# CHAPTER 3
# SYSTEM DESIGN AND IMPLEMENTATION

## 3.1 System Design

The system design of this thesis is shown in Figure 3.1. The system will retrieve a collection of tweets from users. Text from user then preprocessed into vector data. Classification process will classify user's text into a labeled dataset. The results are predictions for each Big Five traits, personality Characteristics. System developed is a web application. The programming language used is Python with library Scikit-Learn.

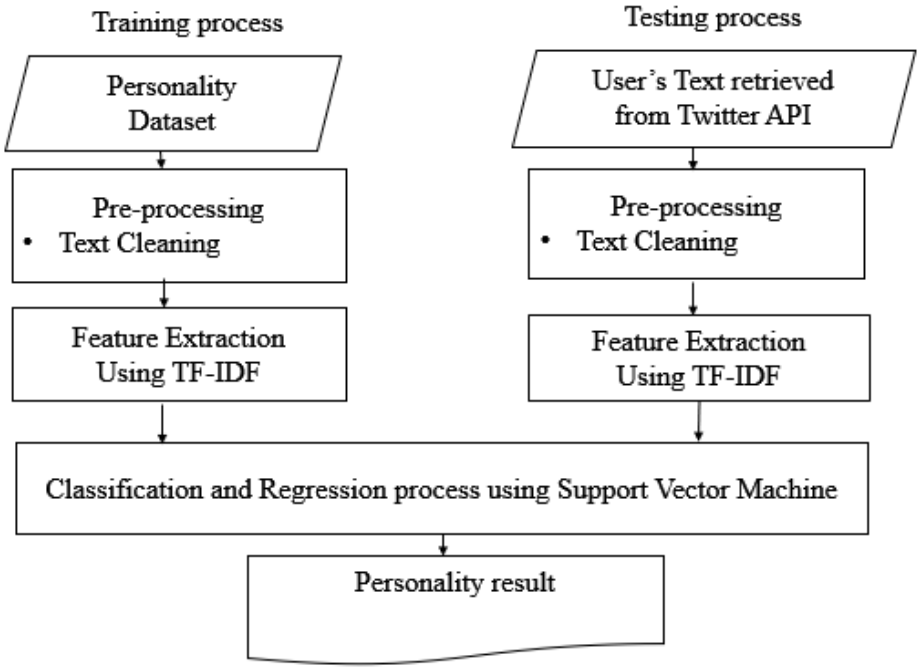



**Figure 3.1 System Design**

## 3.2 Methodology

This section describes and explain the methodology of the system design.

### 3.2.1 Data Collection

To train the model, the system uses MyPersonality dataset. MyPersonality dataset consists of 10.000 status updates from 250 users, which already labeled into Big Five personality dimensions. Original dataset then slightly modified. All posts from

single user ID appended into one long string which considered single document. Final dataset is in the form of 250 documents from 250 users.

To predict the Twitter user personality, user text is taken from collection tweets of a Twitter user. System will take last 1.000 texts in the form of tweets (post made directly by the user) and re-tweets (re-posting someone else's text). Collection of tweets from users is also made into a single document/one long string.

### 3.2.2 Preprocessing Text

In text classification, text data will be represented in vector space model. Textual data is likely to contain noise, errors, and less-informative words, which may lower down the quality of the model and harm its overall accuracy, and those issues are even more pronounced when dealing with data extracted from Social Networks, which presents noticeably higher rates of slang terms and spelling and grammar errors. It is then evident that an accurate preprocessing phase is required before exploiting the data to train the model. Text processing steps that we sequentially apply are the following:

- converting to *lower case*
- fix contraction (eg. ain't - am not)
- removing *mentions* (eg. @iamtedking)
- removing *urls* (eg. http://t.co/UCZzP9YpHk)
- removing *emoji*
- removing *punctuation* (eg. !"#$%&'()*)
- removing *digits* (eg. 123…)
- removing *stopwords* (eg. each, if, and, having, will…)

### 3.2.3 Feature Extraction or Transforming Text Documents into Vector Space

Text documents in their original form are not suitable to learn from. They must be transformed to match the learning algorithm's input format. Because most of the learning algorithms use the attribute-value representation, this means transforming it into a vector space.

