**Articles Journal using Data Mining**

**Prepared By:**

1. **Youssof Kamal Eldin**
2. **Abdelrahman Salah Abdelrazzak**
3. **Abdelhady Eslam Mohamed**
4. **Mahmoud Hamed Hafez**
5. **Abdelrahman Nasser Ahmed**
6. **Mohamed sayed Saad**

**Under supervision:**

**Prof. Marghny Hassan Mohamed**

**Contents**

**Abstract**

**1. Introduction 9** 1.1 Overview …………………………………………….. 9

1.2 Project Motivation …………………………………… 9

1.3 Project Objective ……………………………………. 10

1.4 Data Sets Used ……………………………………… 10  
 1.4.1 Economic News Article Tone and Relevance…… 10  
 1.4.2 U.S. economic performance based on news articles 11

1.5 Tools ………………………………………………….. ….12  
 1.5.1 Language …………………………………………....12  
 1.5.2 Databases …………………………………………...12  
 1.5.3 Backend (Django Platform) ………………………...13  
 1.5.4 Front End …………………………………………...13  
 1.5.5 Classification Algorithms …………………………..15

**2. Sentiment Analysis 16** 2.1 Data Preprocessing………………………………………..16  
 2.2 Classifiers & Training………………………………….....22  
 2.2.1 Logistic Regression Classifier……………………...23  
 2.2.2 K-Nearest Neighbor Classifier……………………..25

2.2.3 Support Vector Classifier(SVC)…………………...26

2.2.4 Decision Tree………………………………………27

2.2.5 Random Forest……………………………………..28  
 2.3 Visualization……………………………………………...29

2.3.1 Logistic Regression Classifier……………………...30

2.3.2 K-Nearest Neighbor Classifier……………………..31

2.3.3 Support Vector Classifier(SVC)……………………31

2.3.4 Decision Tree………………………………………34

2.3.5 Random Forest…………………………………….35

2.4 Conclusion……………………………………………….41

**3. Software project management plan 43**

3.1 Introduction ………………………………………………43  
 3.1.1 Project Overview…………………………………...43  
 3.1.2 Project Deliverables………………………………..43  
   
 3.2 Project Organization…………………………………….44  
 3.2.1 Software Process Model…………………………..44  
 3.2.2 Tools and Techniques……………………………..51  
 3.3 Project Management Plans……………………………...55  
 3.1.1 Tasks…………………………………………........55

1. **Analysis and Design 56**4.1 Design: ………………………………………………… 56  
    4.1.1 ERD ……………………………………………..56  
    4.1.2 Data Tables:……………………………………...57  
   4.2 project requirement:…………………...………………..66  
   4.3 Project values:…………………………………………..66  
   4.4 Special Issues Or Constrains:……………………………...66

**Implementation 67**  
5.1 User Manual ..................................................................………...67  
 5.1.1 Sign UP ………………………………………………….67  
 5.1.2 Log in ………………………………………..…………..69  
 5.1.3 Log Out ………………………………………………….70  
 5.1.4 Make Articles ………………………………………....... 70  
 5.1.5 Main Page ……………………………………………….72  
 5.1.6 My Profile………………………………………………..73  
 5.1.7 User Profile ……………………………………………...73  
 5.1.8 Show Article …………………………………………….74  
 5.1.9 My Notifications ………………………………………...76  
 5.1.10 Settings ………………………………………...............77  
 5.1.11 Help Center …………………………………………….79  
 5.1.12 Privacy Policy…………………………………………. 79

**References 80**

**List of Figures**

2.3.1.1 Logistic Regression - maxItr 3 ……………………………………………..30  
2.3.1.2 Logistic Regression - maxItr 10 ……………………………………………30  
2.3.1.3 Logistic Regression - maxItr 30 ……………………………………………30  
2.3.2.1 K-Nearest Neighbor - maxItr 30 …………………………………………..31  
2.3.3.1 SVC with Linear kernel ………………………………………………………. 31  
2.3.3.2 SVC with Radial Basis function kernel ………………………………… 32  
2.3.3.3 SVC with sigmoid/Gaussian kernel …………………………………….. 32  
2.3.3.4 SVC with Polynomial kernel - degree 2 ………………………………. 33  
2.3.3.5 SVC with Polynomial kernel - degree 3 ………………………………. 33  
2.3.3.6 SVC with Polynomial kernel - degree 4 ………………………………. 34  
2.3.4.1 Decision Tree Classifier using gini ………………………………………. 34  
2.3.4.2 Decision Tree Classifier using entropy ……………………………….. 35  
2.3.5.1 Random Forest Classifier - gini - 5 estimators …………………….. 35  
2.3.5.2 Random Forest Classifier - gini - 10 estimators …………………… 36  
2.3.5.3 Random Forest Classifier - using gini - using 25 estimators … 36  
2.3.5.4 Random Forest Classifier - gini - 100 estimators……………………37  
2.3.5.5 Random Forest Classifier - gini - 130 estimators……………………37  
2.3.5.6 Random Forest Classifier - gini - 300 estimators……………………38  
2.3.5.7 Random Forest Classifier - entropy - 5 estimators…………………38  
2.3.5.8 Random Forest Classifier - entropy - 10 estimators ……………..39  
2.3.5.9 Random Forest Classifier - entropy - 25 estimators ……………..39  
2.3.5.10 Random Forest Classifier - entropy - 100 estimators ………….40  
2.3.5.11 Random Forest Classifier - entropy - 130 estimators ………….40  
2.3.5.12 Random Forest Classifier - entropy - 300 estimators ………….41

4.0ERD:…………………………………………………………………………………………56  
4.1Database articles\_journal:……………………………………………………….574.2Register\_Users:……………………………………………………………………..584.3Register\_notifications:…………………………………………………………..594.4Articles\_posts:……………………………………………………………………….604.5Articles\_likesdislikes:……………………………………………………………..614.6Articles\_comments:……………………………………………………………… 624.7Django\_session:…………………………………………………………………....634.8Django\_migrations:……………………………………………………………….634.9Django\_content\_type:…………………………………………………………..634.10Django\_admin\_log:……………………………………………………………….64  
4.11 Auth\_user\_user\_permissions:……………………………………………….64  
4.12 Auth\_user\_groups:………………………………………………………………..64  
4.13 Auth\_user:……………………………………………………………………………. 65  
4.14 Auth\_permissions:……………………………………………………………….. 65  
4.15 Auth\_Group\_permissions:……………………………………………………. 65  
4.16 Auth\_group:…………………………………………………………………………. 66

5.1 Sign UP………………………………………………………………………….. 67  
5.2 Sign UP …………………………………………………………………………. 68  
5.3 Sign UP form DB…………………………………………………………….. 68  
5.4 Log in form …………………………………………………………………… 69  
5.5 Log in test …………………………………………………………………….. 69  
5.6 Drop down list ……………………………………………………………… 70  
5.7 Make article area …………………………………………………………. 70  
5.8 Header of page …………………………………………………………….. 71  
5.9 Make article area test …………………………………………………… 71  
5.10 Article from DB ……………………………………………………………. 71  
5.11 Main Page …………………………………………………………………….72  
5.12 Main Page2 …………………………………………………………………..72  
5.13 My Profile page …………………………………………………………….73  
5.14 User Profile page ………………………………………………………….735.15 Show article page ……………………………………………………………….74  
5.16 Comment test …………………………………………………………………….75  
5.17 Comment from DB ……………………………………………………………..75  
5.18 My Notifications list ……………………………………………………………76  
5.19 My Notifications page ………………………………………………………..76  
5.20 My Notifications from DB …………………………………………………..77  
5.21 Name setting ……………………………………………………………………..77  
5.22 Password setting ………………………………………………………………. 78  
5.23 Deactivate& Delete …………………………………………………………… 78

**Abstract**

Data mining can be defined as the process of finding previously unknown patterns , trends in databases and using that information to build predictive models.

Alternatively , it can be defined as the process of data selection and exploration and building models using vast data stores to uncover previously unknown pattern. Data mining in not new it has been used intensively and extensively by financial institution, for credit scoring and fraud detection. Another factor is that the huge amounts of data generated by Article sentiment analysis transaction are too complex and voluminous to be processed and analyzed by traditional methods.

This project aims for mining the relationship of some Article for classification (positive/negative).The data mining methods and techniques will be explored to identify the suitable methods and techniques for efficient classification of Article and in mining useful patterns.

**Our goals**

Save time and effort  
Find the filed of (positive / negative) for article  
Making the decision-making in some things easier

**Chapter 1**

**Introduction**

**1.1 Overview**

our website is called "Articles Blog" it is a website for writing and analyzing articles about Economy and finding positive / negative classified articles based on Natural Language Processing in this website user can:-

– see all articles in the website.  
– search for articles by:

– Tags.  
– Opinions "This article is positive or negative".  
– Words "That if the user wants to search for articles that have specific words".

– make new article with its title and can add tags for the article.  
– Like / Dislike article and Comment on it.

**1.2 Project Motivation:**

Users need positive / negative opinions about specific subject.  
Users spent time and effort for reading all available articles.  
The difficulty of the process of the articles revisions because of its huge number.

**1.3 Project Objective:**

from these points we have mentioned in the motivation we realized that we are really need a solution to help us to overcome these problems. to overcome these problems we will go to make a website to predict the status of the article (positive / negative) using some of the methods of data mining algorithms.

**1.4 Data Sets Used**

**1.4.1 Economic News Article Tone and Relevance**

Contributors read snippets of news articles. They then noted if the article was relevant to the US economy and, if so, what the tone of the article was. Tone was judged on a 9 point scale (from 0 to 9, with 0 representing absolutely negative). Dataset contains these judgments as well as the dates, source titles, and text. Dates range from 1951 to 2014.

**Number Of Articles :** 8000 Articles

**Number of Attributes :** 15 Attributes including Positivity field

**Attributes Information :**

– Unit id:- It has id for each article  
– Golden :- FALSE  
– Unit state :- Finalized  
– Trusted judgments :- 3  
– Last judgment at :- DATE  
– Positivity :- From 0 to 9  
– Positivity : Confidence :- 0 to 1  
– Relevance :- Yes , No , not sure  
– Relevance : Confidence :- 0 to 1  
– Article id :- Serial number  
– Date :- DATE  
– Headline :- Title  
– Text : main article

**Class :**

– Positive  
– Negative

**1.4.2 U.S. economic performance based on news articles**

Contributors viewed a new article headline and a short, bolded excerpt of a sentence or two from the attendant article. Next, they decided if the sentence in question provided an indication of the U.S. Economy health, then rated the indication on a scale of 0-9, with 0 being negative and 9 being positive.

**Number Of Article :** 5000 Article

**Number of Attributes :** 17 Attributes including Positivity field

**Attributes Information :**

– Unit id:- It has id for each article  
– Golden :- FALSE  
– Unit state :- Finalized  
– Trusted judgments :- 3  
– Last judgment at :- DATE  
– Positivity :- From 0 to 9  
– Positivity : Confidence :- 0 to 1  
– Relevance :- Yes , No , not sure  
– Relevance : Confidence :- 0 to 1  
– Article id :- Serial number  
– Date :- DATE  
– Headline :- Title  
– Line id  
– Next sentence  
– Previous sentence  
– Text

**Class :**

– Positive  
– Negative

**1.5 Tools**

**1.5.1 Language**

**Python**

**1.5.2 DataBases**

**MYSQL**

MySQL is a very popular, open source DBMS MySQL databases are relational

**MONGODB**

MONGODB is a very popular, open source RDBMS MONGODB databases are NON relational

**1.5.3 BackEnd (Django PlatForm)**

– it is python based platform for web development  
– Loose coupling  
– Less code.  
– Quick development.  
– Don’t repeat yourself (DRY).  
– Explicit is better than implicit.  
– Consistency.

**1.5.4 Front End**

**HTML :**

The purpose of a web browser (Chrome, IE, Firefox, Safari) is to read HTML documents and display them, the browser does not display the HTML tags, but uses them to determine how to display the document.

**CSS :**

CSS stands for Cascading Style Sheets, CSS saves a lot of work It can control the layout of multiple web pages all at once.

HTML was never intended to contain tags for formatting a web page , HTML was created to describe the content of a web page, like:

<h1>This is a heading</h1>

<p>This is a paragraph.</p>

When tags like *font*, and color attributes were added to the HTML 3.2 specification, it started a nightmare for web developers. Development of large websites, where fonts and color information were added to every single page, became a long and expensive process.  
To solve this problem, the World Wide Web Consortium (W3C) created CSS. 

**JAVA SCRIPT :**

One of many JavaScript HTML methods is

getElementById()

This example uses the method to "find" an HTML element (with id="demo") and changes the element content (innerHTML) to "Hello JavaScript"

getElementById('demo').innerHTML = "Hello JavaScript";

**JQUERY :**

– jQuery is a lightweight, "write less, do more", JavaScript library.  
– The purpose of jQuery is to make it much easier to use JavaScript on your website.  
– jQuery takes a lot of common tasks that require many lines of JavaScript code to accomplish, and wraps them into methods that you can call with a single line of code.  
– jQuery also simplifies a lot of the complicated things from JavaScript, like AJAX calls and DOM manipulation.  
– The jQuery library contains the following features:

– HTML/DOM manipulation  
– CSS manipulation  
– HTML event methods  
– Effects and animations  
– AJAX  
– Utilities

**BOOTSTRAP :**

– Bootstrap is a free front-end framework for faster and easier web development.  
– Bootstrap includes HTML and CSS based design templates for typography, forms, buttons, tables, navigation, modals, image carousels and many other, as well as optional JavaScript plugins.  
– Bootstrap also gives you the ability to easily create responsive designs.  
– There are two ways to start using Bootstrap on your own web site

– Download Bootstrap from getbootstrap.com  
– Include Bootstrap from a CDN

**We use the first way**

**1.5.5 Classification Algorithms**

the classification algorithms we used in this project are:

– Logistic Regression  
– K Nearest Neighbors  
– Support Vector Classifier  
– Decision Tree  
– Random Forests

chapter 2

**Sentiment Analysis**

Sentiment Analysis or Opinion Mining refers to the use of NLP, text analysis and computational linguistics to determine subjective information or the emotional state of the writer/subject/topic. It is commonly used in reviews which save businesses a lot of time from manually reading comments.

