CECS 551 Final Project

1. Introduction

My project is divided into two parts – first, I plan to learn a machine learning tool that is SAS Enterprise miner and secondly twitter Sentimental analysis.

First part of my project will focus on learning a machine learning tool and applying it to Kaggle dataset challenge. SAS Enterprise miner is a tool for creating predictive and descriptive models. It can sample the given dataset by creating one or more data sets for training a machine learning model. It has also capabilities of replacing the missing values in dataset and creating an output file for test data. In this tool, the machine learning process is driven by process flow diagram. So, there is no need of coding for the machine learning process. Due to this, a person who has no coding knowledge but knows about concept of machine learning can easily work on the tool. For example – a student from Maths/Statistics department who is familiar with machine learning concept but does not want to use python or R code for implementing it can use this tool.

Second part of my project will focus on Twitter sentimental analysis. Twitter is attracting significant interests from the research community in the last few years. Twitter Sentiment analysis is one of the hottest topics of research nowadays. In this project, I would try to extract overall sentiments of the tweets using different machine learning algorithm. There are various applications of doing sentimental analysis on tweets such as – predicting presidential election, predicting stock prices and general opinion on any major event.

2. SAS Enterprise Miner Tool

2.1 Advantages -

1) This tool support wide variety of operation for machine learning and data mining process.
2) It is very simple to learn and handle complex problems.
3) This tool is generally used by analyst or statistician who are unware of any programming language.
4) This tool provides an easy usable graphical user interface which reduces the overall development time.

5) We can fine tune the performance of the model by setting different parameters provided in the interface. It has also capabilities of sharing the results with other using creating an output file.

6) The models created using the tool can be compared to see the most efficient model and their stats.

7) It is highly flexible and has open and extendible design.

2.2 Working -

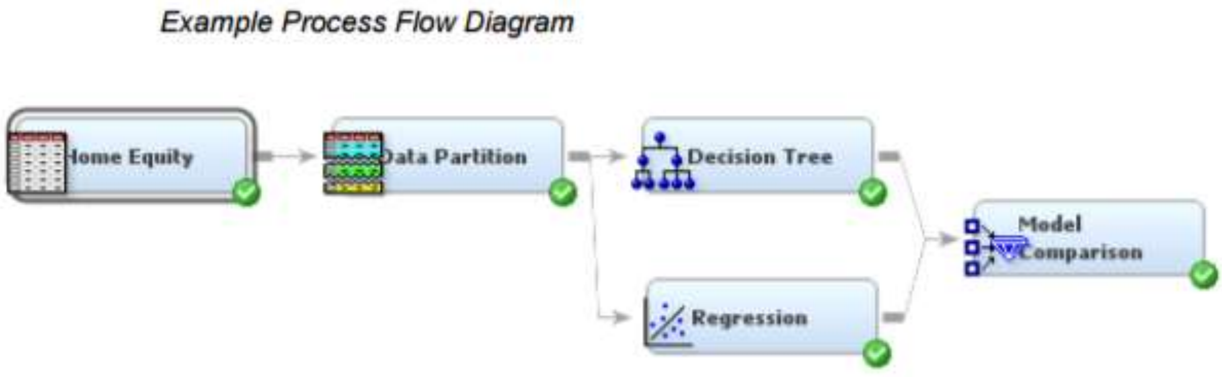
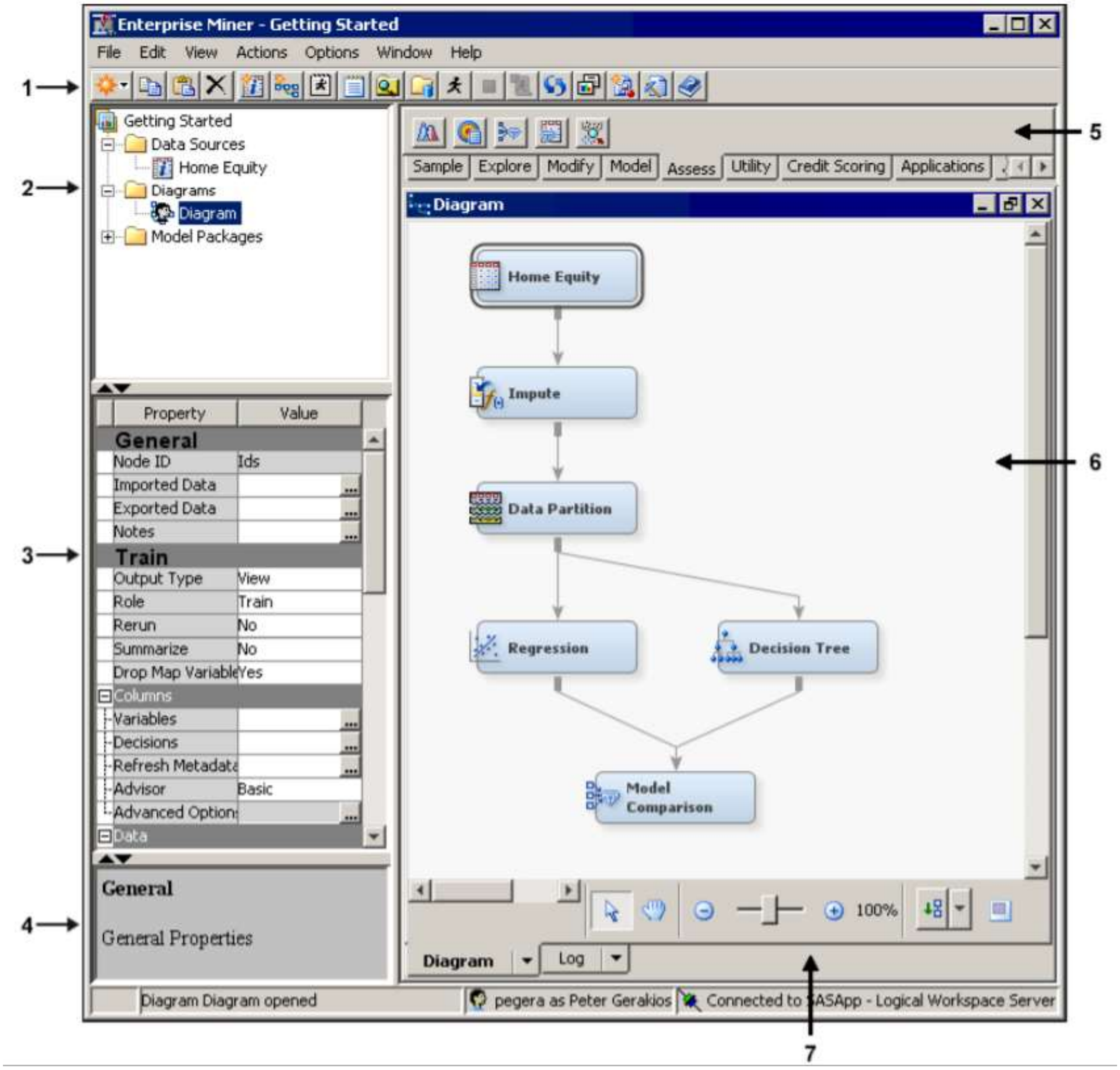