First of all, documents need to be pre-processed. This usually means stopword filtering for omitting meaningless words (e.g. a, an, this, that), word stemming for reducing the number of distinct words, lowercase conversion etc. Then the transformation takes place. Each word will correspond to one dimension, identical

words to the same dimension. Let the word $w_i$ correspond to the $i^{th}$ dimension of the vector space. The most commonly used method is the so-called TF•IDF term-weighting method [1]. Denote TFIDF(i,j) the $i^{th}$ coordinate of the $j^{th}$ transformed document.

$$\mathrm{TFIDF}(i,j) = \mathrm{TF}(i,j) \cdot \mathrm{IDF}(i) \quad (3.1)$$

$$\mathrm{IDF}(i) = \log \frac{N}{\mathrm{DF}(i)} \quad (3.2)$$

Where TF($i$,$j$) means how many times does the $i^{th}$ word occur in the $j^{th}$ document, $N$ is the number of documents, and DF($i$) counts the documents containing the $i^{th}$ word at least once. The transformed documents together form the term-document matrix. It is desirable that documents of different length have the same length in the vector space, which is achieved with the so-called document normalization:

$$TFIDF'(i,j) = \frac{TFIDF(i,j)}{\sqrt[\beta]{\sum_i TFIDF(i,j)^\beta}} \quad (3.3)$$

The dimensionality of the vector space may be very high, which is disadvantageous in machine learning (complexity problems, overlearning), thus dimension reduction techniques are called for. Two possibilities exist, either selecting a subset of the original features, or merging some features into new ones, that is, computing new features as a function of the old ones.

### 3.2.4 Classification and Regression using Support Vector Machine

Support Vector Machine (SVM) is a supervised learning algorithm that analyzes the data and recognizing patterns used for both classification and regression. SVM take the set of training data and marking it as part of a category then predicts whether the test document is a member of an existing class. SVM models represent the data as a point in space divided by a line/hyperplane. SVM working is as follows:

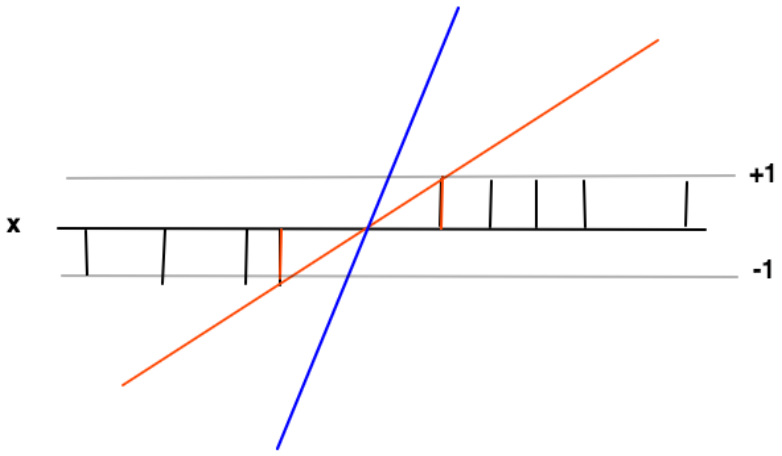
- Get text data and create vectors.
- Calculate weighted value of vectors.
- Get higher values vector and find value of the personality.
- Predict user personality type.

3.2.4.1 Classification

The personality classification case is multi-label classification. It means one person can have more than one personality trait or do not have dominant personality trait at all. Multi-label method used in this experiment is a binary relevance that is transforming each label in binary with independent assumption. The solution to this problem is to create a classifier for each label and train classifier based on the data that has been transformed. Each classifier is a binary classifier which will give output if the test document is a member of the label or not.

In this case the goal is to find a function $f(x) = wx + b$ where $f(x) \geq 1$ for positive examples and $f(x) \leq -1$ for negative examples. Under these conditions we want to maximize the margin (distance between the 2 red bars) which is nothing more than minimizing the derivative of $f' = w$.

The intuition behind maximizing the margin is that this will give us a unique solution to the problem of finding $f(x)$ (i.e. we discard for example the blue line) *and* also that this solution is the most general under these conditions, i.e. it acts as a regularization. This can be seen as, around the decision boundary (where red and black lines cross) the classification uncertainty is the biggest and choosing the lowest value for $f(x)$ in this region will yield the most general solution.