**2.1 Data Preprocessing**

You cannot go straight from raw text to fitting a machine learning or deep learning model  
You must clean your text first, which means splitting it into words and handling punctuation and case.  
In fact, there's a whole suit of text preparation methods that you mat need to use, and the choice of methods really depends on your NLP task.

This chapter has three sections, listed

**– Language**  
**– library**  
**– Steps**

**2.1.1 Language :**

We use **Python** ..why?  
The language is easy and widespread and has many libraries.

**2.1.2 library :**

**– re**  
**– pandas**  
**– Stopwords from nltk.corpus**  
**– word\_tokenize from nltk.tokenize**  
**– PorterStemmer from nltk.stem.porter**

**2.1.3 Steps**

**– Split into Words**

NLTK provides a function called word\_tokenize() for splitting strings into tokens (nominally words). It splits tokens based on white space and punctuation. For example, commas and periods are taken as separate tokens. Contractions are split apart (e.g. “What’s” becomes “What” “‘s“). Quotes are kept, and so on.

          splitintowords

**– Filter Out Punctuation**

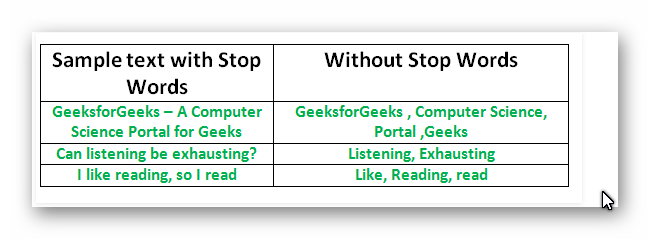
We can filter out all tokens that we are not interested in, such as all standalone punctuation.  
This can be done by iterating over all tokens and only keeping those tokens that are all alphabetic. Python has the function isalpha() that can be used. For example:

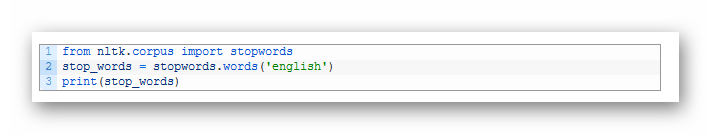
filteroutpunctuation  
  
  
 **– Filter out Stop Words (and Pipeline)**

The process of converting data to something a computer can understand is referred to as pre-processing. One of the major forms of pre-processing is to filter out useless data. In natural language processing, useless words (data), are referred to as stop words.

**– Stop Words:**

A stop word is a commonly used word (such as “the”, “a”, “an”, “in”) that a search engine has been programmed to ignore, both when indexing entries for searching and when retrieving them as the result of a search query.  
We would not want these words taking up space in our database, or taking up valuable processing time. For this, we can remove them easily, by storing a list of words that you consider to be stop words. NLTK(Natural Language Toolkit) in python has a list of stop words stored in 16 different languages. You can find them in the nltk\_data directory.





it print: ['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', 'your', 'yours', 'yourself', 'yourselves', 'he', 'him', 'his', 'himself', 'she', 'her', 'hers', 'herself', 'it', 'its', 'itself', 'they', 'them', 'their', 'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 'that', 'these', 'those', 'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had', 'having', 'do', 'does', 'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'until', 'while', 'of', 'at', 'by', 'for', 'with', ...]

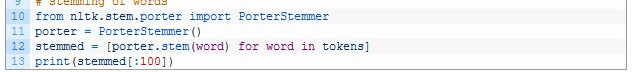
**– Normalizing Case**

– It is common to convert all words to one case.  
– This means that the vocabulary will shrink in size, but some distinctions are lost (e.g. “Apple” the company vs “apple” the fruit is a commonly used example).  
– We can convert all words to lowercase by calling the lower() function on each word.  
– For example:

normalize

**– Stem Words:**

– Stemming refers to the process of reducing each word to its root or base.  
– For example “fishing”, “fished”, and “fisher” all reduce to the stem “fish”.  
– Some applications, like document classification, may benefit from stemming in order to both reduce the vocabulary and to focus on the sense or sentiment of a document rather than deeper meaning.  
– There are many stemming algorithms, although a popular and long-standing method is the Porter Stemming algorithm.  
– This method is available in NLTK via the PorterStemmer class.  
– For example:



**– Count Vectorize:**

Dataset 'text' column can be viewed as a a big chunk of text separated by spaces and feed-back and carriage-returns '\r', '\n' and ' ', anything else is words and punctuation. when vectorizing text we simply create a big vector, at least big enough, with its size equal to different words in text. We here are considering words after all steps above are applied(splitting, removing punctuation, normalizing, and stemming). With this vector, every text is expressed as a vector with its length same as the big vector. where each element represent count of a word in this specific text.

from sklearn.feature\_extraction.text import CountVectorizer

v = CountVectorizer(analyzer = "word")

X\_train\_features = v.fit\_transform(X\_train)

**2.1.4 Summary**

All what has been discussed can above can be summarized with the full Data Preprcoessing snippet from source code.

df1 = pd.read\_csv('Full-Economic-News-DFE-839861.csv',encoding='ISO-8859-1')

df2 = pd.read\_csv('us-economic-newspaper.csv', encoding='ISO-8859-1')

df = pd.concat([df1, df2], sort=False)

df = df.sample(frac=1).reset\_index(drop=True)

# Data Preprocessing steps

# step 1; select columns "text", "positivity"

# step 2; remove rows with NaN positivity

# step 3; remove punctuation

# step 4; remove stop words

# step 5; stemming words, to get effictive part of word in training

# step 6; count vectorizer, convert articles to vecotrs of 0's - 1's

def tweet\_to\_words(raw\_tweet):

letters\_only = re.sub("[^a-zA-Z]", " ",raw\_tweet)

words = letters\_only.lower().split()

stops = set(stopwords.words("english"))

meaningful\_words = [w for w in words if not w in stops]

return(" ".join( meaningful\_words ))

Df = df[["positivity", "text"]].dropna()

Df["sentiment"] = 0

Df["words"] = ""

Df.sentiment = Df.positivity.apply(lambda x: 0 if x<5 else 1)

Df.words = Df.text.apply(lambda x: tweet\_to\_words(x))

del(Df['text'])

del(Df['positivity'])

train, test = train\_test\_split(Df,test\_size=0.2 ,random\_state=42)

train, val = train\_test\_split(train,test\_size=0.25 ,random\_state=42)

X\_train, y\_train = train.words, train.sentiment

X\_test, y\_test = test.words, test.sentiment

X\_val, y\_val = val.words, val.sentiment

def stemmming(x):

porter = PorterStemmer()

splitted = x.split(' ')

x = ' '.join([porter.stem(w) for w in splitted])

return x

X\_train = X\_train.apply(lambda x:stemmming(x))

X\_test = X\_test.apply(lambda x:stemmming(x))

X\_val = X\_val.apply(lambda x:stemmming(x))

v = CountVectorizer(analyzer = "word")

X\_train\_features = v.fit\_transform(X\_train)

X\_test\_features = v.transform(X\_test)

X\_val\_features = v.transform(X\_val)  
v = CountVectorizer(analyzer = "word")

X = pd.concat([X\_train, X\_test, X\_val])

X\_features = v.fit\_transform(X)

y = pd.concat([y\_train, y\_test, y\_val])

**2.2 Classifiers**

While dealing this dataset, we had to use more than one classifier. Mainly, because wee seek a model with best possible performance. And because the models we've trained at first weren't well enough for our purpose or possible to launch.

What classifiers we used, they're mentioned following and ordered chronological.

**Trained Classifiers**

─ Logistic Regression  
─ K-Nearest Neighbor  
─ Support Vector Classifier  
─ Decision Trees  
─ Random Forest

from sklearn.linear\_model import LogisticRegression

from sklearn.neighbors import KNeighborsClassifier

from sklearn.svm import SVC, LinearSVC

from sklearn.tree import DecisionTreeClassifier

from sklearn.ensemble import RandomForestClassifier

from sklearn.ensemble import GradientBoostingClassifier

from sklearn.metrics import accuracy\_score

from sklearn.model\_selection import validation\_curve

Every Classifier has what is called **‘Hyper-parameter’**. It is, simply, some parameters that a classifier use in its equations and processing, or learning. So, In addition to training more than one model, we need to train that same model again with some modification, which mostly, changes its performance, increasing or decreasing or even same result are all possible results. Modifying or changing hyper-parameter values is called Hyper-paramter *tuning*.   
  
Searching for best values for Hyper-parameters is, in some way, similar to searching for some value in, say array, or any other data structure. So, We may use some already implemented and ready-to-use algorithm, but there's also normal way for searching, which is, just like linear-search. Here we try every possible value for every Hyper-parameter and train this model using this value. But such method is too much power- and time-consuming, instead we try various values in a range this is known to give best results. It is, after all, known that Machine Learning Algorithms and Techniques are some form of research and there are already many researchers and papers for all known Machine Learning Algorithms.   
  
Here, I used the normal method to search and is used with all classifiers and it's fairly simple. Iterating over set of values in some common range of some parameter. Using **‘validation\_curve’** function which automatically train the model over given (X, y) dataset with possible variations of only one parameter, we can compare between all models.   
  
Following, we talk about the classifiers we used and what modifications we made.

**2.2.1 Logistic Regression**

Logistic Regression classifier has many hyper-parameters, and what arouse our interest, here, are **‘max\_iter’** and **‘C’**.  
Iterating over set of values of **‘max\_iter’**, we train models with varying value of another hyper-parameter, **‘C’**, or the regularization term.   
  
Following is the code, in Python, we used to train our Logistic Regression model.

def LogisticRegressionModel():

best\_acc = 0

c\_range = [.1, .3, 10, 30]#, 100]

# for c in ([.01,.03,.1,.3,10]):#,3,10,30,100

for itr in ([3,10,30,100,300,1000]):

classifier = LogisticRegression(solver='liblinear',max\_iter=itr)

prop = ' - itr ' + str(itr)

title = classifier.\_\_class\_\_.\_\_name\_\_ + prop

train\_scores, test\_scores = validation\_curve(

classifier, X\_features, y, 'C', c\_range, cv=10)

We trained our models, so we need to store performance for those trained models. Following, the code we use to store preformance values.

train\_scores\_mean = np.mean(train\_scores, axis=1)

train\_scores\_std = np.std(train\_scores, axis=1)

test\_scores\_mean = np.mean(test\_scores, axis=1)

test\_scores\_std = np.std(test\_scores, axis=1)

Now, we indeed have significant data that represent our performance for all models, but that data has little to no meaning, yet. We need to visualize this data to give some information about if this model is better than other models, or about some values of a hyper-parameters are better than other other values. Following, the code we used to plot, or visualize, our performance.

We use results from **‘validation\_curve’**, after the training is done, and it returns two parameters represent training, test accuracy respectively.  
Each parameter is essentially a list of results, each element represent result of training at some variation of variated parameter, which is here, **‘C’** and variations are **‘c\_range’** list.  
Mean and std for training and testing results, give fair results across all models trained.

plt.figure(figsize=(8,5))

plt.xlabel("C values")

plt.ylabel("Score")

plt.title('Accuracy = {0:.05f}'.format(100\*test\_scores\_mean.max()) + prop)

plt.grid()

lw = 2

plt.plot(c\_range, train\_scores\_mean, 'o-', color="r",

label="Training score", lw=lw)

plt.plot(c\_range, test\_scores\_mean, 'o-', color="g",

label="Test score", lw=lw)

plt.fill\_between(n\_range, train\_scores\_mean - train\_scores\_std,

train\_scores\_mean+train\_scores\_std, alpha=0.1, color="r", lw=lw)

plt.fill\_between(n\_range, test\_scores\_mean - test\_scores\_std,

test\_scores\_mean+test\_scores\_std, alpha=0.1, color="g", lw=lw)

plt.legend(loc="best")

best\_acc = max(best\_acc, test\_scores\_mean.max())

X-axis represents variations of variated parmeter, ‘C’. Y-axis represents performance on scale 0.0 – 1.0 , or 0 – 100%. In section 2.4, We see our results.

**2.2.2 K-Nearest Neighbor**

K-Nearest Neighbors Classifier, similarly, varying parameter ‘n\_neighbors’, over n\_range array. This parameter represents the number which is taken into consideration when deciding for some point if it is ‘near’ to another.  
Iterating over set of values while varying ‘max\_iter’, which represents maximum number the algorithm will train and change weights more and for better until convergence.   
  