Fig. 1

In this tool, the machine learning or data mining process is made possible by process flow diagram as shown in above figure 1. The graphical user interface is very user friendly and it provides a quick overall picture of what process we are following for the predictive analysis. It provides many options to fine tune the parameters related to the different process. For example – In above diagram data partition provides the option of dividing the data set into training, validation and test data using parameters tuning. Similarly, we can fine tune different models in the process flow. For getting most perfect model, we need to use these fine-tuning parameters.

2.3 GUI for SAS Enterprise Miner Tool –

Figure 2 shows the graphical user interface for SAS enterprise miner. In the given figure 2, arrow 1 represents toolbar shortcut buttons which are used for frequently used applications. Arrow 2 represents the panel in which we can see the details regarding the project. It contains all files like imported dataset and diagram created in this panel. 3rd arrow represents the properties panel which is used to edit different parameters to fine tune the model building process. For example – For specifying the output file path and name we need to look into this 3rd panel. 4th box is for property help panel. It shows short description of any property you select. User creates the diagram in area marked with 6. This area is called as diagram workspace. It is used to create a process flow diagram for machine learning models. 7th area in below figure represents the diagram navigation toolbar.



Property	Value
General	
Node ID	Ids
Imported Data	...
Exported Data	...
Notes	...
Train	
Output Type	View
Role	Train
Rerun	No
Summarize	No
Drop Map Variable	Yes
⊟ Columns	
Variables	...
Decisions	...
Refresh Metadata	...
Advisor	Basic
Advanced Options	...
⊟ Data	

General

General Properties

Diagram Diagram opened

pegera as Peter Gerakios

Connected to SASApp - Logical Workspace Server

Figure 2 – GUI

Below figure 3 represents a sample diagram which can be built using the tool for a given dataset for predictive analysis.

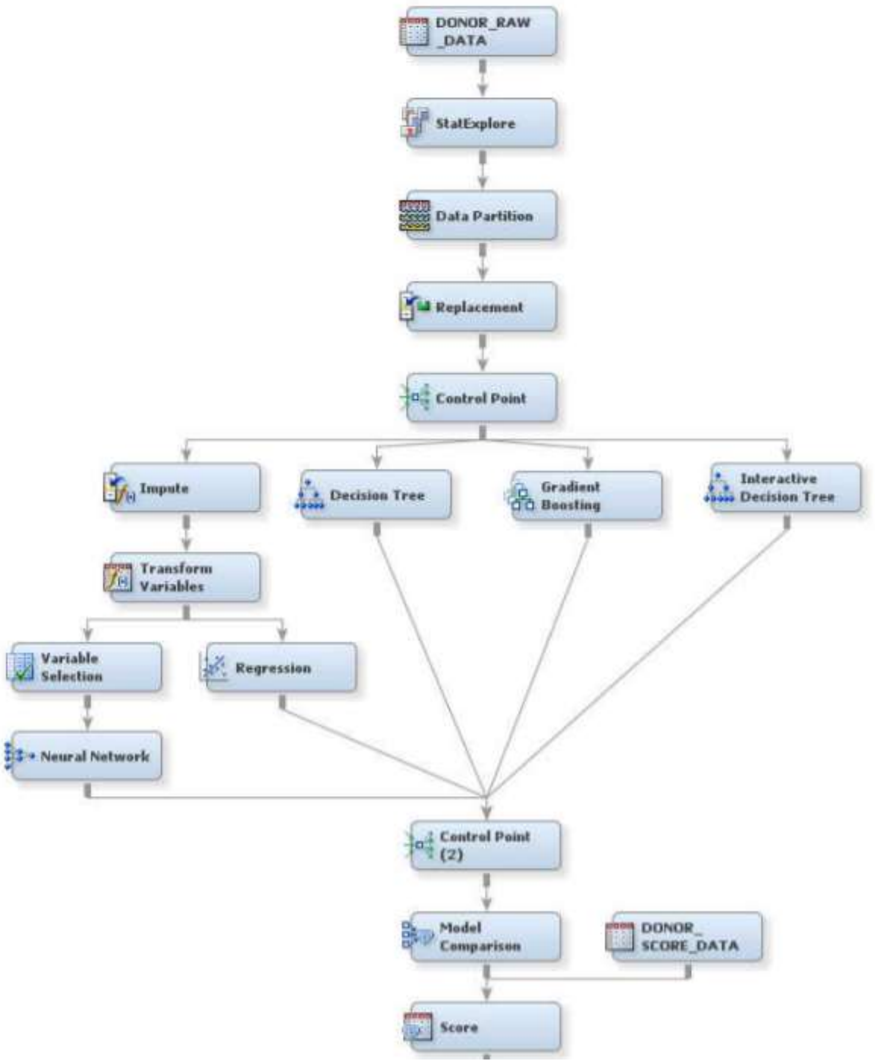


Figure 3

2.4 **Model development process**

The model development process is controlled using 5th arrow in figure 2. This represent Toolbar it is graphical node items which we use to build flow diagrams. We need to drag the nodes from the toolbar into the diagram workspace. After dragging the node, we need to connect the nodes according to the flow of the model building.

In the below figure 4, we can see that the any diagram workspace can be divided between below nodes-

1) Sample – It consist of input data upon which we need to build the model. Also, we have data partition node (for dividing the data into training and validation), sample node(for creating samples with/without replacement), filter (detect outliers)etc. It is used for specifying the required and excluded fields in the given dataset.