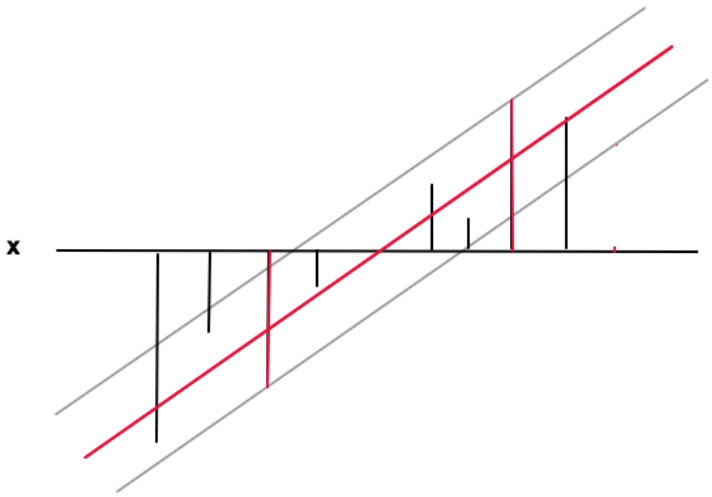



The data points at the 2 red bars are the support vectors in this case, they correspond to the non-zero Lagrange multipliers of the equality part of the inequality conditions $f(x) \geq 1$ and $f(x) \leq -1$

3.2.4.2 Regression

Regression is the task of predicting a continuous, real valued output from a set of predictors. As the name implies, univariate regression refers to estimating a regression model with one dependent variable (one outcome), while multivariate regression refers to building a regression model with more than one dependent variable (several outcomes).

In this case the goal is to find a function $f(x) = wx + b$ (red line) under the condition that $f(x)$ is within a required accuracy $\epsilon$ from the *value* $y(x)$ (black bars) of every data point, i.e. $|y(x) - f(x)| \leq \epsilon$ where *epsilon* is the distance between the red and the grey line. Under this condition we again want to minimize $f'(x) = w$, again for the reason of regularisation and to obtain a unique solution as the result of the convex optimization problem. One can see how minimizing $w$ results in a more general case as the extreme value of $w = 0$ would mean no functional relation at all which is the most general result one can obtain from the data.




The data points at the 2 red bars are the support vectors in this case, they correspond to the non-zero Lagrange multipliers of the equality part of the inequality condition $|y - f(x)| \leq \epsilon$.

Both cases result in the following problem:

Under the condition that:

$$min \frac{1}{2} w^2$$

- All examples are classified correctly (Classification)
- The value $\boldsymbol{y}$ of all examples deviates less than $\epsilon$ from $f(x)$. (Regression)

### 3.3 Evaluation Approaches

For classification process, we evaluate the classification result base on Accuracy, Precision, Recall and F1-score. All the measures can be calculated by using the four parameters of confusion matrix which are:

- True Positives (TP) - These are the correctly predicted positive values which means that the value of actual class is yes and the value of predicted class is also yes.
- True Negatives (TN) - These are the correctly predicted negative values which means that the value of actual class is no and value of predicted class is also no.
- False Positives (FP) – When actual class is no and predicted class is yes.
- False Negatives (FN) – When actual class is yes but predicted class in no.

After understanding of these four parameters then Accuracy, Precision, Recall and F1 score can be calculated.

Accuracy - Accuracy is the most intuitive performance measure and it is simply a ratio of correctly predicted observation to the total observations.

