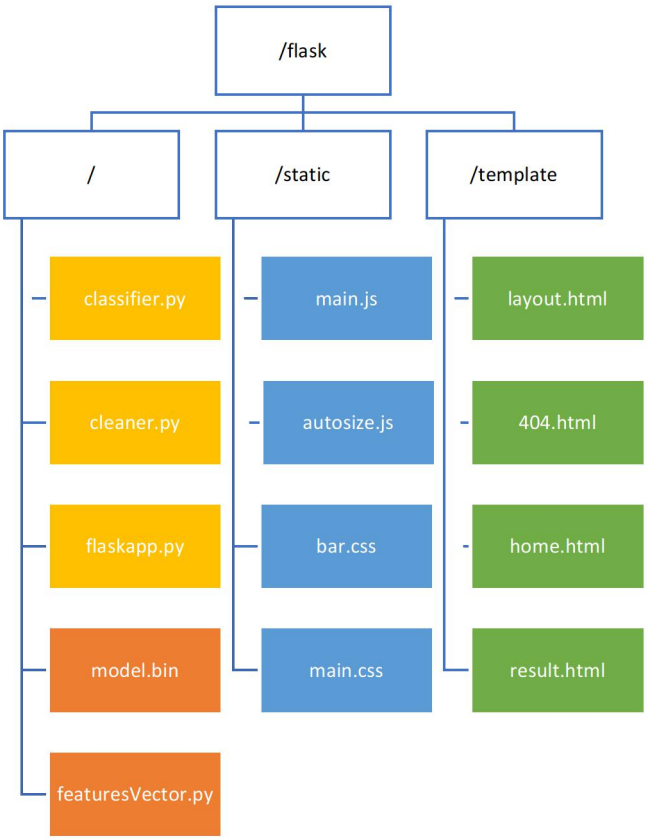


Figure 26 Design Code Strucure

9.3 Application Screenshots

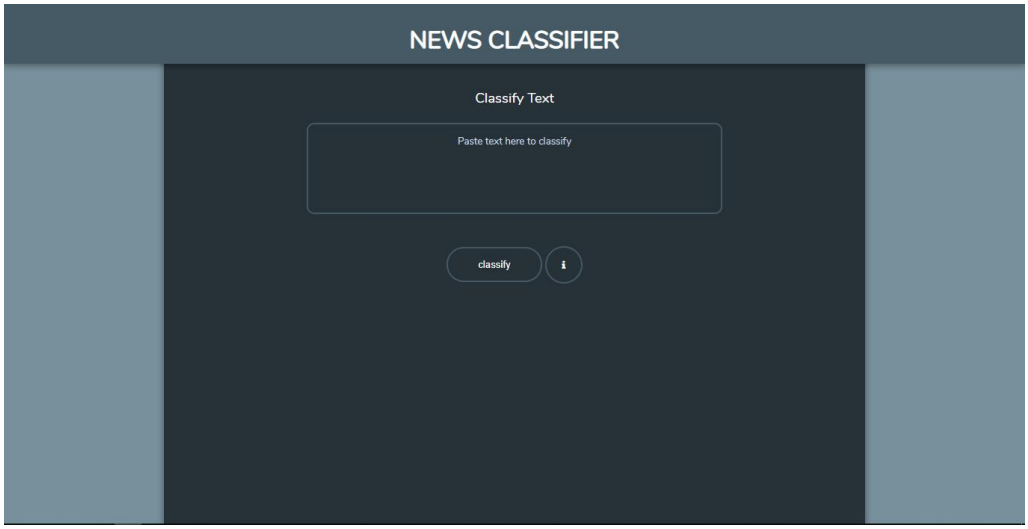


Figure 27 Screenshot1

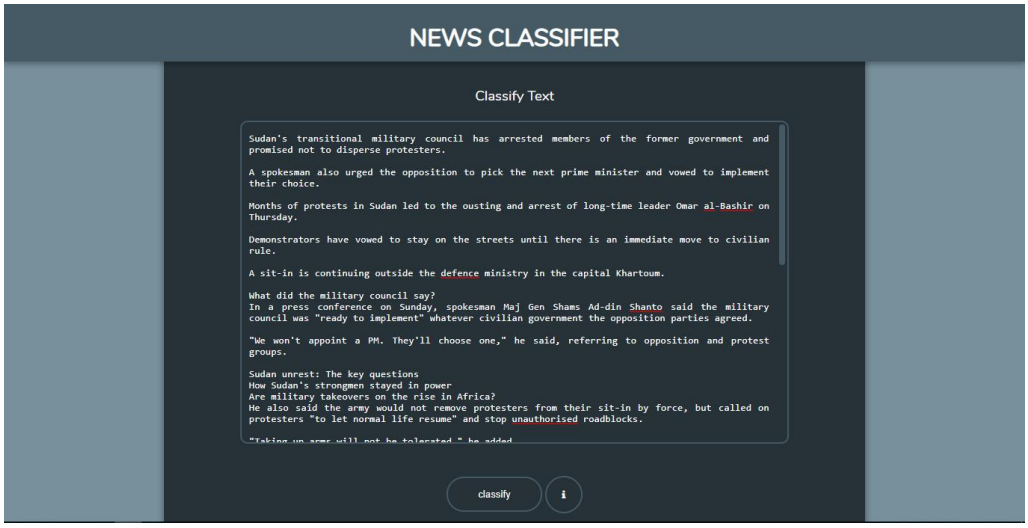


Figure 28 Screenshot2

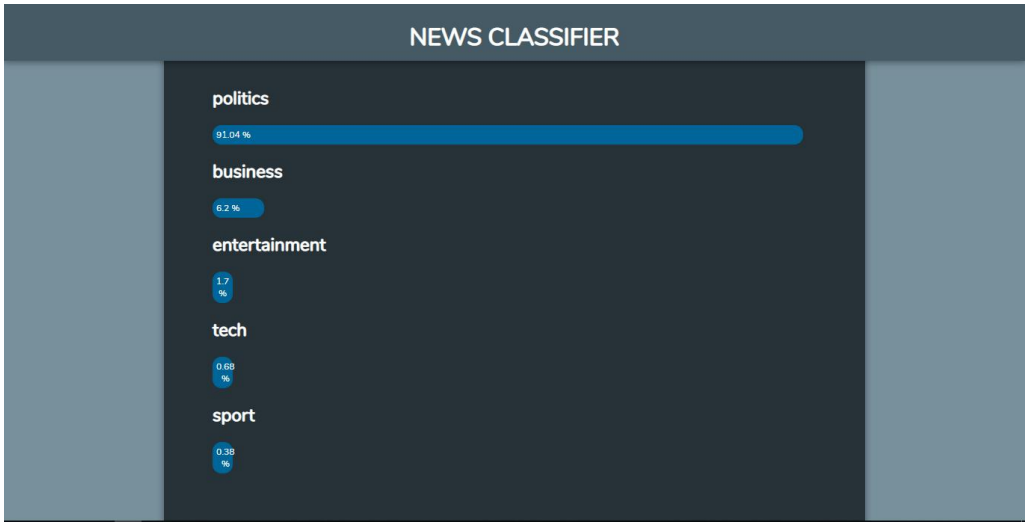


Figure 29 Screenshot3

10. REFERENCES

1. Ali J., Khan R., Ahmad N., Maqsood I. (2012). Random Forest and decision trees.
2. Bijalwan V., Kumar V., Kumari P., Pascual J. (2014). KNN based Machine Learning Approach for Text and Document Mining.
3. Breiman, L. (2001). Random Forests, Machine Learning.
4. Charu A., Cheng Z. (2012). Mining Text Data.
5. Joachims T. (1999). Transductive interface for text classification using support vector machines.
6. Khorsheed M., Al-Thubaity A. (2013) Comparative evaluation of text classification techniques using a large diverse Arabic dataset.
7. Kumar B. (2017). Multi-Class Text Document Classification with OCSVM and LSI.
8. Mita D., Mukesh Z. (2011). Automatic Text Classification: A Technical Review.
9. StatSoft, Inc. Electronic Statistics Textbook. (2013). http://www.statsoft.com/textbook/.
10. Sparck J. (1972). A statistical interpretation of term specificity and its application in retrieval.
11. Salton G., Fox E., Wu W. (1983). Extended Boolean information retrieval.
12. Salton G., McGill M. (1983). Introduction to modern information retrieval.
13. Salton G., Buckley C. (1988). Term-weighting approaches in automatic text retrieval.
14. Sebastiani F. (2002). Machine Learning in Automated Text Categorization.
15. Wang L. (2005). Support Vector Machines: Theory and Applications.
16. Wu H., Luk R., Wong K., Kwok K. (2008). Interpreting TF-IDF term weights as making relevance decisions.
17. Yousef M., Manevitz L. (2006). Multi-Class document classification via Neural Networks.
18. MVC Framework. https://www.tutorialspoint.com/mvc_framework/mvc_framework_introduction.htm.

19. Tripathi M. (2018). How to process textual data using TF-IDF in Python. https://medium.freecodecamp.org/how-to-process-textual-data-using-tf-idf-in-python-cd2bbc0a94a3.
20. Chang C., Lin C. (2018). LIBSVM. https://www.csie.ntu.edu.tw/~cjlin/libsvm/.