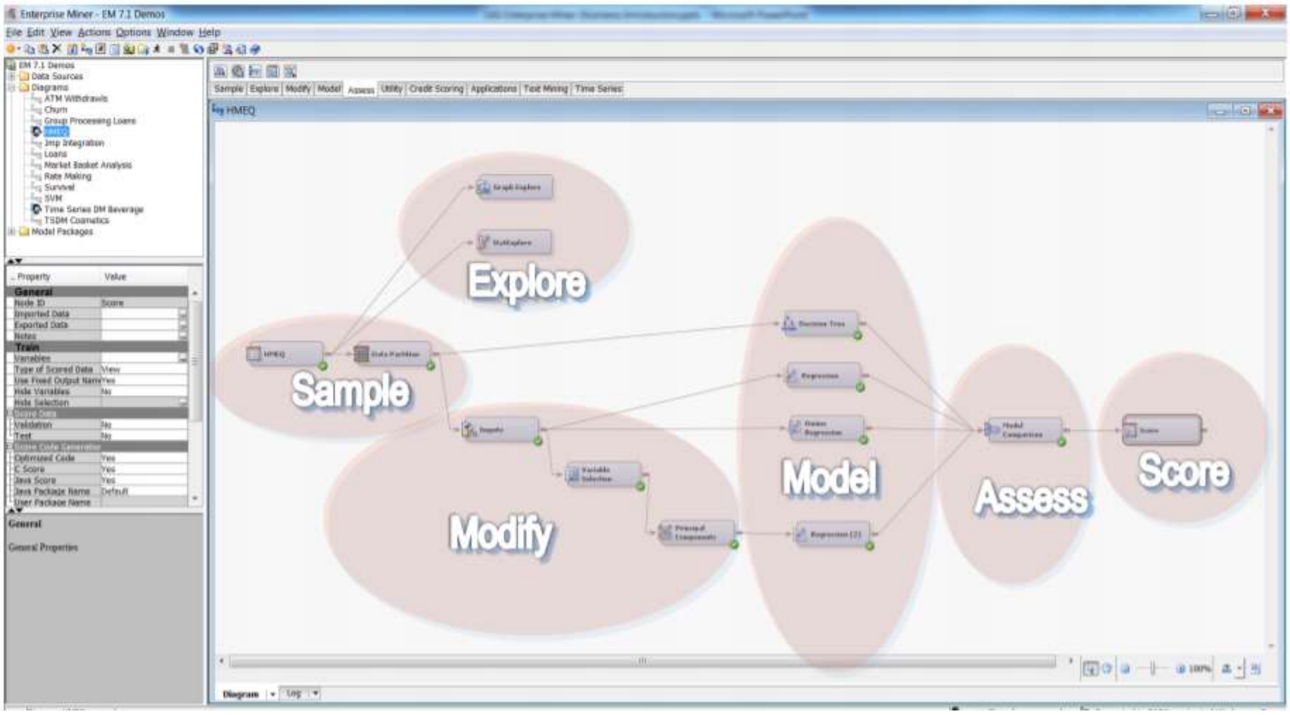
2) Explore – It is used for statistical measure , identifying outliers, data visualization and variable creation and selection. It consist of – graph explorer(to generate graphical reports), multipot(generate various charts and plots) and statExplorer(generate summary and association statistics).

3) Modify – This is used to modify the given dataset. We may need this to replace the missing values. Replacement node is used to replace specific data values and unknown level for class variables.

4) Model – It is set of all the possible models for machine learning. For example - neural network, decision trees, gradient boosting and regression etc.

5) Assess – This is useful for model comparison. It is also used to generate the final output file which is expected after applying the best model on the given test data using score node. We can also specify decisions. It defines target profiles for a target that produces optimal decisions. The decisions are made using a user-specified decision matrix and output from a subsequent modeling procedure.

Figure 5 represent all possible nodes which can used in diagram workspace.



Property	Value
General	
Node ID	Score
Imported Data	
Exported Data	
Notes	
Train	
Variables	
Type of Scored Data	View
Use Fixed Output Name	Yes
Hide Variables	No
Hide Selection	
Score Data	
Validation	No
Test	No
Score Code Generation	
Optimized Code	Yes
C Score	Yes
Java Score	Yes
Java Package Name	Default
User Package Name	

General

General Properties

Figure 4

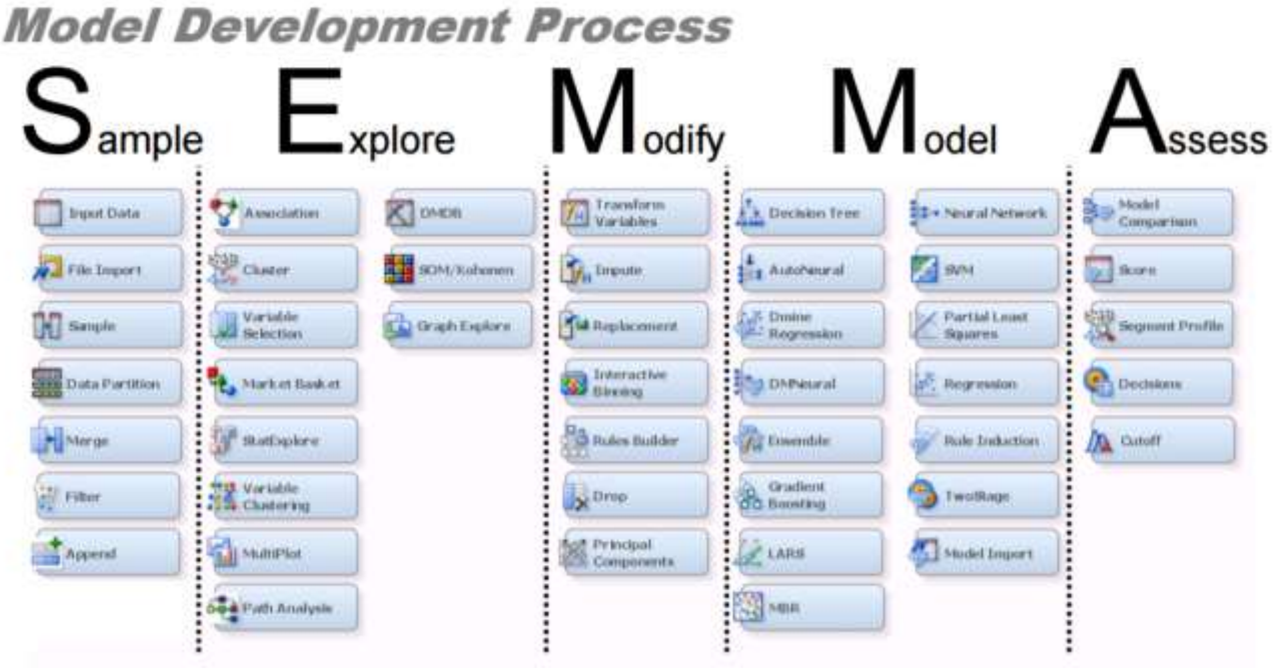
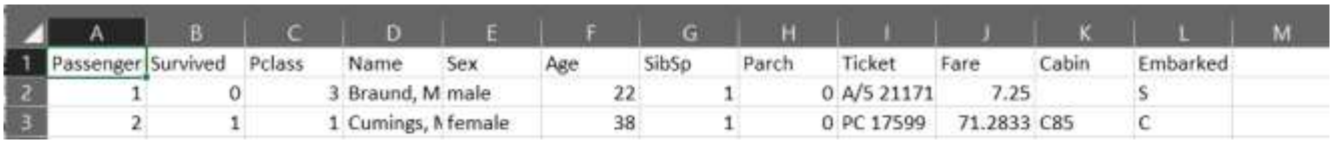