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+FN+TN} \quad (3.4)$$

Precision is the ratio of correctly predicted positive observations to the total predicted positive observations.

$$\text{Precision} = \frac{TP}{TP+FP} \quad (3.5)$$

Recall is the ratio of correctly predicted positive observations to the all observations in actual class - yes.

$$\text{Recall} = \frac{TP}{TP+FN} \quad (3.6)$$

F1 Score is the weighted average of Precision and Recall.

$$\text{F1 Score} = 2 \times \frac{(\text{Recall} \times \text{Precision})}{(\text{Recall} + \text{Precision})} \quad (3.7)$$

For regression process, we evaluate the regression results based on *Root Mean Squared Error* ***(****RMSE****)*** and *Coefficient of Determination (R2)*. RMSE measures the difference between the predicted values by a model and the observed values. RMSE ranges from 0 to ∞ where lower values signify better models. RMSE can be described by the following formula:

$$\text{RSME} = \sqrt{\frac{\sum_{t=1}^{n}(y_{obs}^{t} - y_{pred}^{t})}{n}} \quad (3.8)$$

where $y_{obs}^{t}$ and $y_{pred}^{t}$ are the observed and predicted scores for instance *t* (where *t* = 1 . . . *n*) and *n* is the sample size. *R*2 is the ratio of the model's absolute error and the baseline mean predicted scores. It is expressed as:

$$R^2 = 100 \times (1 - \frac{\sum_{t=1}^{n}(y_{obs}^{t} - y_{pred}^{t})}{\sum_{t=1}^{n}(y_{obs}^{t} - \hat{y_{obs}})}) \qquad (3.9)$$

where $y_{obs}^{t}$ and $\hat{y_{obs}}$ are respectively the observed scores and their mean, and $y_{pred}^{t}$ are the predicted scores by the model. *R*2 measures the relative improvement of the mean squared error using the automatic predictor compared to the average baseline. Positive values indicate that the model accounts for a greater proportion of the variance in the data thus outperforming the constant average baseline. Negative values indicate that variation in the data accounted for by the model is worse than the baseline score, thus not outperforming the baseline.

### 3.4 The Basic SVM Classification Calculation Example

Training sentences:

0. likes the sound of thunder
1. is so sleepy it's not even funny that's she can't get to sleep.
2. is sore and wants the knot of muscles at the base of her neck to stop hurting.

On the other hand...

3. likes how the day sounds in this new song.
4. you are idiot. <3

Target Classes:

y = ['Y',' Y',' Y',' Y',' N']

After preprocessing steps, training sentences will be:

0. likes sound thunder
1. sleepy even funny get sleep
2. sore wants knot muscles base neck stop hurting hand yay illinois
3. likes day sounds new song
4. idiot

Then, sentences are transformed into features using TfidfVectorizer**.**

Result will be: Data Point (Document index, Specific word-vector index) and TFIDF Score.

- TF (t) = (Number of times term t appears in a document) / (Total number of terms in the document)
- IDF (t) = log_e (Total number of documents / Number of documents with term t in it)
- TF (likes) = 1/3
- IDF (likes) = log_e (5/2) = 1.486571372391151
- TFIDF_score (likes) = TF (like)*IDF (like) = 0.4955237907970503

So, result of all training sentences after applying TfidfVectorizer are:

| Features | TFIDF Score | Data Point |
|---|---|---|
| ➢ likes | 0.4955237907970503 | (0 ,10) |
| ➢ sound | 0.6141889663426562 | (0,18) |
| ➢ thunder | 0.6141889663426562 | (0,21) |
| ➢ sleepy | 0.4472135954999579 | (1,15) |
| ➢ even | 0.4472135954999579 | (1,2) |
| ➢ funny | 0.4472135954999579 | (1,3) |
| ➢ get | 0.4472135954999579 | (1,4) |
| ➢ sleep | 0.4472135954999579 | (1,14) |
| ➢ sore | 0.3015113445777636 | (2,17) |
| ➢ wants | 0.3015113445777636 | (2,22) |
| ➢ knot | 0.3015113445777636 | (2,9) |
| ➢ muscles | 0.3015113445777636 | (2,11) |
| ➢ base | 0.3015113445777636 | (2,0) |
| ➢ neck | 0.3015113445777636 | (2, 12) |
| ➢ stop | 0.3015113445777636 | (2, 20) |
| ➢ hurting | 0.3015113445777636 | (2, 7) |
| ➢ hand | 0.3015113445777636 | (2, 5) |
| ➢ yay | 0.3015113445777636 | (2, 23) |
| ➢ illinois | 0.3015113445777636 | (2, 8) |
| ➢ likes | 0.3741047724501572 | (3, 10) |
| ➢ day | 0.4636932227319092 | (3, 1) |
| ➢ sounds | 0.4636932227319092 | (3, 19) |
| ➢ new | 0.4636932227319092 | (3, 13) |