Following is the code, in Python, we used to train our Logistic Regression model.

def KNeighborsClassifierModel():

classifier = KNeighborsClassifier(n\_neighbors=5, leaf\_size=3)

n\_range = range(3,7)

prop = ''

title = classifier.\_\_class\_\_.\_\_name\_\_ + prop

train\_scores, test\_scores = validation\_curve(

KNeighborsClassifier(), X\_train\_features, y\_train,

'n\_neighbors', n\_range, cv=10)

Again, plotting is useful when we need to know why accuracy isn’t what we expected, if it’s the case. And That’s because validation curve shows us how the model behaves through training and after many iterations across variations of some parameter, and that helps us to know if some solution may help or not. Because, we not only have model behavior at some state, but we know also how it’ll behave if we change some variable.

plt.figure(figsize=(8,5))

plt.xlabel("N-Neigbors")

plt.ylabel("Score")

plt.title('Accuracy = {0:.05f}'.format(100\*test\_scores\_mean.max()) + prop)

plt.grid()

lw = 2

plt.plot(n\_range, train\_scores\_mean, 'o-', color="r",

label="Training score", lw=lw)

plt.plot(n\_range, test\_scores\_mean, 'o-', color="g",

label="Test score", lw=lw)

plt.fill\_between(n\_range, train\_scores\_mean - train\_scores\_std,

train\_scores\_mean+train\_scores\_std, alpha=0.1, color="r", lw=lw)

plt.fill\_between(n\_range, test\_scores\_mean - test\_scores\_std,

test\_scores\_mean+test\_scores\_std, alpha=0.1, color="g", lw=lw)

plt.legend(loc="best")

Accuracy.append(test\_scores\_mean.max())

**2.2.3 Support Vector Classifier**

Support Vector Classifier, similarly, varying parameters ‘kernel’, ‘degree’ over range 2-5, and ‘C’ over c\_range array.  
‘C’ parameter represents the factor that affects Regularization term.  
‘kernel’ variable iterates over 4 different kernels. Kernel here refers to the ‘d’ variable iterates over a set of dimensions for the ‘polynomial’ kernel.  
‘degree’ is enabled when kernel is set as polynomial. This parameter refers to degree of features.

def SVCModel():

c\_range = [.001, .003, .01, .03, .1, .3, 1, 3, 10]

best\_acc = 0

for krnl in ['linear', 'rbf', 'poly', 'sigmoid']:

f = 0

for d in range(2,5):

classifier = SVC(kernel='rbf', gamma='scale', degree = d,

probability = True, max\_iter = 200)

if krnl != 'poly':

prop = ' - krnl ' + krnl

if f == 1:

continue

else:

pass

else:

prop = ' - krnl poly with d ' + str(d)

title = classifier.\_\_class\_\_.\_\_name\_\_ + prop

train\_scores, test\_scores = validation\_curve(

classifier, X\_train\_features, y\_train, 'C', c\_range, cv=3)

Variations of variated parameter, ‘C’, are stored in ‘c\_range’ list.  
Again, we here calculate mean and std for return parameter to get an accurate estimation.  
Accuracy for this classifier is maximum result across all iterations at different kernels and degrees for polynomial kernel.

plt.figure(figsize=(8,5))

plt.xlabel("C")

plt.ylabel("Score")

plt.title('Accuracy = {0:.05f}'.format(100\*test\_scores\_mean.max()) + prop)

plt.grid()

lw = 2

plt.plot(c\_range, train\_scores\_mean, 'o-', color="r",

label="Training score", lw=lw)

plt.plot(c\_range, test\_scores\_mean, 'o-', color="g",

label="Test score", lw=lw)

plt.fill\_between(n\_range, train\_scores\_mean - train\_scores\_std,

train\_scores\_mean+train\_scores\_std, alpha=0.1, color="r", lw=lw)

plt.fill\_between(n\_range, test\_scores\_mean - test\_scores\_std,

test\_scores\_mean+test\_scores\_std, alpha=0.1, color="g", lw=lw)

plt.legend(loc="best")

best\_acc = max(best\_acc, test\_scores\_mean.max())

f = 1

Accuracy.append(best\_acc)

X-axis represents variations of variated parameter, ‘C’. Y-axis represents scoring on scale 0.0 – 1.0 , or 0 – 100%.  
Foremost, What we may need from a validation curve, other than know about dataset training behavior, is to know how much this model is ‘good’. So, we may compare it to another model, or same one with different values for hyper parameters.

**2.2.4 Decision Trees**

Decision Tree Classifier, similarly, has hyper-parameters needs to be ‘tuned’.  
‘crit’ variable iterates over 2 different criterion, either ‘gini’ or ‘entropy’, and what they refer to is how the classifier will behave in calculations and decisions when splitting data set. Both are useful and have their respective situations where they do best.

def DecisionTreeClassifierModel():

s\_range = [2,3,4,5,6]

best\_acc = 0

for crit in ['gini', 'entropy']:

classifier = DecisionTreeClassifier(criterion = crit)

prop = ' - ' + crit

title = classifier.\_\_class\_\_.\_\_name\_\_ + prop

train\_scores, test\_scores = validation\_curve(

classifier, X\_train\_features, y\_train,

'min\_samples\_split', s\_range, cv=10)

Variations of variated parameter, ‘min\_samples\_split’, are stored in ‘s\_range’ list.  
Again, we here calculate mean and std for return parameter to get an accurate estimation.  
Accuracy for this classifier is maximum result between model based on ‘gini’ criterion or ‘entropy’.  
X-axis represents variations of variated parameter, ‘min\_samples\_split’. Y-axis represents scoring on scale 0.0 – 1.0 , or 0 – 100%.

plt.figure(figsize=(8,5))

plt.xlabel("sample split")

plt.ylabel("Score")

plt.title('Accuracy = {0:.05f}'.format(100\*test\_scores\_mean.max()) + prop)

plt.grid()

lw = 2

plt.plot(s\_range, train\_scores\_mean, 'o-', color="r",

label="Training score", lw=lw)

plt.plot(s\_range, test\_scores\_mean, 'o-', color="g",

label="Test score", lw=lw)

plt.fill\_between(n\_range, train\_scores\_mean - train\_scores\_std,

train\_scores\_mean+train\_scores\_std, alpha=0.1, color="r", lw=lw)

plt.fill\_between(n\_range, test\_scores\_mean - test\_scores\_std,

test\_scores\_mean+test\_scores\_std, alpha=0.1, color="g", lw=lw)

plt.legend(loc="best")

best\_acc = max(best\_acc, test\_scores\_mean.max())

Accuracy.append(best\_acc)

**2.2.5 Random Forest**

Random Forest Classifier, similarly, has hyper-parameters needs to be ‘tuned’.  
‘crit’ variable iterates over 2 different criterion, either ‘gini’ or ‘entropy’.  
‘n\_est’ variable iterates over a set of choices for ‘# neighbors’ what decision will be made based on.  
‘s\_range’ variable is a list represents variations for function to create validation curve.

def RandomForestClassifierModel():

best\_acc = 0

s\_range = [2,3,4,5,6]

for crit in ['gini', 'entropy']:

for n\_est in [5,10,25,100,130,300]:

print(crit, n\_est)

classifier = RandomForestClassifier(criterion=crit, n\_estimators=n\_est)

prop = ' - criterion = ' + crit + ' estimators = ' + str(n\_est)

title = classifier.\_\_class\_\_.\_\_name\_\_ + prop

train\_scores, test\_scores = validation\_curve(classifier,

X\_train\_features, y\_train, 'min\_samples\_split', s\_range, cv=10)

Variations of variated parameter, ‘min\_samples\_split’, are stored in ‘s\_range’ list.  
Again, we here calculate mean and std for return parameter to get an accurate estimation. Accuracy for this classifier is maximum result across models based on ‘gini’ criterion or ‘entropy’, and #neighbors as ‘n\_est’ variable.

X-axis represents variations of variated parameter, ‘min\_samples\_split’. Y-axis represents scoring on scale 0.0 – 1.0 , or 0 – 100%.

plt.figure(figsize=(8,5))

plt.xlabel("sample split")

plt.ylabel("Score")

plt.title('Accuracy = {0:.05f}'.format(100\*test\_scores\_mean.max()) + prop)

plt.grid()

lw = 2

plt.plot(s\_range, train\_scores\_mean, 'o-', color="r",

label="Training score", lw=lw)

plt.plot(s\_range, test\_scores\_mean, 'o-', color="g",

label="Test score", lw=lw)

plt.fill\_between(n\_range, train\_scores\_mean - train\_scores\_std,

train\_scores\_mean+train\_scores\_std, alpha=0.1, color="r", lw=lw)

plt.fill\_between(n\_range, test\_scores\_mean - test\_scores\_std,

test\_scores\_mean+test\_scores\_std, alpha=0.1, color="g", lw=lw)

plt.legend(loc="best")

best\_acc = max(best\_acc, test\_scores\_mean.max())

Accuracy.append(best\_acc)

**2.3 Data Visualization**

Following figures are training results, specifically Validation Curves and are called plots.  
Every group of plots are enlisted under some classifier, and each plot represents some variations as mentioned below it. 

**2.3.1 Logistic Regression Classifier**

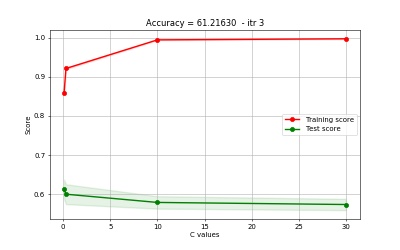


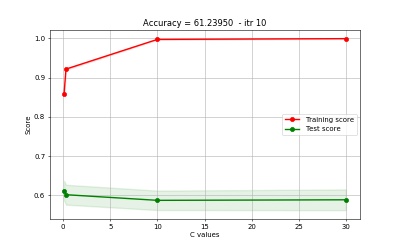
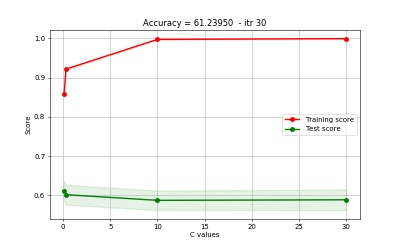
                                Figure 2.3.1.1 Logistic Regressin - maxItr 3  
 

                                Figure 2.3.1.2 Logistic Regressin - maxItr 10

                            
       Figure 2.3.1.3 Logistic Regressin - maxItr 30

**2.3.2 K-Nearest Neighbor Classifier**

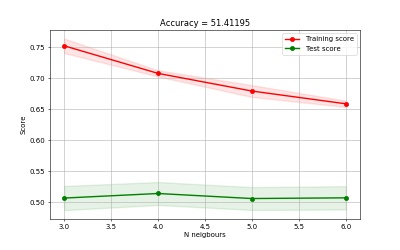


                                Figure 2.3.2.1 K-Nearest Neighbour - maxItr 30

**2.3.3 Suppor Vector Classifier(SVC)**

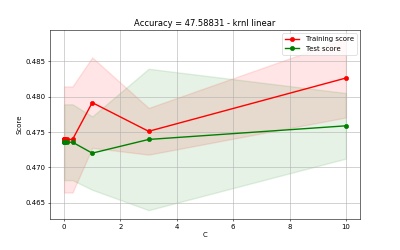


                                Figure 2.3.3.1 SVC with Linear kernel

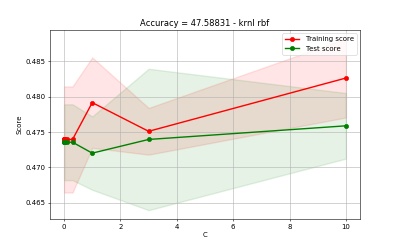


                                Figure 2.3.3.2 SVC with Radial Basis function kernel

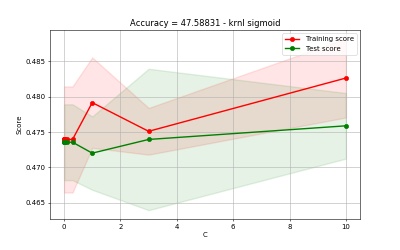


                                Figure 2.3.3.3 SVC with sigmoid\/gasussian kernel

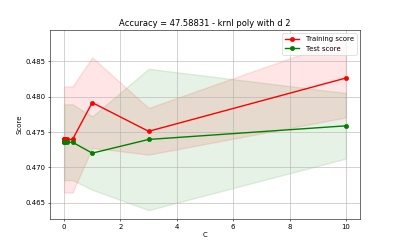


                                Figure 2.3.3.4 SVC with Polynomial kernel - degree 2

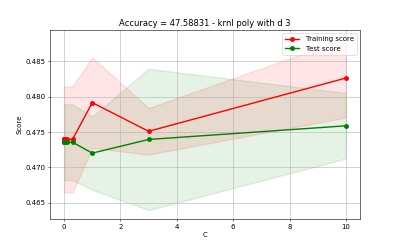


                                Figure 2.3.3.5 SVC with Polynomial kernel - degree 3

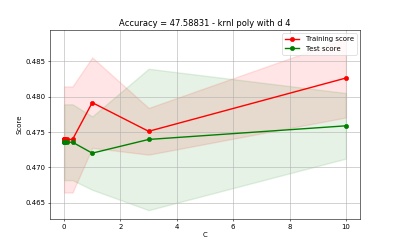


                                Figure 2.3.3.6 SVC with Polynomial kernel - degree 4

**2.3.4 Decision Tree Classifier**

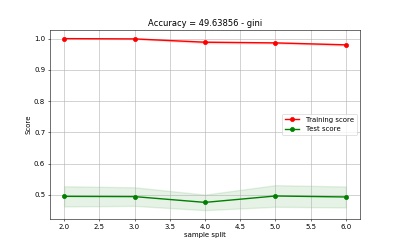


                                Figure 2.3.4.1 Decision Tree Classifier using gini

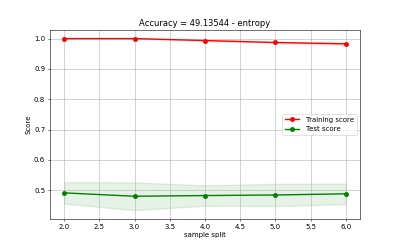


                                Figure 2.3.4.2 Decision Tree Classifier using entrop

**2.3.5 Random Forest**

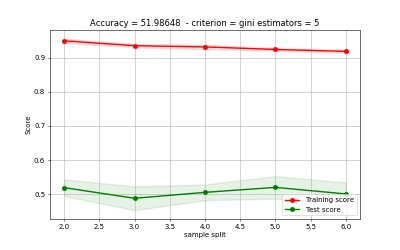


                                Figure 2.3.5.1 Random Forest Classifier  
                                                    using gini - using 5 estimators

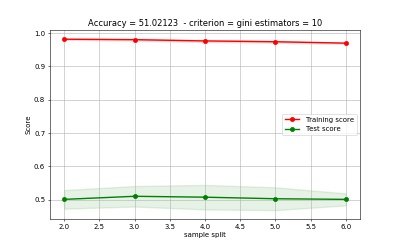


                                Figure 2.3.5.2 Random Forest Classifier  
                                                    using gini - using 10 estimators

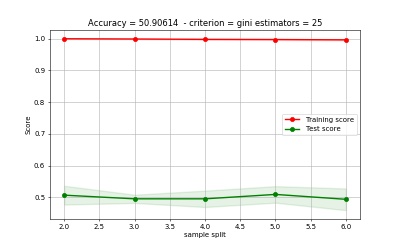


                                Figure 2.3.5.3 Random Forest Classifier  
                                                    using gini - using 25 estimators