Figure 5

2.5 <u>Solving Kaggle Challenge using SAS Enterprise Miner tool</u> –

Description of problem is given in the Kaggle website link - https://www.kaggle.com/c/titanic
It is our job to predict if a passenger survived the sinking of the Titanic or not.
For each PassengerId in the test set, we must predict a 0 or 1 value for the *Survived* variable.
We have been given training and test data with the format and meaning of all the variables.
Below are the steps taken to solve the titanic problem –

1) Downloaded and analyzed the training and test excel sheets from Kaggle.



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Passenger	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	
2	1	0	3	Braund, M	male	22	1	0	A/5 21171	7.25		S	
3	2	1	1	Cumings, N	female	38	1	0	PC 17599	71.2833	C85	C	

Above is a screenshot of training dataset excel sheet. It contains 892 entries.

2) After downloading the dataset, I imported the excel sheet into the SAS Enterprise Guide tool. Go to File ->Import data and specify the filename and path to excel file.

3) After this, I created a new project under SAS enterprise miner tool and added the data library by going to File->library->add new library. Through this we will import the dataset into project environment.

4) Next, I created new data source in project panel and provided the link to the library created in previous step. While creating the data source we need to specify the role of the variables in the dataset. In our case we would change the role of PassengerId to ID and Survived to Target variable. Next, we would check if datatype for all variable is defined correctly.

5) After creating the data source, I created a new diagram to define the process flow.

6) Next, drag the train dataset from the data source Column and put it under newly created diagram.

7) Added graph explorer and stat explorer for visualizing the data.

8) After this we need to select a model from toolbar which we need to train from our given dataset. I added more models like – neural network and regression into the diagram workspace.

9) I added model comparator which would compare the results of above 3 models and choose the best among them.

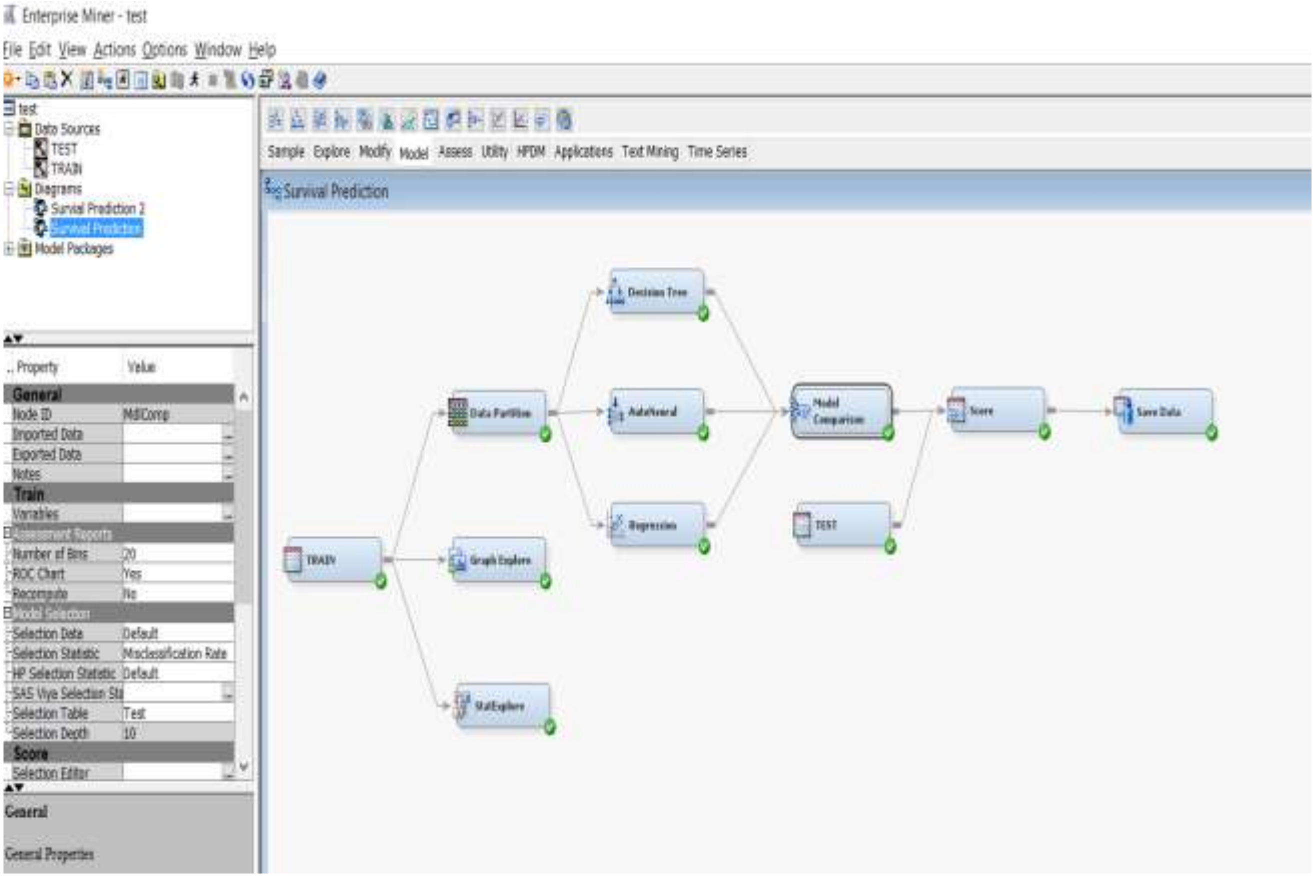
10) For getting more accurate results on test data I added data partition node which is used to partition the data into training and validation data. One can specify the percentage of data to be divided into training and validation data.