- song 0.4636932227319092 (3, 16)
- idiot 1.0 (4, 6)

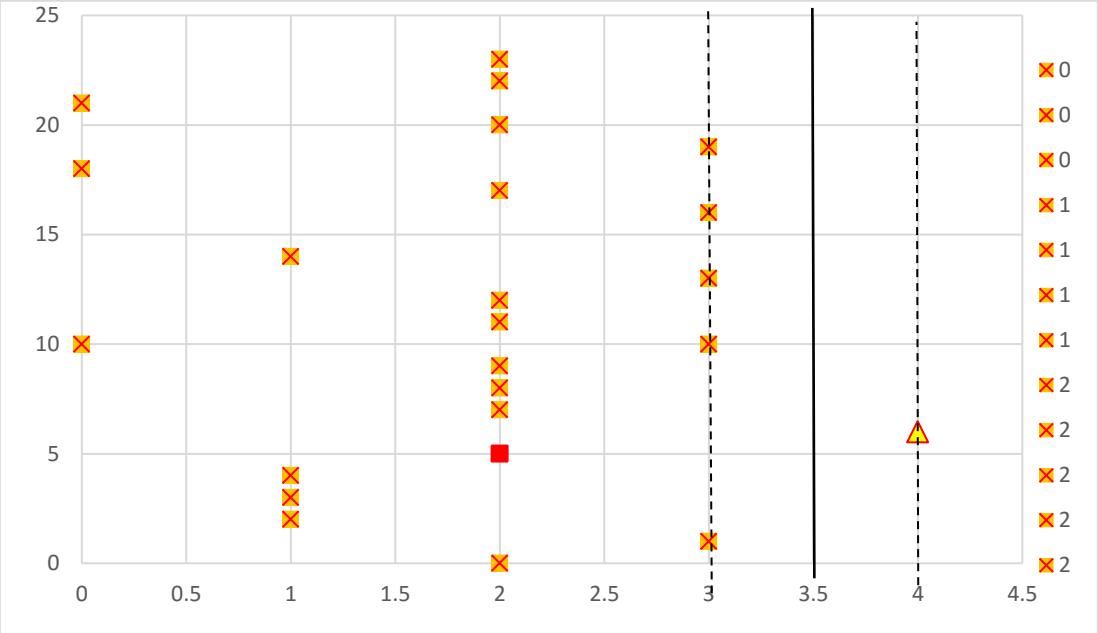



**Figure 3.2 Support Vector Machine**

Support vectors are:

$s1=\begin{pmatrix}3\\1\end{pmatrix}$, $s2=\begin{pmatrix}3\\10\end{pmatrix}$, $s3=\begin{pmatrix}3\\10\end{pmatrix}$, $s4=\begin{pmatrix}3\\16\end{pmatrix}$, $s5=\begin{pmatrix}3\\19\end{pmatrix}$, $s6=\begin{pmatrix}4\\6\end{pmatrix}$

$y = wx + b$

For s1, $\begin{pmatrix}w0\\w1\end{pmatrix}\begin{pmatrix}3\\1\end{pmatrix} + b = 1$, $3w0 + w1 + b = 1$

For s2, $3w0 + 10w1 + b = 1$

For s3, $3w0 + 13w1 + b = 1$

For s4, $3w0 + 16w1 + b = 1$

For s5, $3w0 + 19w1 + b = 1$

For s6, $4w0 + 6w1 + b = -1$

Finally, $w0 = -2$, $w1 = 0$, $b = 7$

Therefore, $\vec{w} = \begin{pmatrix} w0 \\ w1 \end{pmatrix} = \begin{pmatrix} -2 \\ 0 \end{pmatrix}$ , b = 7

The basic concept of the classification and regression with Support Vector Machine (SVM) are the same.