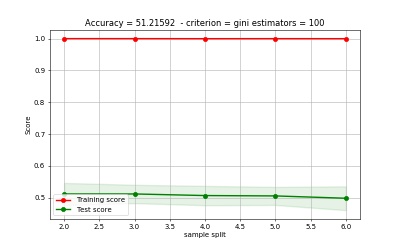


                                Figure 2.3.5.4 Random Forest Classifier  
                                                    using gini - using 100 estimators

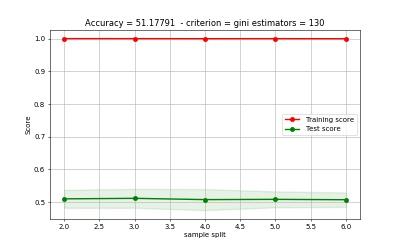


                                Figure 2.3.5.5 Random Forest Classifier  
                                                    using gini - using 130 estimators

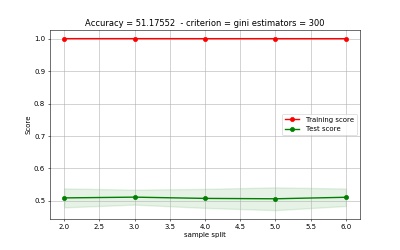


                                Figure 2.3.5.6 Random Forest Classifier  
                                                    using gini - using 300 estimators

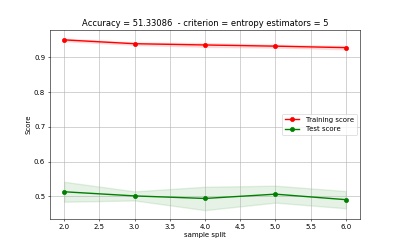


                                Figure 2.3.5.7 Random Forest Classifier  
                                                    using entropy - using 5 estimators

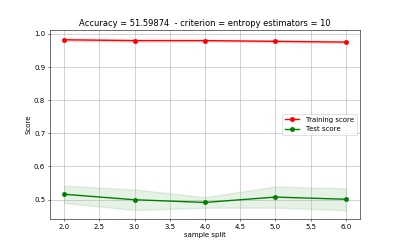


                                Figure 2.3.5.8 Random Forest Classifier  
                                                    using entropy - using 10 estimators

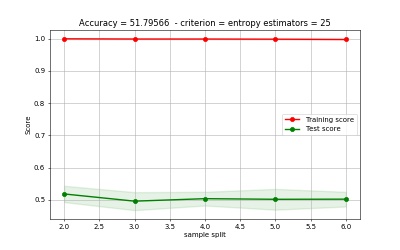


                                Figure 2.3.5.9 Random Forest Classifier  
                                                    using entropy - using 25 estimators

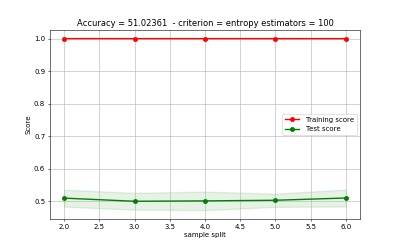


                                Figure 2.3.5.10 Random Forest Classifier  
                                                    using entropy - using 100 estimators

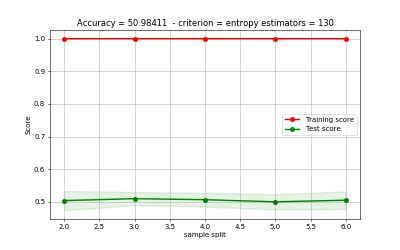


                                Figure 2.3.5.11 Random Forest Classifier  
                                                    using entropy - using 130 estimators

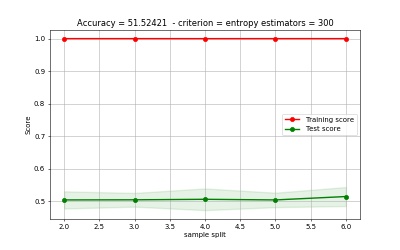


                                Figure 2.3.5.12 Random Forest Classifier  
                                                    using entropy - using 300 estimators

**2.4 Conclusion**

**So far, our best model has accuracy of about 61.23%**  
              **Logistic Regression model, for this dataset, wins.**

**Now, we’ve seen the data, knew about it and we even made a model that trains this data and generalizes how it deal with other articles or sentences and give a correct sentiment if it’s positive or negative.**

**We may do one thing more, maybe we classify it as multiclassification [negative, neutral and positive], and such classification exists and is already common.**

**We also need to look into the data, see what words were the most important and the classifier give most attention, this way we may choose less features and prevent overfitting, which is a thing our model suffer due to insufficient data.**

**We also may train these models more on better CPU or using GPU, it’ll help us search on 2 parameters and get better validation curves, and have a wider range of variations to test our model on.**

**One thing worth mentioning is that there’re classifiers that are able to classify articles based on not only sentiment, but also subjective. It means that it can tell if this article relates to sport, economic, politics, medicine, etc. etc.**

Chapter 3

**Software Project Management Plan**

**3.1 Introduction**

**3.1.1 Project Overview**

**Motivation:**

Number of online articles is huge and increases over time. Sometimes users need positive/negative opinions about specific subject in economy so they spend time and effort for reading all available articles online to know the positivity/negativity about it.

**Project Objective:**

We build this project to facilitate the process of articles revisions and spot a critical opinion about each one.

**3.1.2 Project Deliverables**

Articles Blog is a website for:

– Writing and Analyzing articles about Economy.  
– Finding positive / negative classified articles based on Natural language processing.

**3.2 Project Organization**

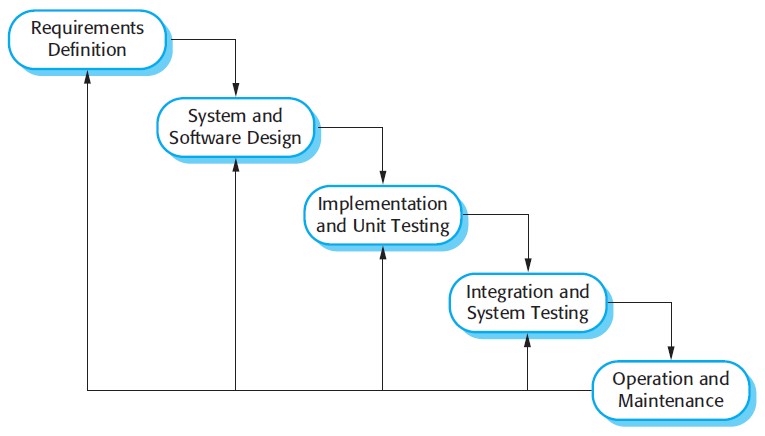
**3.2.1 Software Process Model**

A (software/system) process model is a description of the sequence of activities carried out in SE project, and the relative order of these activities.

It has many types such as: 

**Waterfall Model**

The waterfall model is a sequential approach, where each fundamental activity of a process represented as a separate phase, arranged in linear order. In the waterfall model, you must plan and schedule all of the activities before starting working on them (plan-driven process). The phases of the waterfall model are: Requirements, Design, Implementation, Testing, and Maintenance.

                                                  Figure 3.1 Waterfall Model 

**The Nature of Waterfall Phases**

In principle, the result of each phase is one or more documents that should be approved, and the next phase shouldn’t be started until the previous phase has completely been finished.

In practice, however, these phases overlap and feed information to each other. For example, during design, problems with requirements can be identified, and during coding, some of the design problems can be found, etc.

The software process therefore is not a simple linear but involves feedback from one phase to another. So, documents produced in each phase may then have to be modified to reflect the changes made.

**When to Use?**

In principle, the waterfall model should only be applied when requirements are well understood and unlikely to change radically during development as this model has a relatively rigid structure which makes it relatively hard to accommodate change when the process in underway.

**Advantages of waterfall model:**

– It allows for departmentalization and managerial control.  
– Simple and easy to understand and use.  
– Easy to manage due to the rigidity of the model – each phase has specific deliverables and a review process.  
– Phases are processed and completed one at a time.  
– Works well for smaller projects where requirements are very well understood.  
– A schedule can be set with deadlines for each stage of development and a product can proceed through the development process like a car in a car-wash, and theoretically, be delivered on time.

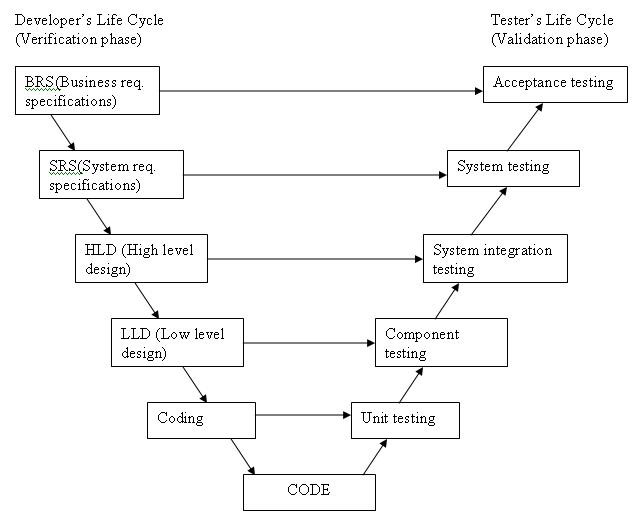
**Disadvantages of waterfall model:**

– It does not allow for much reflection or revision.  
– Once an application is in the testing stage, it is very difficult to go back and change something that was not well-thought out in the concept stage.  
– No working software is produced until late during the life cycle.

**V-Model**

V-Model means Verification and Validation model. Just like the waterfall model, the V-Shaped life cycle is a sequential path of execution of processes. Each phase must be completed before the next phase begins. V-Model is one of the many software development models.

Testing of the product is planned in parallel with a corresponding phase of development in V-model.

  
                                                         Figure 3.2 V-model 

**The various phases of the V-model are as follows:**

Requirements like BRS and SRS begin the life cycle model just like the waterfall model. But, in this model before development is started, a system test plan is created. The test plan focuses on meeting the functionality specified in the requirements gathering.

The high-level design (HLD) phase focuses on system architecture and design. It provides overview of solution, platform, system, product and service/process. An integration test plan is created in this phase as well to test the pieces of the software systems ability to work together.

The low-level design (LLD) phase is where the actual software components are designed. It defines the actual logic for each component of the system. Class diagram with all the methods and relation between classes comes under LLD. Component test are created in this phase as well.

The implementation phase is, again, where all coding takes place. Once coding is complete, the path of execution continues up the right side of the V where the test plans developed earlier are now use. Coding: This is at the bottom of the V-Shape model. Module design is converted into code by developers. Unit testing is performed by the developers on the code written by them.

**When to use the V-model:**

– The V-shaped model should be used for small to medium sized projects where requirements are clearly defined and fixed.  
– The V-Shaped model should be chosen when ample technical resources are available with needed technical expertise.  
– High confidence of customer is required for choosing the V-Shaped model approach. Since, no prototypes are produced, there is a very high risk involved in meeting customer expectations.

**Advantages of V-model:**

– Simple and easy to use.  
– Testing activities like planning, test designing happens well before coding. This saves a lot of time, hence higher chance of success over the waterfall model.  
– Proactive defect tracking – that is defects are found at early stage.  
– Avoids the downward flow of the defects.  
– Works well for small projects where requirements are easily understood.

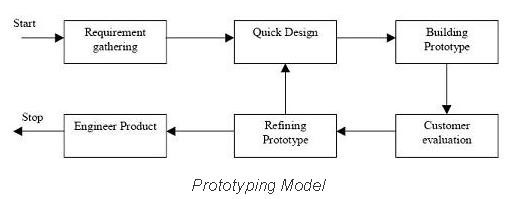
**Disadvantages of V-model:**

– Very rigid and least flexible.  
– Software is developed during the implementation phase, so no early prototypes of the software are produced.  
– If any changes happen in midway, then the test documents along with requirement documents must be updated.

**Prototype Model**

The basic idea in Prototype model is that instead of freezing the requirements before a design or coding can proceed, a throwaway prototype is built to understand the requirements. This prototype is developed based on the currently known requirements. Prototype model is a software development model. By using this prototype, the client can get an "actual feel" of the system, since the interactions with prototype can enable the client to better understand the requirements of the desired system. Prototyping is an attractive idea for complicated and large systems for which there is no manual process or existing system to help determining the requirements The prototype are usually not complete systems and many of the details are not built in the prototype. The goal is to provide a system with overall functionality.

**Diagram of Prototype model:**

   
                                                        Figure 3.3 Prototype model

**Advantages of Prototype model:**

– Users are actively involved in the development.  
– Since in this methodology a working model of the system is provided, the users get a better understanding of the system being developed.  
– Errors can be detected much earlier.  
– Quicker user feedback is available leading to better solutions.  
– Missing functionality can be identified easily.  
– Confusing or difficult functions can be identified.

**Disadvantages of Prototype model:**

– Leads to implementing and then repairing way of building systems.  
– Practically, this methodology may increase the complexity of the system as scope of the system may expand beyond original plans.  
– Incomplete application may cause application not to be used as the full system was designed.  
– Incomplete or inadequate problem analysis.

**3.2.2 Tools and Techniques**

In this section, we mention the tools and techniques that we have used to implement our project like programming languages.

**Language**

We use Python. Why? The language is easy and widespread and has many libraries.