11) Now, dragged and dropped the test data from the datasource into the diagram workspace as we want to get the result on test data after applying the trained model onto it.

12) For generating the output file, I needed to add Save data node for which I specified the output file name and path. After running this node I could see the output file in the specified path.

13) Now, I had to change the column name of output file from EM_Classified to Survived as it was specified in kaggle website that I need to submit the output file in the specified format.

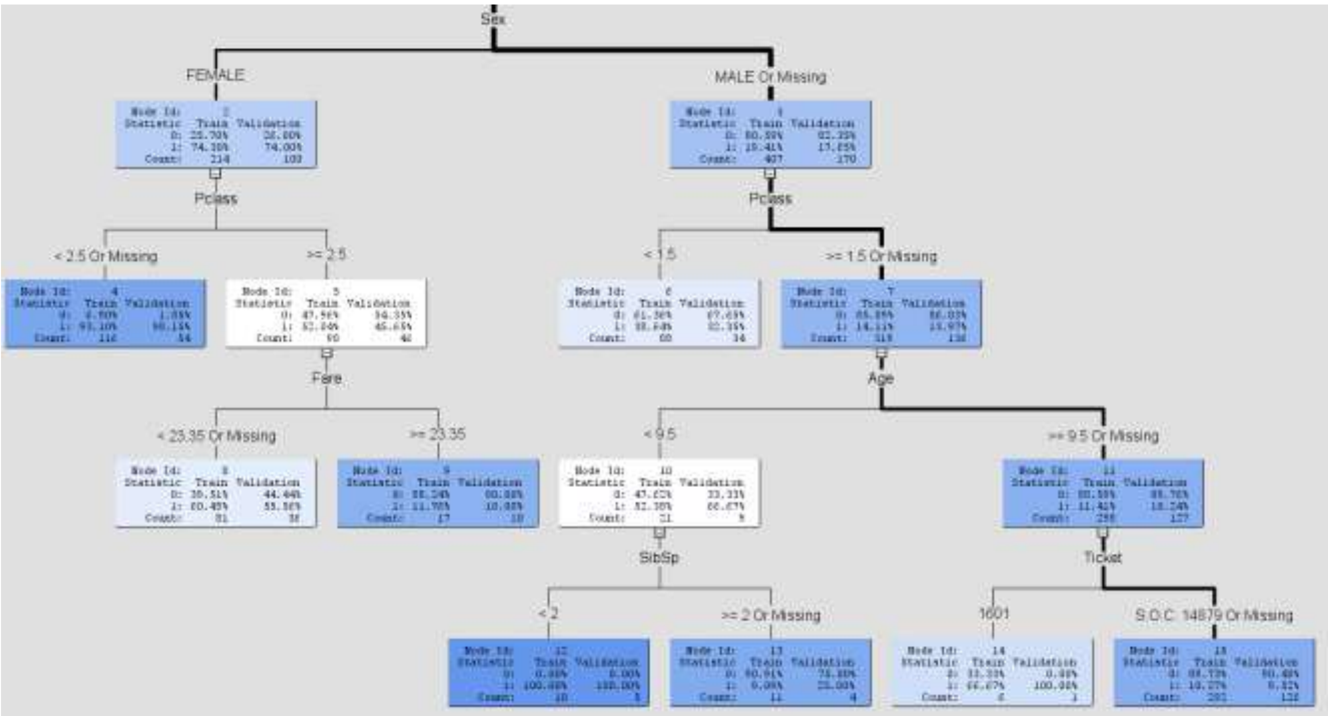
In below figure 6, I have shown the final diagram I created in SAS Enterprise Miner tool.



Property	Value
General	
Node ID	MdlComp
Imported Data	
Exported Data	
Notes	
Train	
Variables	
Assessment Reports	
Number of Bins	20
ROC Chart	Yes
Recompute	No
Model Selection	
Selection Data	Default
Selection Statistic	Misclassification Rate
HP Selection Statistic	Default
SAS Viya Selection Sta	
Selection Table	Test
Selection Depth	10
Score	
Selection Editor	

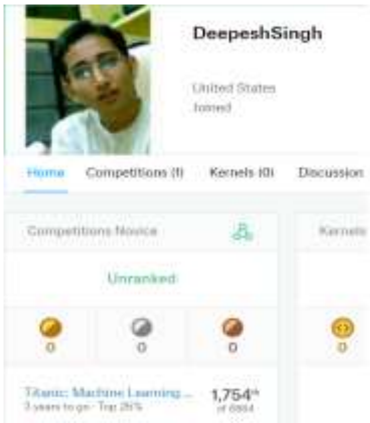
Figure 6

I used decision tree for training the titanic dataset. We can see the decision tree we get from the given data as below –


Sex
FEMALE
Node Id: 2 Statistic Train Validation 0: 25.70% 26.00% 1: 74.30% 74.00% Count: 214 100
MALE Or Missing
Node Id: 1 Statistic Train Validation 0: 80.59% 82.35% 1: 19.41% 17.65% Count: 407 170
Pclass
Pclass
< 2.5 Or Missing
>= 2.5
< 1.5
>= 1.5 Or Missing
Node Id: 4 Statistic Train Validation 0: 6.90% 1.85% 1: 93.10% 98.15% Count: 116 54
Node Id: 5 Statistic Train Validation 0: 47.96% 54.35% 1: 52.04% 45.65% Count: 98 46
Node Id: 6 Statistic Train Validation 0: 61.36% 67.65% 1: 38.64% 32.35% Count: 88 34
Node Id: 7 Statistic Train Validation 0: 85.89% 86.03% 1: 14.11% 13.97% Count: 319 136
Fare
Age
< 23.35 Or Missing
>= 23.35
< 9.5
>= 9.5 Or Missing
Node Id: 8 Statistic Train Validation 0: 39.51% 44.44% 1: 60.49% 55.56% Count: 81 36
Node Id: 9 Statistic Train Validation 0: 88.24% 90.00% 1: 11.76% 10.00% Count: 17 10
Node Id: 10 Statistic Train Validation 0: 47.62% 33.33% 1: 52.38% 66.67% Count: 21 9
Node Id: 11 Statistic Train Validation 0: 88.59% 81.76% 1: 11.41% 18.24% Count: 298 127
SibSp
Ticket
< 2
>= 2 Or Missing
1601
S.O.C. 14879 Or Missing
Node Id: 12 Statistic Train Validation 0: 0.00% 0.00% 1: 100.00% 100.00% Count: 10 5
Node Id: 13 Statistic Train Validation 0: 90.91% 75.00% 1: 9.09% 25.00% Count: 11 4
Node Id: 14 Statistic Train Validation 0: 33.33% 0.00% 1: 66.67% 100.00% Count: 6 1
Node Id: 15 Statistic Train Validation 0: 89.73% 90.48% 1: 10.27% 9.52% Count: 292 126

We can see from the above figure that the first split is made on the basis of variable - Sex then service class in which the person was travelling. Next split is made on age and fare of the ticket etc. We can clearly see the usefulness of this tool as it provides a good visualization of data which helps us to understand the given dataset complexity more easily.