### 3.5 Experiment and Results

In this section, we present the details of the experiments and the results of personality prediction using *myPersonality* dataset which has 9918 rows of Facebook user status posts and their Big Five results. In this experiment, the dataset is split into Training data (70%) and Testing data (30%). The system train on Training data with Support Vector Machine (SVM). And then, the system predicts the Five Factor Model for the remaining Testing data. The experimental results for each traits of Big Five Model are shown below:

Trait: OPENNESS

- Result: 0.9176470588235294
- Accuracy: 74.49%
- Classification Report:

| | precision | recall | f1-score |
|---|---|---|---|
| False | 0.49 | 0.16 | 0.24 |
| True | 0.77 | 0.94 | 0.85 |
| avg / total | 0.70 | 0.74 | 0.69 |

Trait: CONCIENTIOUSNESS

- Result: 0.4121008403361345
- Accuracy: 59.13%
- Classification Report:

| | precision | recall | f1-score |
|---|---|---|---|
| False | 0.62 | 0.66 | 0.64 |
| True | 0.55 | 0.50 | 0.53 |
| avg / total | 0.59 | 0.59 | 0.59 |

Trait: EXTROVERSION

- Result: 0.38285714285714284
- Accuracy: 57.14%
- Classification Report:

| | precision | recall | f1-score |
|---|---|---|---|
| False | 0.63 | 0.66 | 0.64 |
| True | 0.48 | 0.44 | 0.46 |
| avg / total | 0.57 | 0.57 | 0.57 |

Trait: AGREEABLENESS

- Result: 0.5912605042016806
- Accuracy: 57.31%
- Classification Report:

| | precision | recall | f1-score |
|---|---|---|---|
| False | 0.54 | 0.48 | 0.51 |
| True | 0.59 | 0.65 | 0.62 |
| avg / total | 0.57 | 0.57 | 0.57 |

Trait: NEUROTICISM

- Result: 0.22453781512605042
- Accuracy: 62.05%
- Classification Report:

| | precision | recall | f1-score |
|---|---|---|---|
| False | 0.65 | 0.82 | 0.73 |
| True | 0.51 | 0.30 | 0.38 |
| avg / total | 0.60 | 0.62 | 0.59 |

For Regression process, the results are as follow:

| TRAIT | RESULT | RMSE | R2_SCORE |
|---|---|---|---|
| OPENNESS | 4.1229 | 0.58 | 0.00 |
| CONCIENTIOUSNESS | 3.5010 | 0.74 | -0.01 |
| EXTROVERSION | 3.3766 | 0.85 | 0.00 |
| AGREEABLENESS | 3.6247 | 0.68 | 0.01 |
| NEUROTICISM | 3.6130 | 0.76 | -0.01 |

### 3.6 System Implementation

As soon as the system is started, the welcome page or main page which has a button named Login with Twitter and the detail of the system about what are the Big Five Personality Traits. When the user clicks on the button, the system goes to the Login into Twitter page. The main page of the system is shown in Figure 3.3.