**Libraries**

– re  
– nltk  
– pandas  
– stopwords from nltk.corpus  
– word\_tokenize from nltk.tokenize  
– PorterStemmer from nltk.stem.porter

**Training Classifiers**

– Logistic Regression  
– K Nearest Neighbors  
– Support Vector Classifier  
– Decision Tree  
– Random Forests

**DataBases**

**MYSQL**

– MySQL is a very popular, open source DBMS  
– MySQL databases are relational  
– Officially pronounced “my Ess Que Ell” (not my sequel).  
– Handles very large databases;  
– very fast performance; reliable.  
– MySQL is compatible with standard SQL  
– Why are we using MySQL?  
– Free (much cheaper than Oracle!)  
– Each student can install MySQL locally.  
– Multi-user access to a number of databases offered  
– Easy to use Shell for creating tables, querying tables, etc.  
– Easy to use with Java JDBC  
– MySQL is frequently used by PHP and Perl  
– Commercial version of MySQL is also provided (including technical support)

**MONGODB**

– MySQL is a very popular, open source RDBMS  
– MySQL databases are NON relational  
– Handles very large databases;  
– very fast performance; reliable.  
– Why are we using MongoDB?  
– Free  
– Each student can install it locally.  
– Multi-user access to a number of databases offered  
– Easy to use Shell for creating collections, querying collections, etc.  
– MONGODB is frequently used by PHP and Perl

**BackEnd (Django PlatForm)**

It is python based platform for web development

**Loose coupling**

A fundamental goal of Django’s stack is loose coupling and tight cohesion. The various layers of the framework shouldn’t “know” about each other unless absolutely necessary.

For example, the template system knows nothing about Web requests, the database layer knows nothing about data display and the view system doesn’t care which template system a programmer uses. Although Django comes with a full stack for convenience, the pieces of the stack are independent of another wherever possible.

**Less code**

Django apps should use as little code as possible; they should lack boilerplate. Django should take full advantage of Python’s dynamic capabilities, such as introspection.

**Quick development**

The point of a Web framework in the 21st century is to make the tedious aspects of Web development fast. Django should allow for incredibly quick Web development.

**Don’t repeat yourself (DRY)**

Every distinct concept and/or piece of data should live in one, and only one, place. Redundancy is bad. Normalization is good. The framework, within reason, should deduce as much as possible from as little as possible.

**Explicit is better than implicit**

This is a core Python principle listed in PEP 20, and it means Django shouldn’t do too much “magic.” Magic shouldn’t happen unless there’s a really good reason for it. Magic is worth using only if it creates a huge convenience unattainable in other ways, and it isn’t implemented in a way that confuses developers who are trying to learn how to use the feature.

**Consistency**

The framework should be consistent at all levels. Consistency applies to everything from low-level (the Python coding style used) to high-level (the “experience” of using Django).

**3.3 Project Management Plans**

**Dependencies and Constraints:**

We were depending on the economic online newspaper articles between 1951-2014ad As there was a constraint that the used data must be real to build an accurate system model with a good performance.

**Risks and Contingencies:**

There were some risks as there was no suitable machine/hardware for data training and data scarcity.

Chapter 4

**Analysis & Design**

4.1 Design

4.1.1 ERD

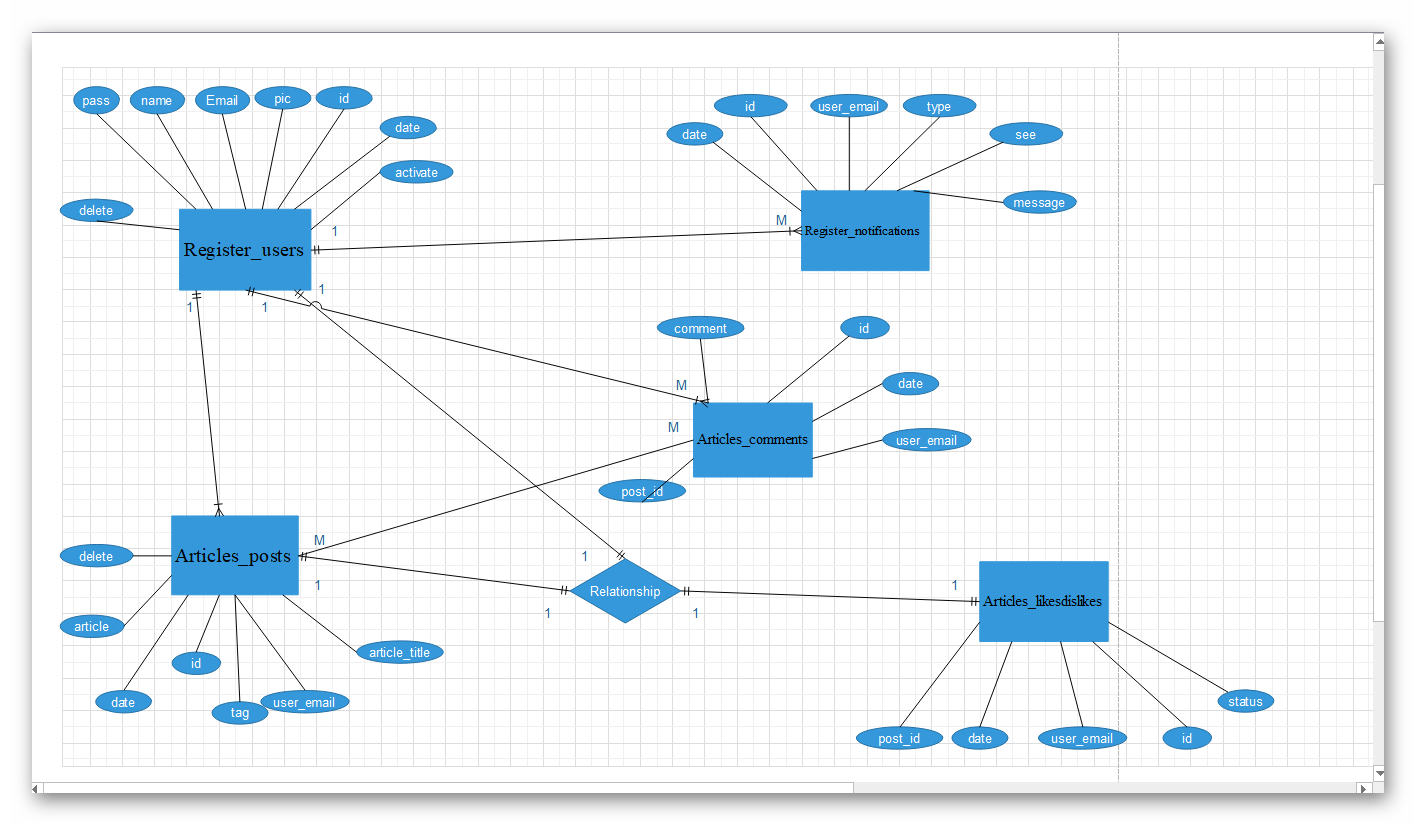
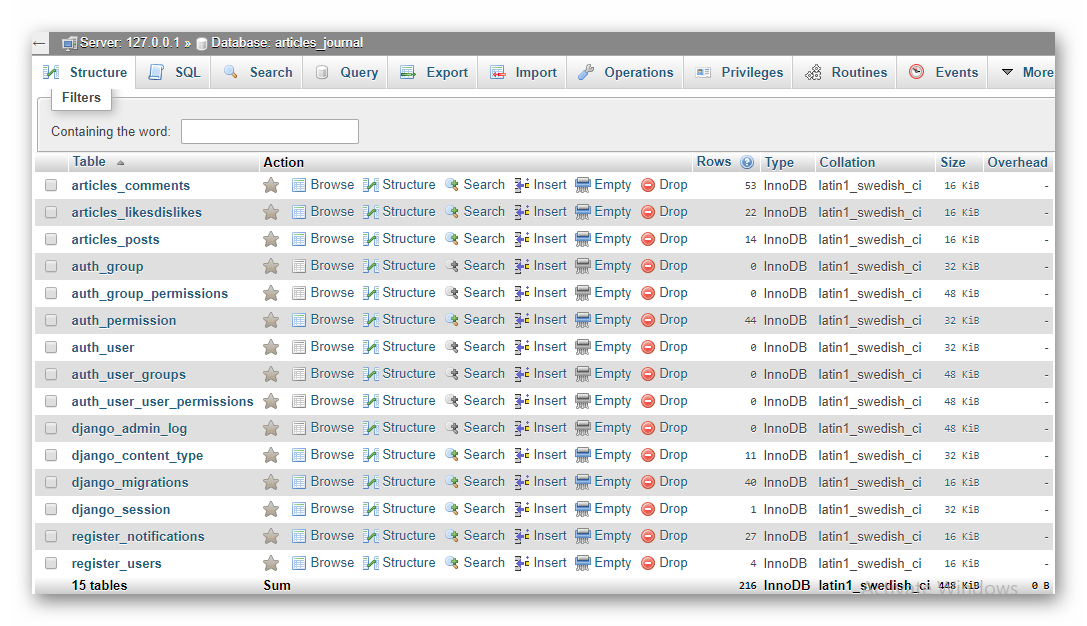
****

Figure 4.0 ERD

4.1.2 Database tables

1. Database Name : “articles\_journal”

Command :”CREATE DATABASE IS NOT EXISTS articles\_journal`”;

 Figure 4.1 Database articles\_journal

2. Tables names and discerption

1. Register\_users:-  
   Discerption :here, we store users data

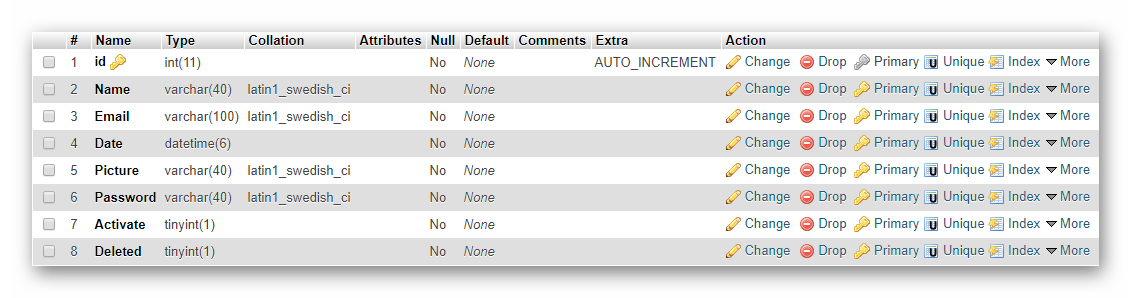


Figure 4.2 Register\_Users

Command:

“CREATE TABLE IF NOT EXISTS `register\_users`(  
`id` INT(11) NOT NULL AUTO\_INCREAMENT PRIMARY KEY,  
`Name` VARCHAR(400) NOT NULL,  
`Email` VARCHAR(100) NOT NULL,  
`Date` DATETIME(6) NOT NULL,  
`Picture` VARCHAR(40) NOT NULL,  
`Password` VARCHAR(40) NOT NULL,  
`Activate` TINYINT(1) NOT NULL,  
`Deleted` TINYINT(1) NOT NULL,  
)ENGINE=InnoDB DEFAULT CHARSET=utf8 AUTO\_INCREAMENT=1;

1. Register\_notifications  
    Discerption : here, we store users notifications

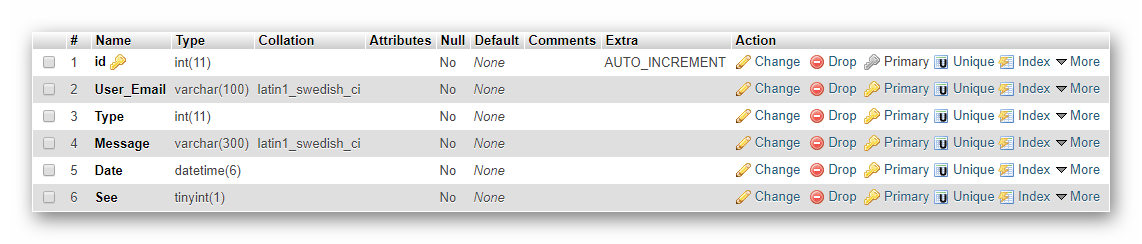


Figure 4.3 Register\_notifications  
  
Command:  
  
CREATE TABLE IF NOT EXISTS `register\_notifications`(

id` INT(11) NOT NULL AUTO\_INCREAMENT PRIMARYKEY,  
`User\_Email` VARCHAR(100) NOT NULL,  
`Type` INT(11) NOT NULL,  
`Message` VARCHAR(300) NOT NULL,  
`Date` DATETIME(6) NOT NULL,  
`See` TINYINT(1) NOT NULL,  
)ENGINE=InnoDB DEFAULT CHARSET=utf8 AUTO\_INCREAMENT=1;

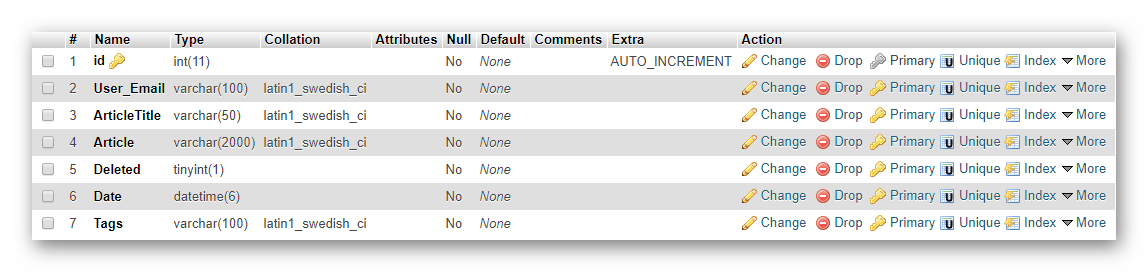
1. Articles\_posts  
    Discerption : here, we store users posts 

Figure 4.4 Articles\_posts

Command:

CREATE TABLE IF NOT EXISTS `register\_notifications`(  
`id` INT(11) NOT NULL AUTO\_INCREAMENT PRIMARY KEY,  
`User\_Email` VARCHAR(100) NOT NULL,  
`ArticleTitle` VARCHAR(50) NOT NULL,  
`Article` VARCHAR(2000) NOT NULL,  
`Deleted` TINYINT(1) NOT NULL,  
`Date` DATETIME(6) NOT NULL,  
`Tags` VARCHAR(100) NOT NULL,  
)ENGINE=InnoDB DEFAULT CHARSET=utf8  
AUTO\_INCREAMENT=1;