After submitting my output file, I could see the result of my submission within 1-2 minutes. I could achieve score around 80 % and was ranked 1754 among 6884 teams as shown below -



DeepeshSingh

United States
Joined

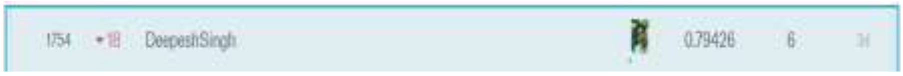
Home Competitions (1) Kernels (0) Discussion

Competitions Novice

Unranked

0 0 0 0

Titanic: Machine Learning ... 1,754th
3 years to go · Top 26% of 6884



1754 ▾18 DeepeshSingh 0.79426 6

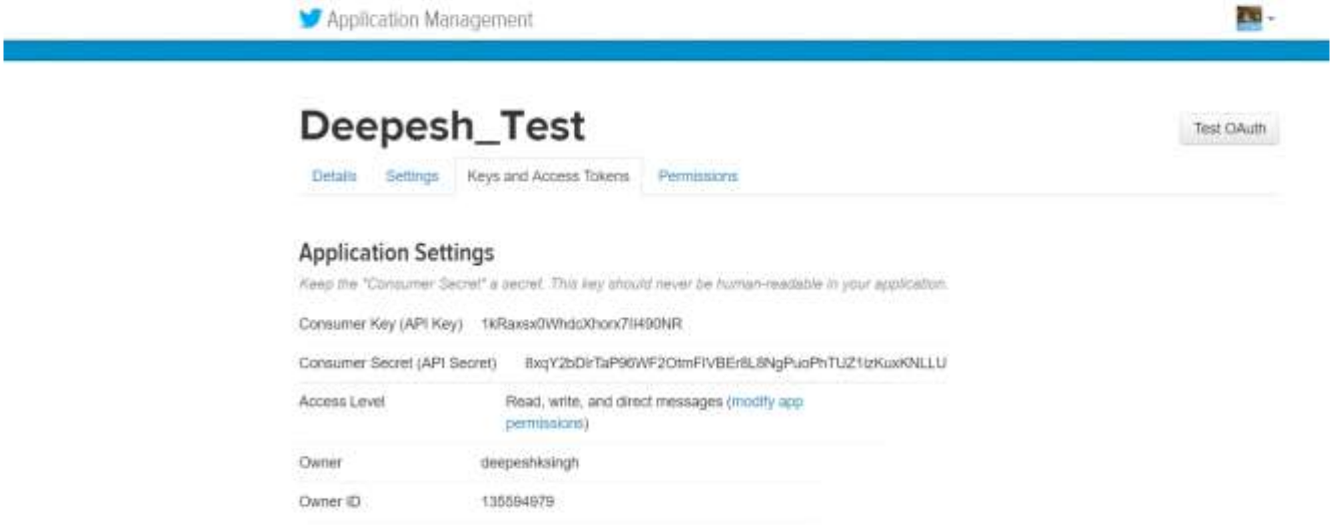
3. Twitter Sentimental analysis

Twitter Sentimental analysis is divided into two parts –

1) Data Collection and text pre-processing.
2) Sentimental analysis on the collected data.

3.1 Data Collection –

For getting data from twitter we need to create an app that would use twitter API. First, we need to register the app in http://apps.twitter.com . Login using your credential and select - register a new application. After registration, we would receive consumer key and consumer secret key as shown in below figure 7. These keys should not shared with anyone.



Deepesh_Test

Application Settings

Keep the "Consumer Secret" a secret. This key should never be human-readable in your application.

Consumer Key (API Key)	1kRaxsx0WhdoXhorx7lI490NR
Consumer Secret (API Secret)	8xqY2bDirTaP96WF2OtmFIVBEr8L8NgPuoPhTUZ1izKuxKNLLU
Access Level	Read, write, and direct messages (modify app permissions)
Owner	deepeshksingh
Owner ID	135594979

Figure 7

After this, I created a sample python program to test my application and see if I can see any data getting fetched from the twitter. I used Twitter OAuth to create a connection with twitter using its API as shown in figure 8. After app authentication with twitter I tested twitter streaming API and rest API. These are the two types of method using which one can get data from the twitter.

Function TwitterStream as shown below is used for streaming API and Twitter method call is used for Rest API.

Figure 8

A persistent HTTP connection is required to be open all the time for connecting to the streaming API. Figure 9 explains about streaming API. For an example, consider a web application which accepts user requests, makes one or more requests to Twitter's API, then formats and prints the result to the user, as a response to the user's initial request:

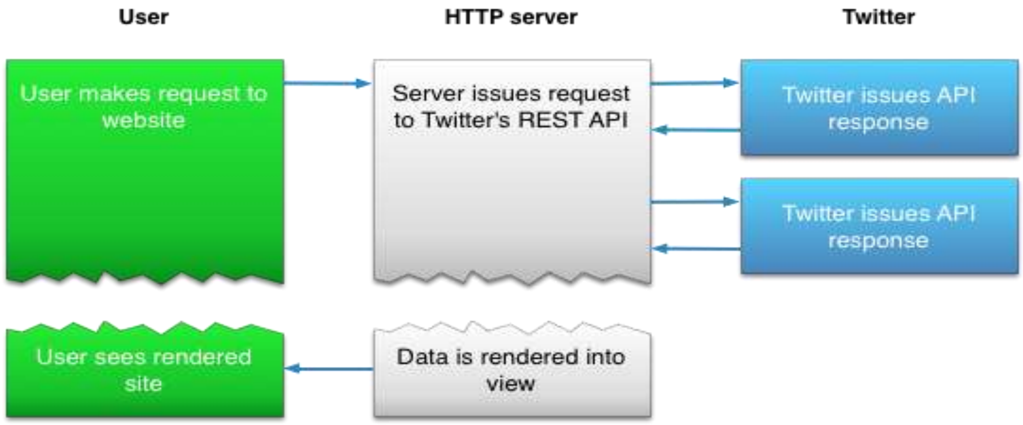

User
HTTP server
Twitter
User makes request to website
Server issues request to Twitter's REST API
Twitter issues API response
Twitter issues API response
User sees rendered site
Data is rendered into view

Figure 9

The streaming process gets the input Tweets and performs any parsing, filtering, and/or aggregation needed before storing the result to a data store. The HTTP handling process queries the data store for results in response to user requests. While this model is more complex than the first example, the benefits from having a realtime stream of Tweet data make the integration worthwhile for many types of apps.