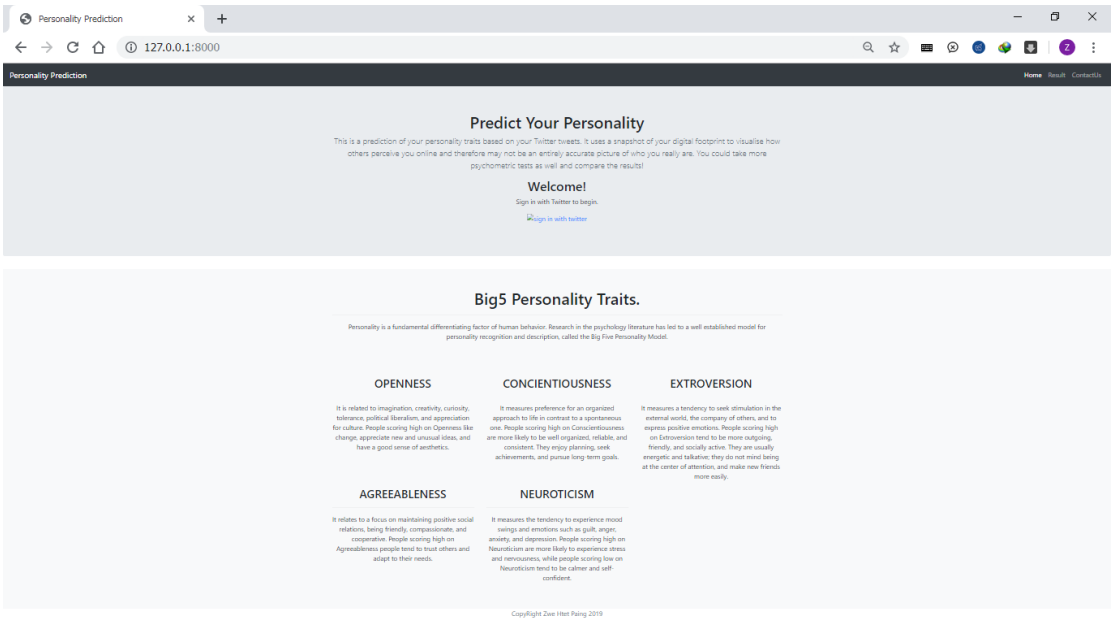



**Figure 3.3 Main Page**

When the user reaches the twitter login page, the user needs to enter the username or email address and password of their Twitter account to use the system as shown in Figure 3.4.

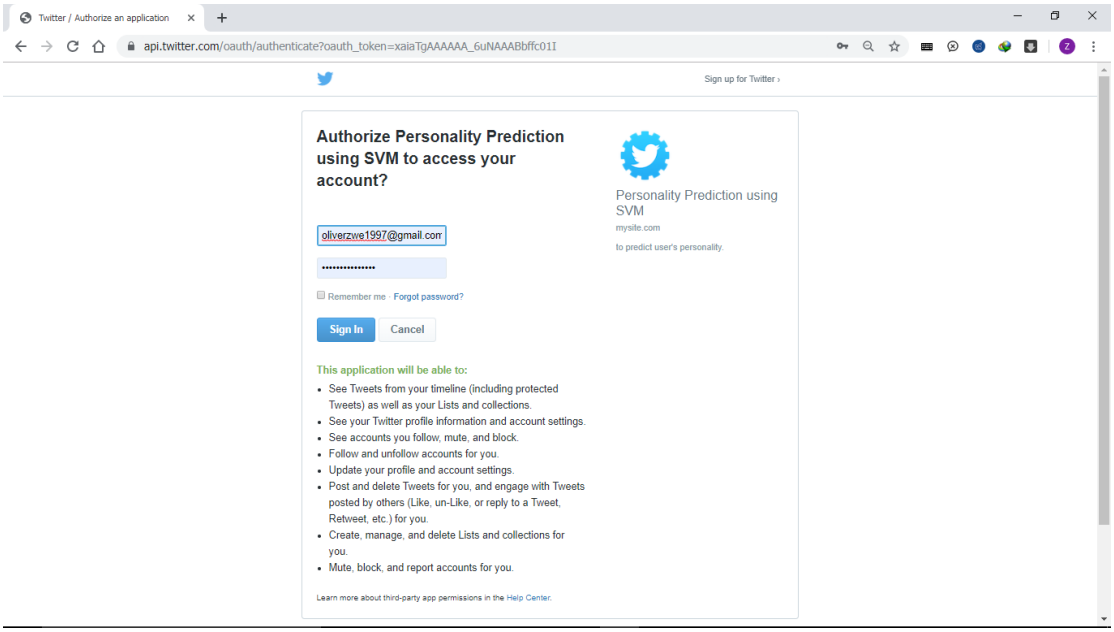



**Figure 3.4 Login to Twitter Page**

After the user login to the system with Twitter account, the system goes to the Result page. At the Result page, the user's tweets and the big five personality traits of the user with YES/NO option and the percentage of each trait will be known by the user as shown in Figure 3.5.