4. Articles\_likesdislikes  
 Discerption : here, we store users likes and dislikes in specific  
 articles.

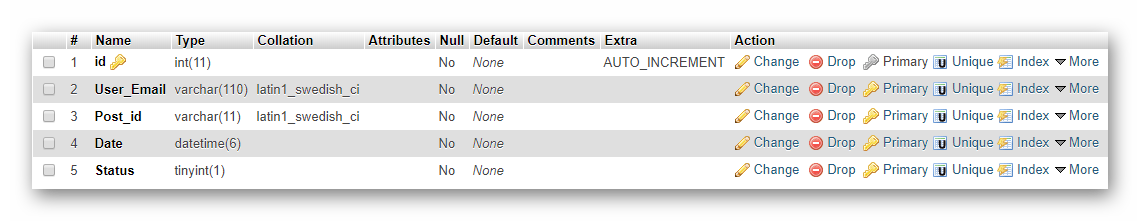


Figure 4.5 Articles\_likesdislikes

Command:  
  
CREATE TABLE IF NOT EXISTS `register\_notifications`(  
`id` INT(11) NOT NULL AUTO\_INCREAMENT PRIMARY KEY,  
`User\_Email` VARCHAR(100) NOT NULL,  
`Post\_id` VARCHAR(11) NOT NULL,  
`Date` DATETIME(6) NOT NULL,  
`Status` TINYINT(1) NOT NULL,  
)ENGINE=InnoDB DEFAULT CHARSET=utf8 AUTO\_INCREAMENT=1;

1. Articles\_comments  
    Discerption:- here, we store users comments in specific articles

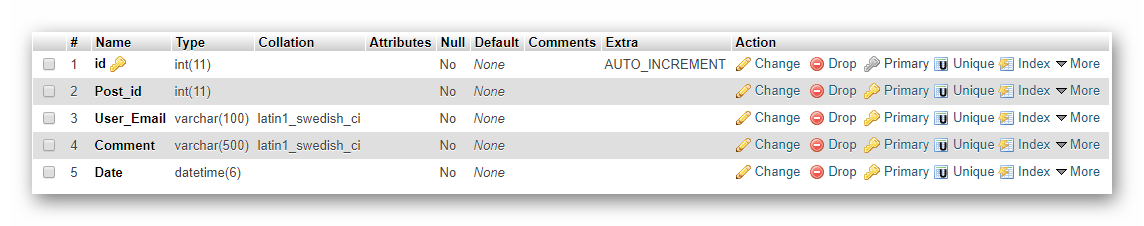


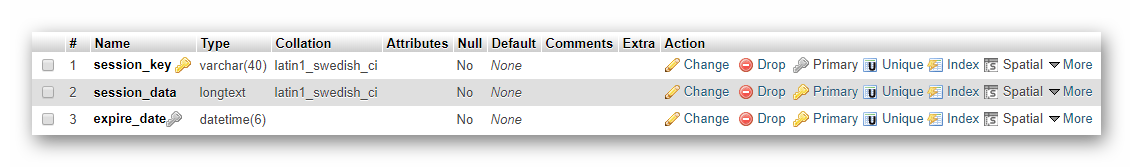
Figure 4.6 Articles\_comments

Command:

CREATE TABLE IF NOT EXISTS `register\_notifications`(

`id` INT(11) NOT NULL AUTO\_INCREAMENT PRIMARY KEY,   
`User\_Email` VARCHAR(100) NOT NULL,  
 `Post\_id` VARCHAR(11) NOT NULL,  
`Comment` VARCHAR(500) NOT NULL,  
`Date` DATETIME(6) NOT NULL,  
)ENGINE=InnoDB DEFAULT CHARSET=utf8 AUTO\_INCREAMENT=1;

1. Platform Tables

1. django\_session  
  
 Figure 4.7 Django\_session

2. django\_migrations

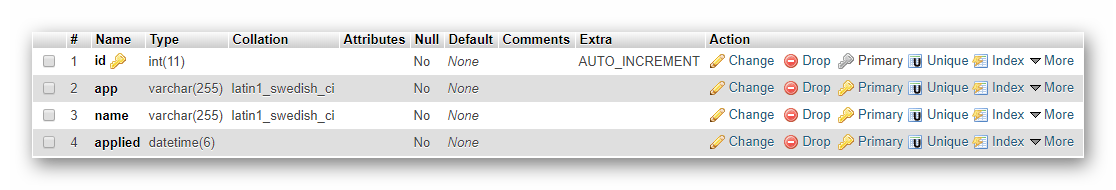


Figure 4.8 Django\_migrations

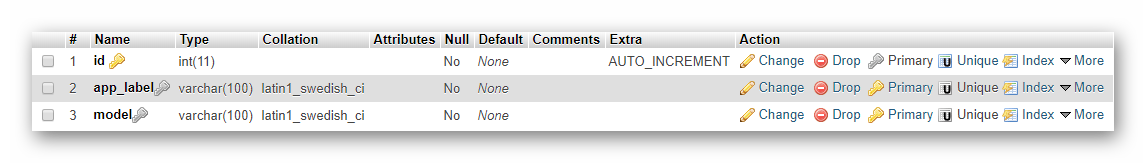
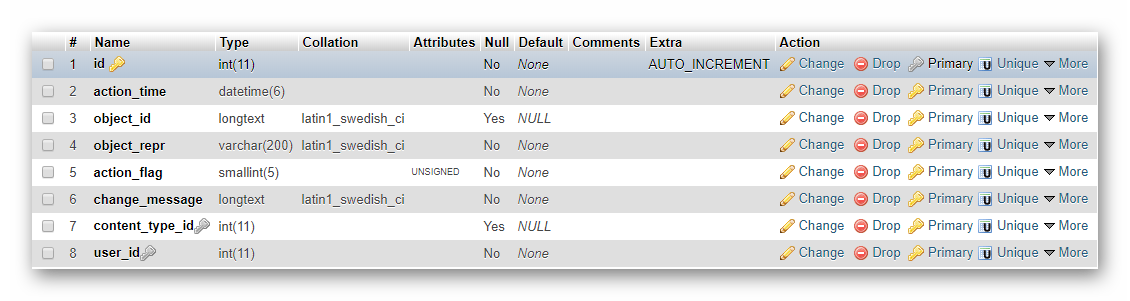
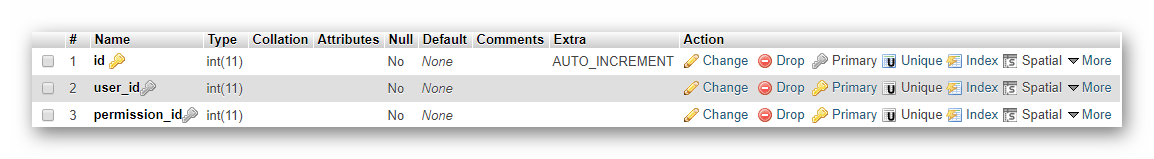
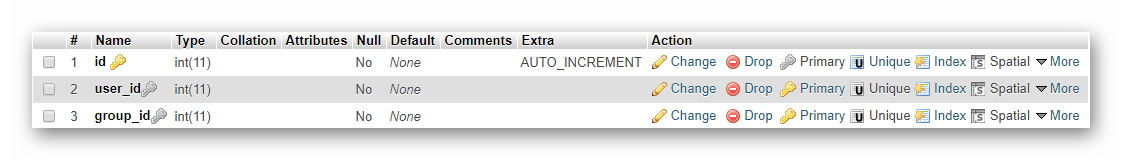
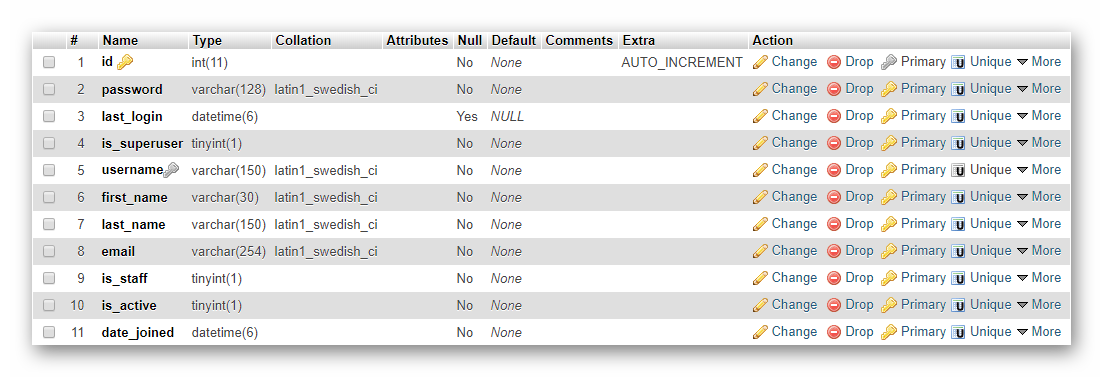
3. django\_content\_type  


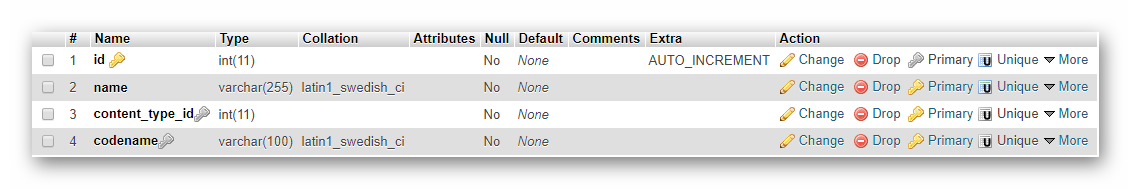
Figure 4.9 Django\_content\_type

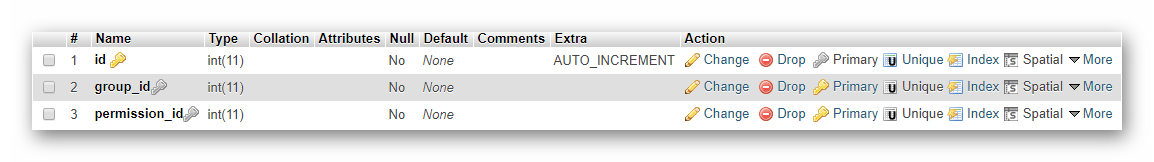
4. django\_admin\_log  
  
 Figure 4.10 Django\_admin\_log

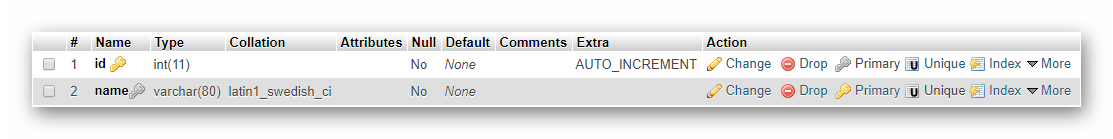
5. auth\_user\_user\_permissions  
  
 Figure 4.11 Auth\_user\_user\_permissions

6. auth\_user\_groups  
  
 Figure 4.12 Auth\_user\_groups

7. auth\_user  
  
 Figure 4.13 Auth\_user

8. auth\_permissions  
  
 Figure 4.14 Auth\_permissions

9. auth\_Group\_permissions  
  
 Figure 4.15 Auth\_Group\_permissions

10. auth\_group  
  
 Figure 4.16 Auth\_group

**4.2 Project Requirement**

1. Server To Put Project File in it

2. Browser to access website

**4.3 Project Values**

1. Can Write Articles in it

2. Can (Comment / Like / Dislike / Delete / Edit) Articles

3. Save time and effort for searching for specific opinion  
 (positive or negative) for specific subject so user don't need to  
 read all articles.

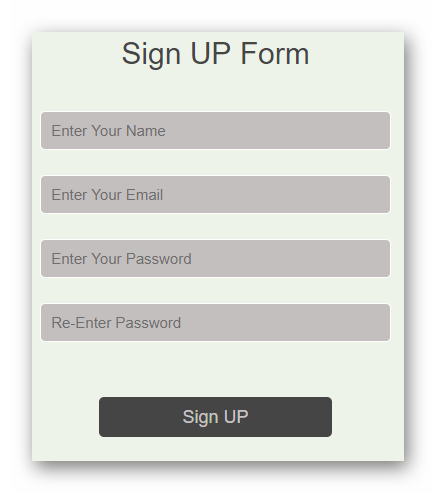
**4.4 Special Issues Or Constraints:**

1. User data is not deleted because it is used for machine  
 learning and training data for future.

2. Project must be done in one year

3. Project should be easy to use by users

Chapter 5  
 **Implementation**

5.1 User Manual  
  
 **5.1.1 Sign UP**   
 Figure 5.1 Sign UP  
Here user can signup To register in website by putting

**Name :**

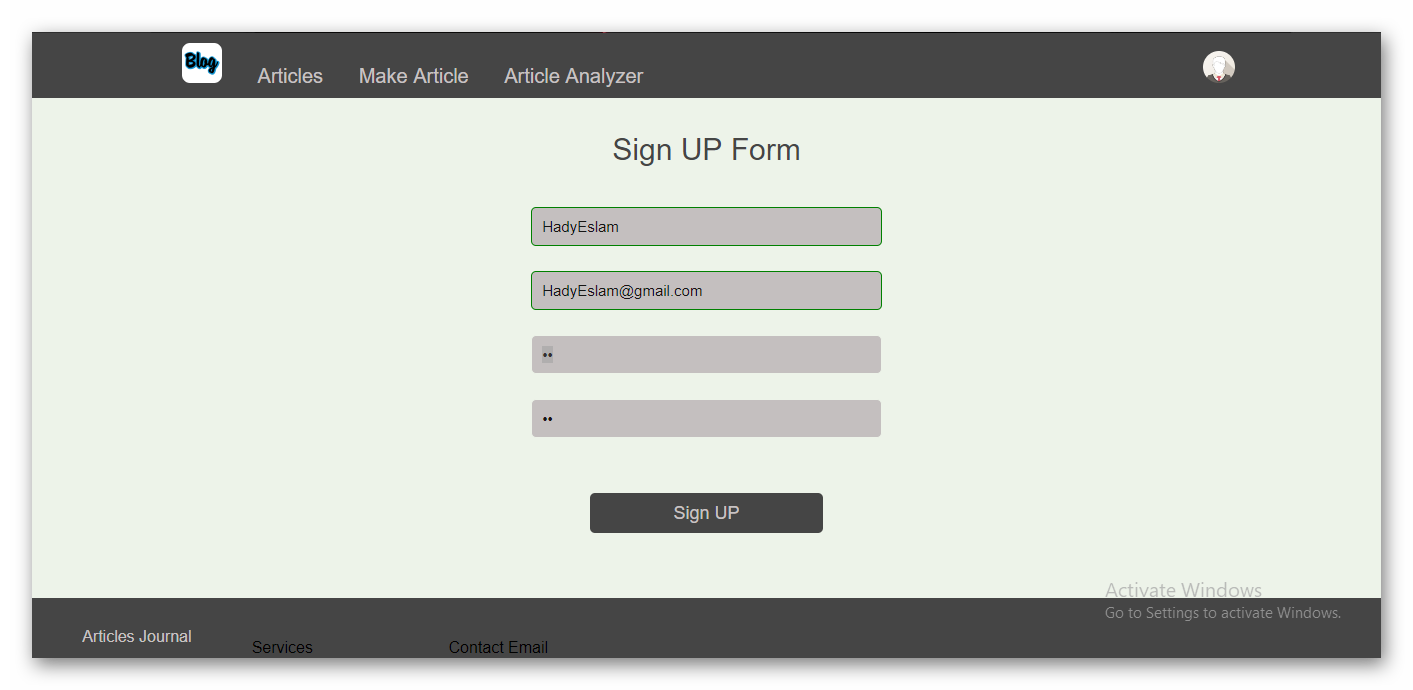
should be unique.  
should be 40 Characters only