Data fetched from the twitter using streaming and Rest API would be in JSON format where each field is shown in figure 10. We need to extract all these fields from the JSON and store the required data in an excel sheet or database like MongoDB.

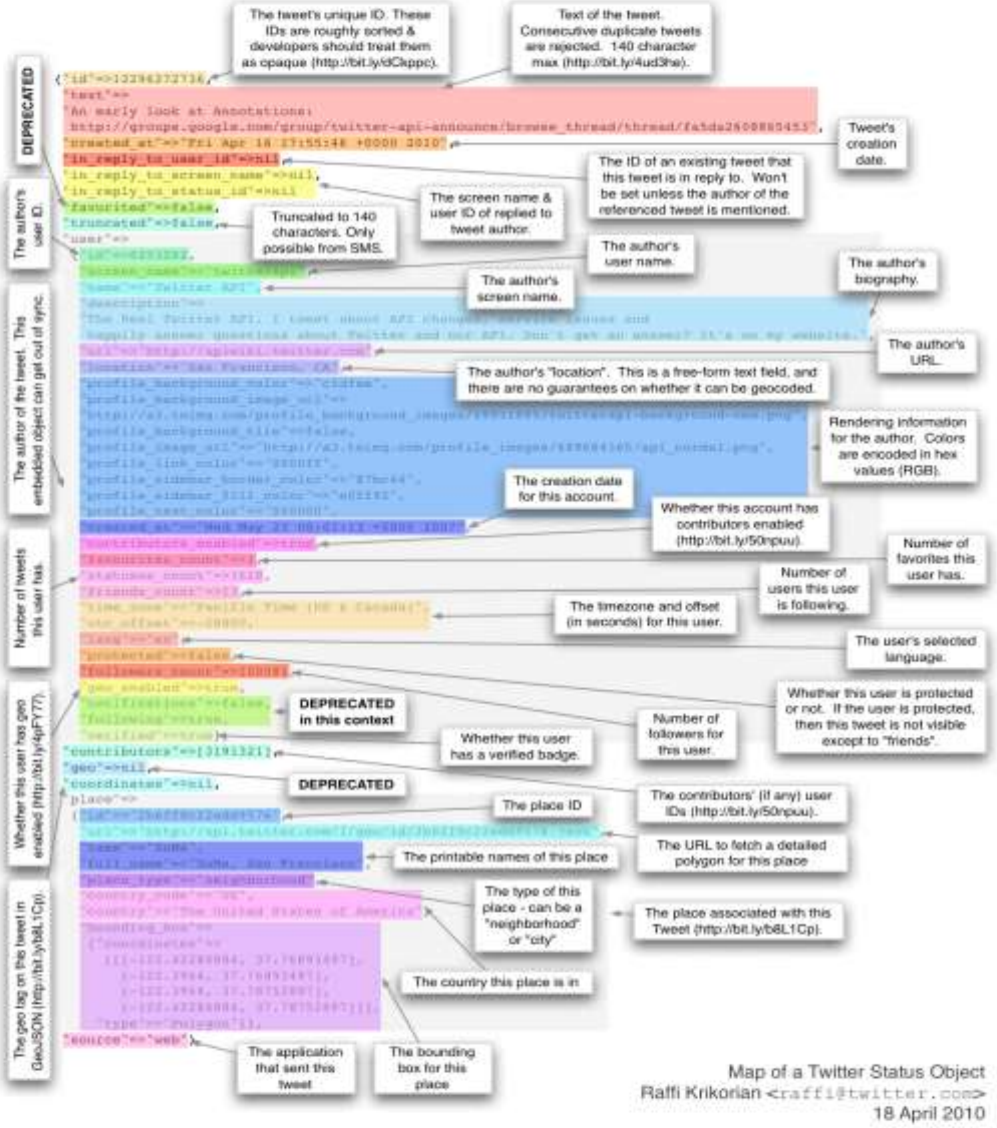

The tweet's unique ID. These IDs are roughly sorted & developers should treat them as opaque (http://bit.ly/dCkppc).
Text of the tweet. Consecutive duplicate tweets are rejected. 140 character max (http://bit.ly/4ud3he).
DEPRECATED
Tweet's creation date.
The ID of an existing tweet that this tweet is in reply to. Won't be set unless the author of the referenced tweet is mentioned.
The screen name & user ID of replied to tweet author.
The author's user ID.
Truncated to 140 characters. Only possible from SMS.
The author's user name.
The author's screen name.
The author's biography.
The author's URL.
The author of the tweet. This embedded object can get out of sync.
The author's "location". This is a free-form text field, and there are no guarantees on whether it can be geocoded.
Rendering information for the author. Colors are encoded in hex values (RGB).
The creation date for this account.
Whether this account has contributors enabled (http://bit.ly/50npuu).
Number of favorites this user has.
Number of tweets this user has.
Number of users this user is following.
The timezone and offset (in seconds) for this user.
The user's selected language.
Whether this user is protected or not. If the user is protected, then this tweet is not visible except to "friends".
DEPRECATED in this context
Number of followers for this user.
Whether this user has a verified badge.
Whether this user has geo enabled (http://bit.ly/4pFY77).
The contributors' (if any) user IDs (http://bit.ly/50npuu).
DEPRECATED
The place ID
The URL to fetch a detailed polygon for this place
The printable names of this place
The type of this place - can be a "neighborhood" or "city"
The place associated with this Tweet (http://bit.ly/b8L1Cp).
The geo tag on this tweet in GeoJSON (http://bit.ly/b8L1Cp).
The country this place is in
The application that sent this tweet
The bounding box for this place
Map of a Twitter Status Object
Raffi Krikorian <raffi@twitter.com>
18 April 2010

Figure 10

3.2 Data Preprocessing –

Tweets will contain lot of data which is not important like – HTML tags(URLs), @mentions, numbers, and hash tags etc. We can use regex or NLTK python library to remove these unwanted data from tweets. We can consider keeping emoticons as they represent the mood or sentiment of the employee. We can follow below steps for preprocessing the tweeter data -

1) Lower Case - Convert the tweets to lower case.
2) URLs - We can eliminate all of these URLs via regular expression matching or replace with generic word URL.
3) @username - we can eliminate "@username" via regex matching

4) #hashtag - hash tags can give us some useful information, so it is useful to replace them with the exact same word without the hash. E.g. #nike replaced with 'nike'.
5) Punctuations and additional white spaces – We can remove punctuation at the start and ending of the tweets. We can also replace multiple whitespaces with single whitespace.