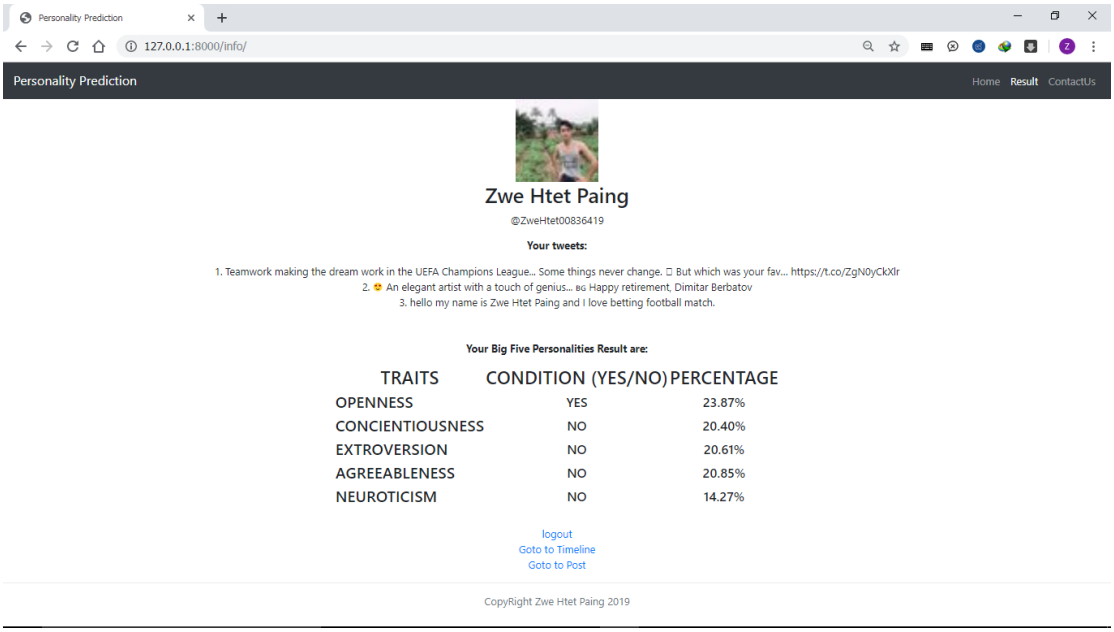



**Figure 3.5 Result Page**

There are two more option about the system – Twitter Timeline page and Post page. At the Twitter timeline page, the Timeline of the user's Twitter account can be seen as shown in Figure 3.6.

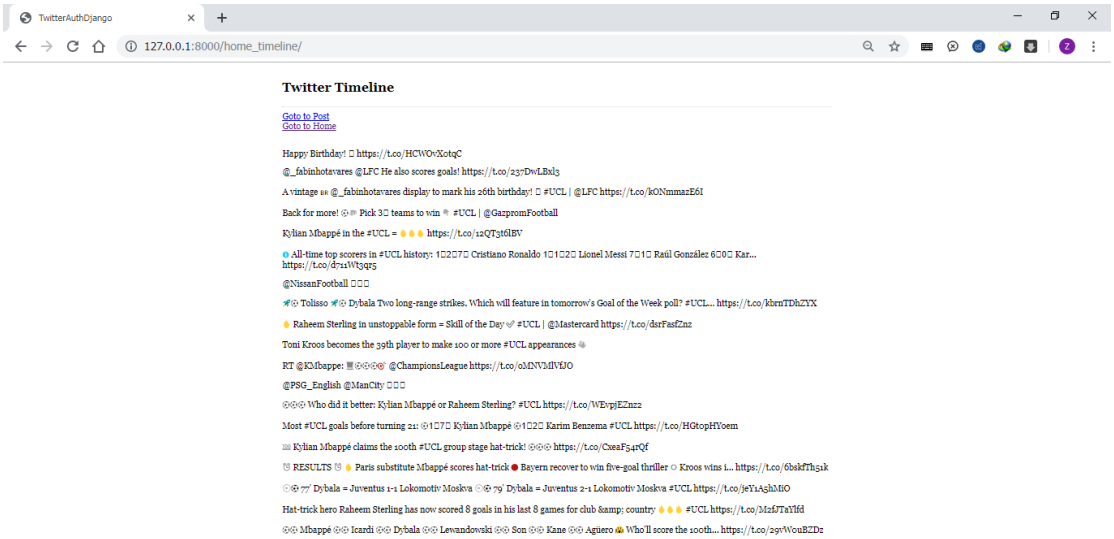



**Figure 3.6 Timeline Page**

At the Post page, the user can post the text that the user wants to tweet in his/her Twitter account as shown in Figure 3.7.


**Figure 3.7 Post Page**

## 3.7 Chapter Summary

The system design, experiment and result, example calculation of SVM and system implementation are described in this chapter.

The conclusion, limitation and future extension of the system will be explained in the next chapter.