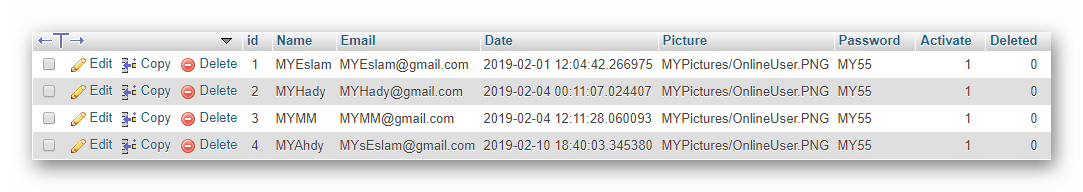
**Email :**

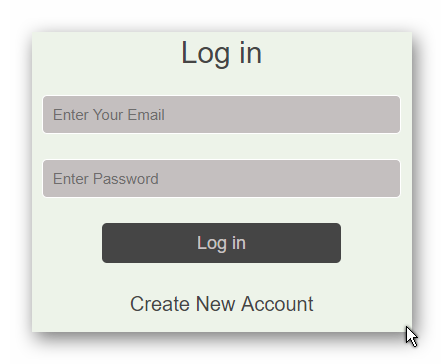
should be unique.  
should be 50 Characters only

**Password :**

should be 40 Characters only

  
Figure 5.2 Sign UP test

Figure 5.3 Sign UP form DB

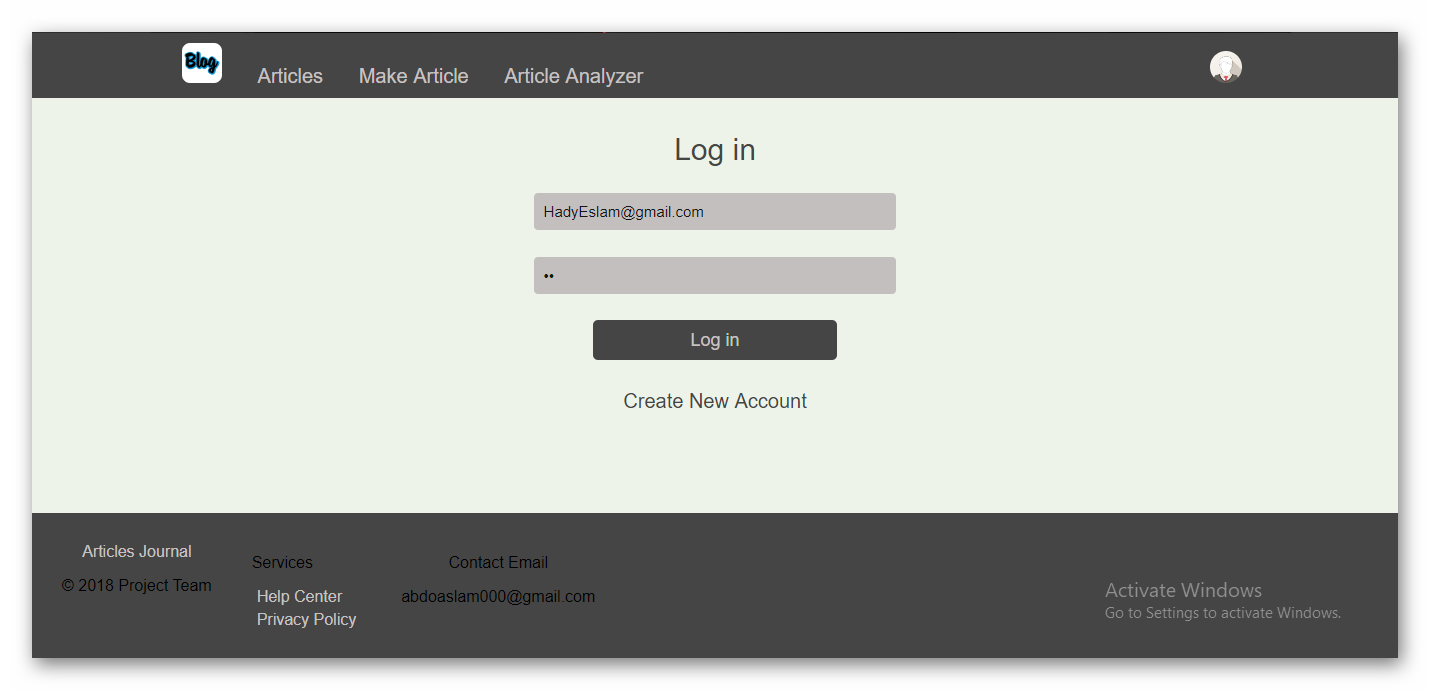
**5.1.2 Log in**  Figure 5.4 Log in form  
 Here user log in to website by putting:-

**Email :**

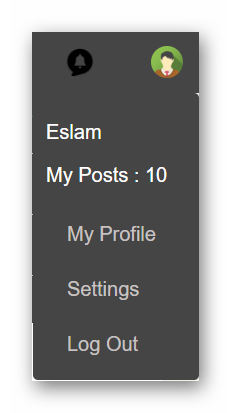
should be unique.  
should be 50 Characters only

**Password :**

should be 40 Characters only

 Figure 5.5 Log in test

**5.1.3 Log Out** “User Can Log Out”

  
 Figure 5.6 Drop down list   
**5.1.4 Make Articles** Figure 5.7 Make article area

Here registered Users can make Articles by clicking in *make Article* Link  


Figure 5.8 Header of page

Do these Steps to post an Article:-

**Article Title**

should not be more than 100 Characters

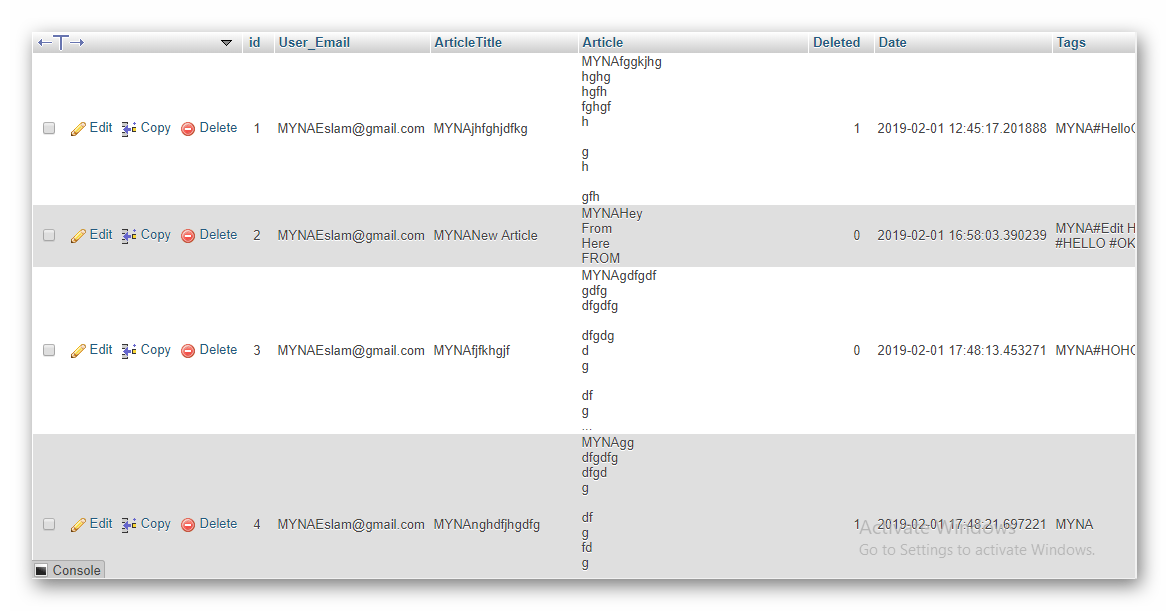
**Tags**

should Be no more than 500 Characters

**Article**

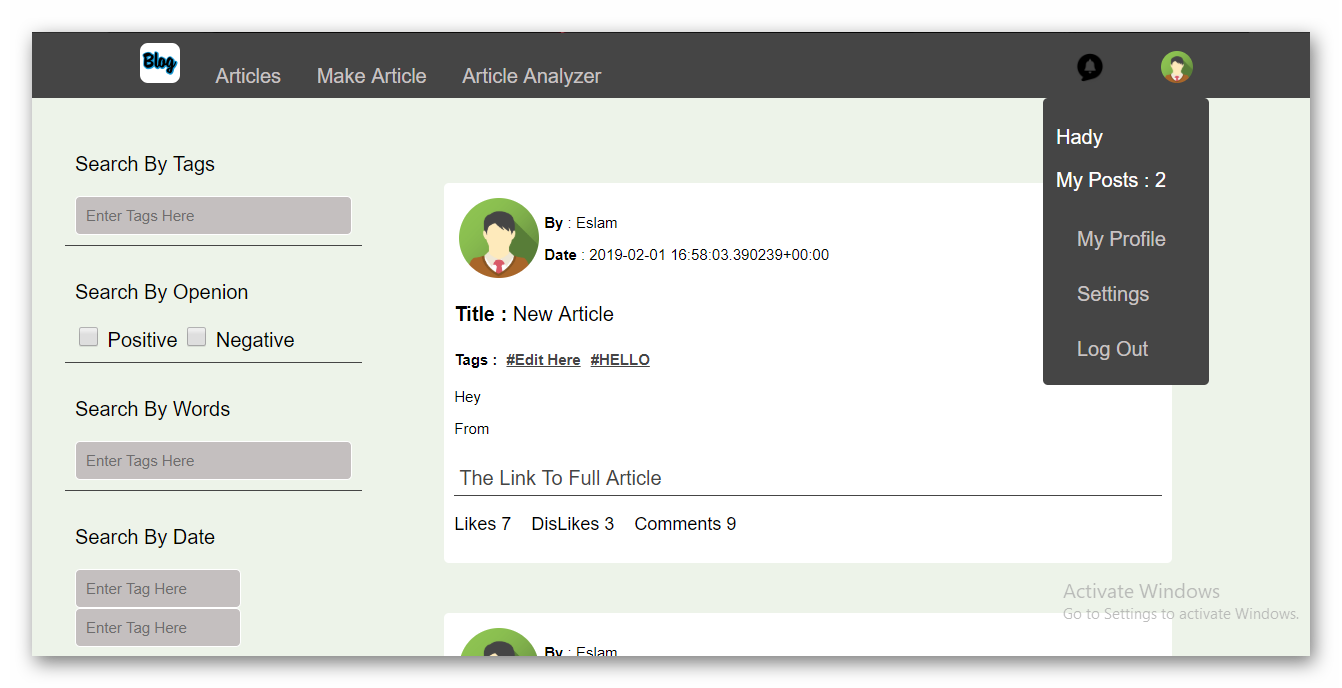
should Be no more than 2000 Characters

  
 Figure 5.9 Make article area test

  
 Figure 5.10 Article from DB

**5.1.5 Main Page**Here user can see all articles in website and search for articles to see Full Article click on Link [The Link To Full Article]

  
 Figure 5.11 Main Page

  
 Figure 5.12 Main Page2

**5.1.6 My Profile**

Here user can see his information’s and all of his articles and can do

**Edit Articles**

By clicking The Arrow in Post The Click Edit Link

**Delete Articles**

By clicking The Arrow in Post The Click Delete Link

  
 Figure 5.13 My Profile page

**5.1.7 User Profile**

Here, users can see other user's information’s and all his articles  
  
 Figure 5.14 UserProfile page

**5.1.8 Show Article**

Here, users can see the whole Article and can do:-

**like the Article:**

by clicking Like Button

**dislike the Article:**

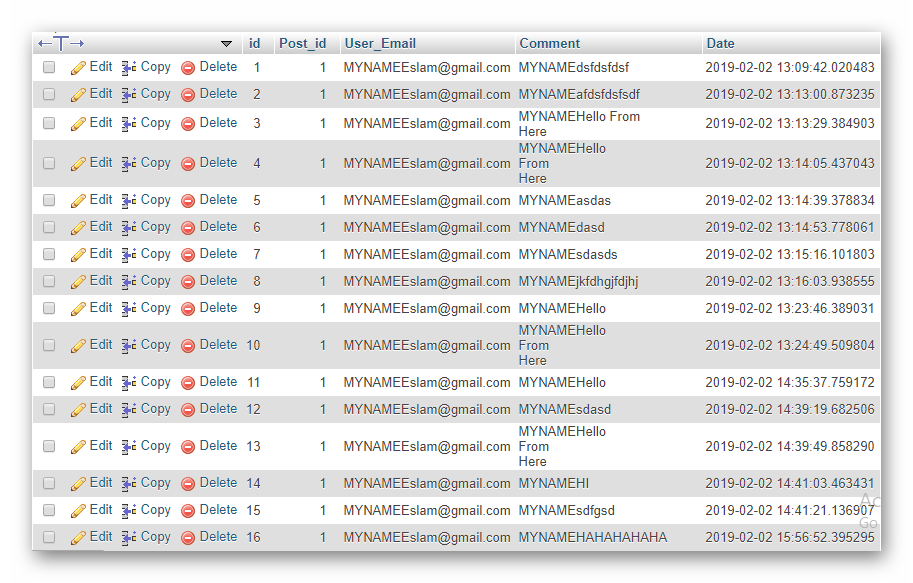
by clicking 'Dislike Button'