3.3 Sentimental Analysis –

In this section I will be talking about different methods used in machine learning to predict the sentiments of a text mainly Naive Bayes Classifier, Maximum Entropy Classifier and Support Vector Machines.

A. Naïve Bayes Classifier

This model is simple probabilistic model and works well for text classification. Compare to support vector machine it takes relatively less time to train the model. High level of accuracy can be achieved using this model. Naïve Bayes classifier is a simple probabilistic model based on Bayes rule.

Given a set of positive or negative words, these sets are independent of each other. This assumption helps in fast classification of words. We need to define two classes – positive and negative. Maximum probability that a word belong to particular class is –

$$P(x_i \mid c) = \frac{Count\ of\ x_i\ in\ documents\ of\ class\ c}{Total\ no\ of\ words\ in\ documents\ of\ class\ c}$$

Frequency of each word is noted in a data structure during the training of model. Here, xi is the original word in the document. The probability that a particular word belongs to a given class is –

$$P(c_i|d) = \frac{P(d\ |c_i) * P(c_i)}{P(d)}$$

Model accuracy can be tested using 10 fold cross validation. Accuracy is calculated by referring to confusion matrix.

I have implemented sentimental analysis using Naïve Bayes Classifier in NLTK library. In code, I have defined two dictionaries for positive and negative words and assigned label to them. Then, I used these dictionaries to train the Naïve Bayes classifier for NLTK. After this, I split the given sentence into tokens of words and predicting the polarity of each word using the Bayesian trained model. At the end, I printed out the quantification of positive or negative sentiments in the given sentence.

As this dictionary was created by me, it had limited number of positive and negative words in it. So, I used movie review corpus which are categorized into positive and negative. This corpus is provided by NLTK library. Using this method, we can train our model for 2000 classified texts. This would give us a more accurate result.

B. Support Vector Machines

Support Vector machines (SVM) are known to be highly effective at text classification and they generally outperform Naive Bayes. They are not dependent on probability like Naive Bayes and Maximum Entropy. Basic algorithm used in this is to find a hyperplane, denoted by vector w, that separates the document vectors in one class from other. We try to keep the margin as large as possible. It is constrained optimization problem where the result belongs to 1 or -1 which is for positive and negative.

the correct class of document dj , the solution can be written as –

$$\vec{w} := \sum_j \alpha_j c_j \vec{d_j}, \ \ \alpha_j \geq 0,$$

where the a_j are obtained by solving a dual optimization problem Those d_j such that a_j is greater than zero are called support vectors, since they are the only document vectors contributing to w.
Classification of test data would consists simply of determining which side of w's hyperplane they would fall. Compare to Naïve Bayes they are take more time to run.

C. Maximum Entropy

Maximum entropy classification is another probablistic approach .This technique has been effective in a number of Natural language processing applications. Sometimes, it has been observered that it outperforms Naive Bayes at standard text classification. The P(c|d) formualae which we have seen above in Naive Bayes takes the below exponential form –

$$P_{\mathrm{ME}}(c \mid d) := \frac{1}{Z(d)} \exp\left(\sum_i \lambda_{i,c} F_{i,c}(d,c)\right)$$

In the above equation the Z(d) is a normalization function. $F_{i,c}$ is a feature or class function for feature fi and class c, defined as below –

$$F_{i,c}(d,c') := \begin{cases} 1, & n_i(d) > 0 \text{ and } c' = c \\ 0 & \text{otherwise} \end{cases}$$

Unlike Naive Bayes, maximum entropy does not make any assumption about the relationship between features, and so might also perform better when conditional independence assumptions are not met. The Parameter values are set so as to maximize the entropy of the induced distribution subject to the constraint that expected values of the feature or class function with respect to model are equal to expected values with respect to training data. The underlying philosophy is that we should choose the model making the fewest assumptions about the data while still remaining consistent with it, which makes intuitive sense.

4. Challenges in Sentimental Analysis

Negation –

Negation handling is one of the major factor that affects the accuracy of the classifier. In Sentiment classification task, negation is one of the issue faced.For example when we use each word as feature, the word "good" in the phrase "not good" will contribute to positive sentiment rather than negative sentiment as there is a "not" before it which is not considered while calculating the sentiment.

To solve this problem, we can use an algorithm for handling negations using state variables and bootstrapping. We can use idea of using different representation of negated forms. It transforms a word followed by a not into "not_"+word. So, whenever the negation state variable is set, the word being read are treated as "not_"+word. State variable is reset when a punctuation mark is encountered or when there is double negation.

n-grams –

Sentiments is also conveyed by adjectives in the given text. This information can be captured by adding features like consecutive pairs of words(bigrams). Words such as "very" or "definitely" don't provide much of sentiment info. as an individual word but when they are used in phrases like "very bad" or "definitely good" increase the probability of a document being negatively or positively biased. Counts of n-grams were stored in a hash table along with the counts of unigrams. We need to have a substantial amount of data in the training set to tackle the n-grams.

5. Conclusion –

In this project, I have learned about using a machine learning tool SAS Enterprise Miner and Twitter Sentimental Analysis.

SAS enterprise miner is a machine learning tool that is used for predictive and descriptive models. After learning the tool, I applied it to dataset given in Kaggle website. During this exercise, my main aim was to learn about this new tool and make myself familiar with Kaggle competitions. Now, I can solve more Kaggle competitions to practice and continue learning machine learning.

Sentiment analysis is the process of determining the opinion or feeling of a piece of text. We humans are very good at this process. But companies across the world have implemented machine learning to do this automatically for them. It is very useful for gaining insight into people opinions. People around the world post huge amount of reactions and opinions on every hot topic every second.

Twitter Sentimental analysis is divided into three parts – collection of data, preprocessing data collected and building machine learning model. There are various applications of doing sentimental analysis on tweets such as – predicting presidential election, predicting stock prices and general opinion on any major event.