**Comment on the Article:**

by writing in Comment text area then clicking in Send Picture Beside it

User can only Like Or Dislike The Article Once And Comment Many Comments if he wants

  
 Figure 5.15 Show article page

  
 Figure 5.16 Comment test  
   
 Figure 5.17 Comment from DB

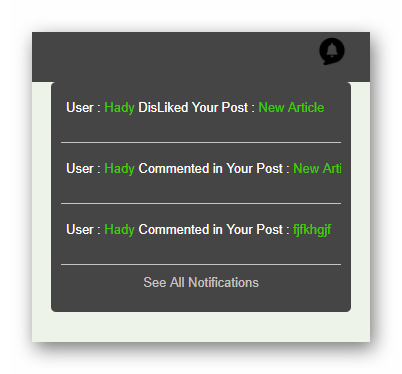
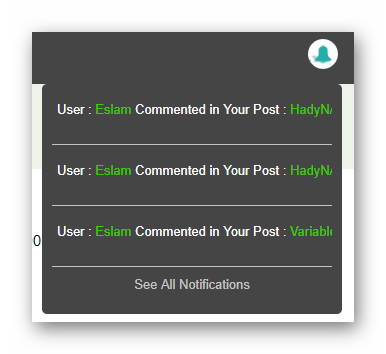
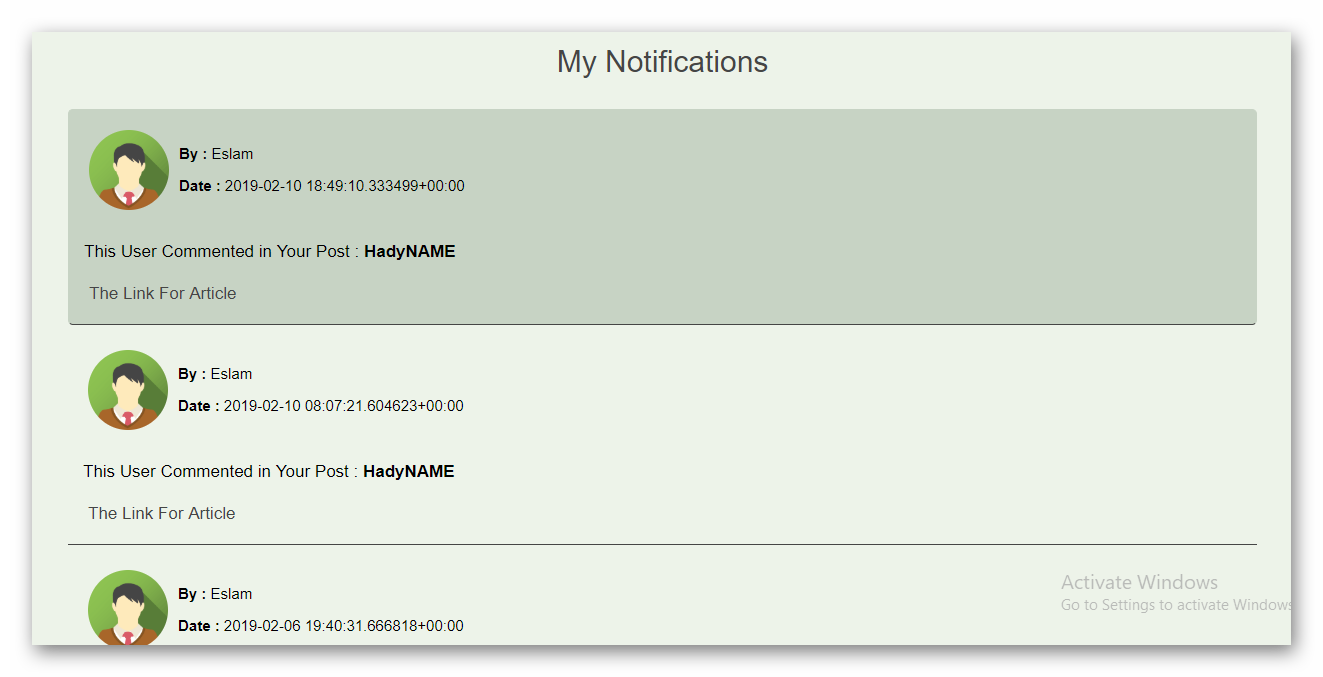
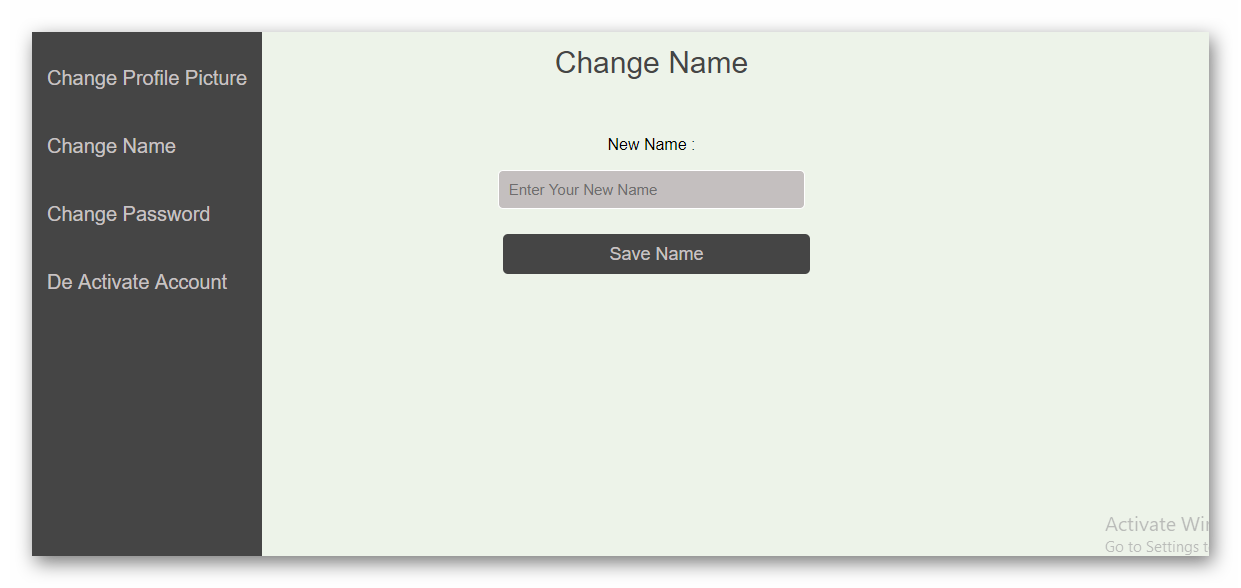
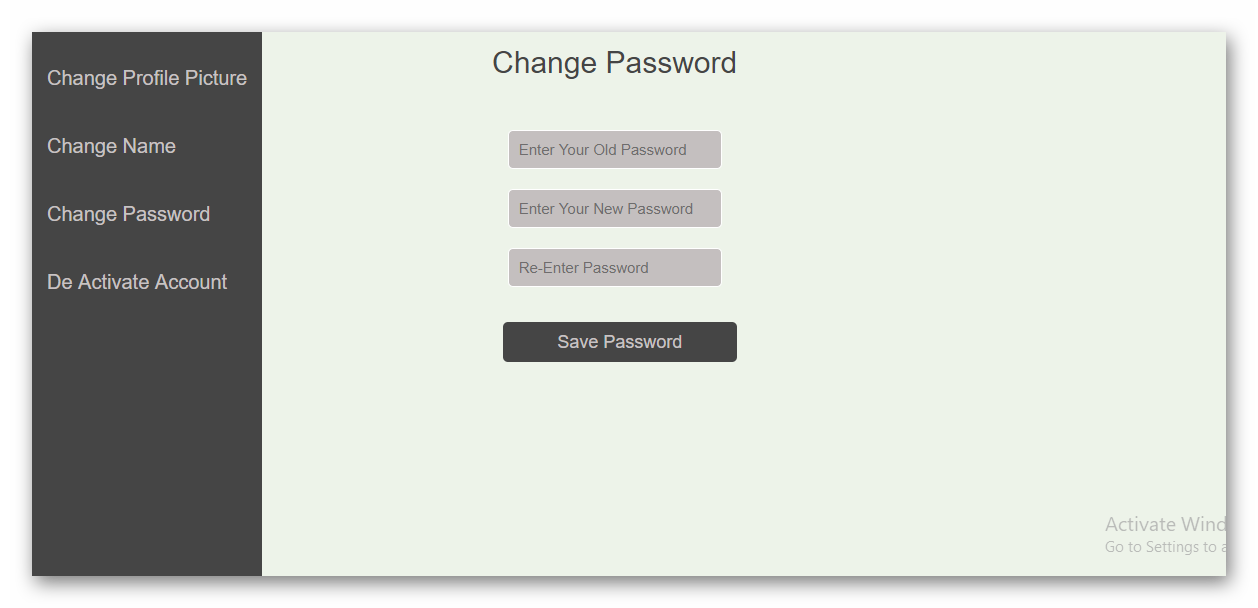
**5.1.9 My Notifications** Here User Can See All Of His Notifications  
   
 Figure 5.18 My Notifications list  
  
to see All Notifications Click in Link [See All Notifications]   
 Figure 5.19 My Notifications page

Figure 5.20 My Notifications from DB

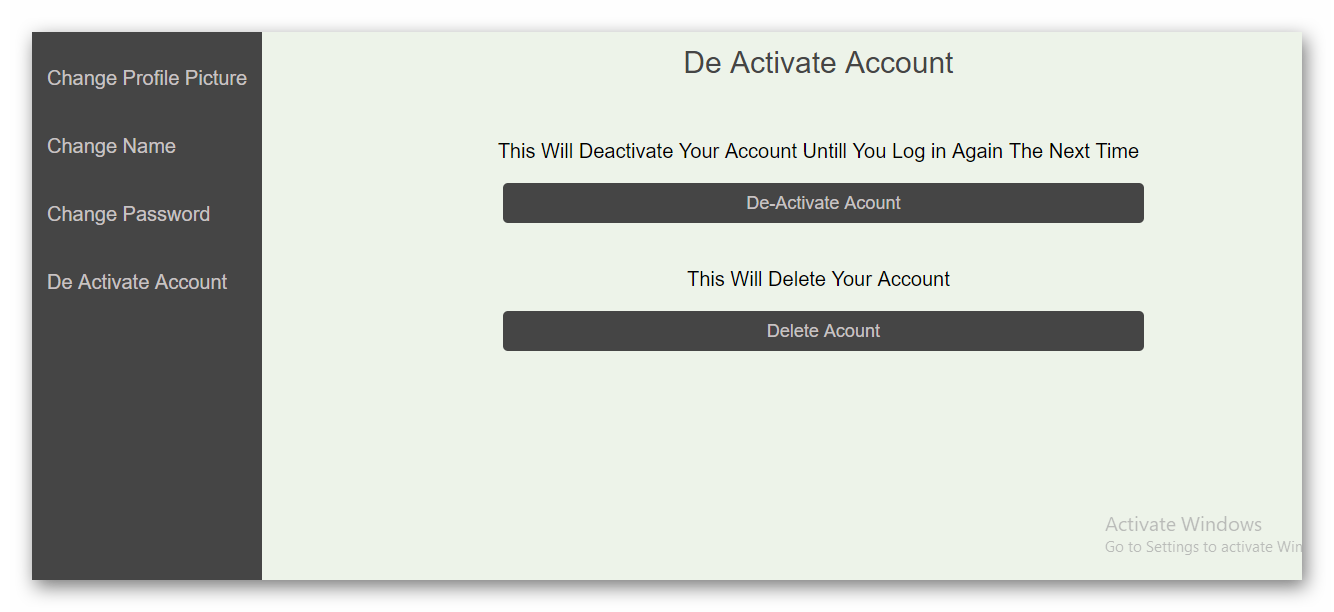
**5.1.10 Settings**Here, user can change his settings like by clicking [Settings] Link from drop down list  
He can change:-

1. **Name Setting**  
 Figure 5.21 Name setting  
 **2. Password Setting** Figure 5.22 Password setting

**3**. **Deactivate Account**  
 By clicking this button, user's profile and articles will not   
 be shown in website

**4. Delete Account**

By Clicking this button, user's Profile and articles will be deleted from website but it will not be actually deleted so it will be used by website For Machine Learning.

 Figure 5.23 Deactivate& Delete

**5.1.11 Help Center**

Here, user can get help in using Website

**5.1.12 Privacy Policy**

Here, user can know the policy of the website and the restriction in using website

**References**https://www.figure-eight.com/data-for-everyone/

https://www.w3schools.com/html/html\_intro.asp

https://www.w3schools.com/css/css\_intro.asp

https://www.w3schools.com/js/js\_intro.asp

https://www.w3schools.com/bootstrap/bootstrap\_get\_started.asp

https://www.w3schools.com/jquery/jquery\_intro.asp

https://machinelearningmastery.com/clean-text-machine-learning-python/

https://www.geeksforgeeks.org/python-stemming-words-with-nltk/

http://istqbexamcertification.com/what-are-the-software-development-models/