Application of Sentiment Analysis in Talent Management

Ayushi Jha [1]
[1]ICT Department, Manipal Institute of Technology, India

1 Abstract

Management is pivotal for an organisation. It influences all its sectors and is one of the most apprehensive jobs. To keep up with the pace, an organisation needs a manager with good leadership qualities, communication skills, integrity, reliability and confidence. The main objective of this paper is to analyse the feedback(s) of a manager and output an opinion (In the form of rating) along with the most suitable sub-department (i.e., Logistics, Web-development, Sponsorship and Finance) that they can work in. The techniques used in the paper are Naive Baye's theorem for numeral ratings and a hybrid of Support-vector machine and decision tree for textual comments. The numeral rating will contribute in determining the final 5-star rating of a manager as well as their allotted sub-department. On the other hand, SVM/DT will be used to classify the textual reviews into either positive or negative and hence, contribute to the final rating.

Keywords: *Decision Tree, Support-vector Machine, Naive Baye's, Manager, Rating*

2 Introduction

A good manager is a key to well led organisations. An organisation must be conscious of who they appoint as their manager and conduct regular evaluations to keep a consistency check. The best way to judge the performance of a manager is by taking inputs from the people who work or deal with him/her. The data from these feedback forms can then be analysed to come up with several useful results. In this paper we work on enhancing talent management of an organisation by implementing efficient techniques and methodologies. We use Naive Baye's algorithm on the feedback data to get a sub-department under management which will be most suitable for a manager. Naive Baye's works best on numeric data, requires less training data, it is scalable, can make probabilistic predictions for any missing data and can be used for multiclass classification. These features make the algorithm less expensive, not much time consuming and efficient for a company. We have implemented this algorithm because it was the most suitable for the feedback form used by us. The next algorithm methodised in this paper is the Support-vector machine and Decision Tree hybrid algorithm. The SVM/DT is implemented on the textual review in the feedback form to contribute to the final 5-star rating. SVM being known as the most suitable technique for text classification, also has its own drawbacks. In the paper, we have discussed one of the major drawbacks and a way to overcome it.

3 Methodology

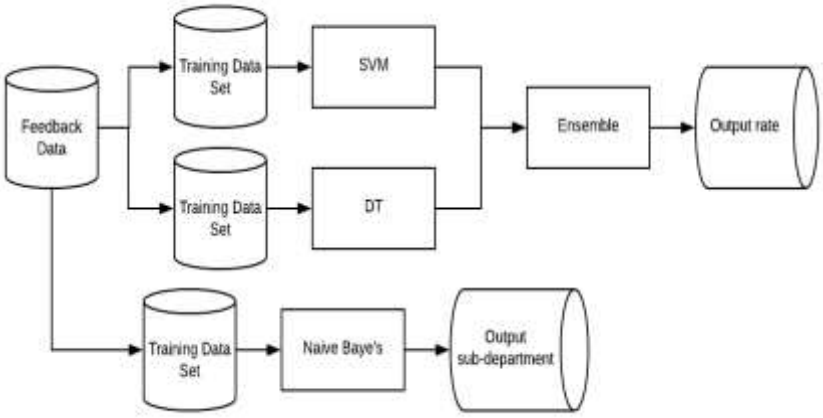

Fig .1 Proposed workflow focusing on the major aspects of the paper

Before working on the details of the Support-vector machine and Decision Tree hybrid algorithm, we first introduce the two base classifiers independently. This will help distinguish their roles in the final algorithm and make the linking of procedures more understandable.

3.1 Decision Tree

A decision tree is a tool that allows us to make decisions over a course of action to arrive at an optimal result. It is a diagrammatic (tree-like model) depiction of decision making and its outcomes. The decision tree has influenced a major part of machine learning as well as data mining. In this paper we will focus more on its mining aspects.
We have a set of data, $D=\{d_1, d_2, d_3,..., d_n\}$. Each data item is defined by a set of attributes. The vector of these attribute values are represented by the set, $X=\{x_1, x_2, x_3,...x_n\}$ where $X \in Z$, Z being a set of all possible attribute vectors. Each attribute of set X is assigned to a class. Therefore, we can say that every attribute of a data item is classified. A decision tree uses this phenomenon to inculcate decision analysis. A data item is classified based on its attribute set. The main problem is to allot attributes of a class that will be most suitable for data classification. Once, the data attributes are decided, a decision tree starts partitioning data items to form groups of similarly classified data. This process is performed until we get subsets each having data items of a particular class. [1]

3.2 Support-vector Machine

Support-vector Machine is a classifier that segregates real-time features into a higher dimensional feature space. Features are sectioned into various classes by forming hyperplanes in the graph.

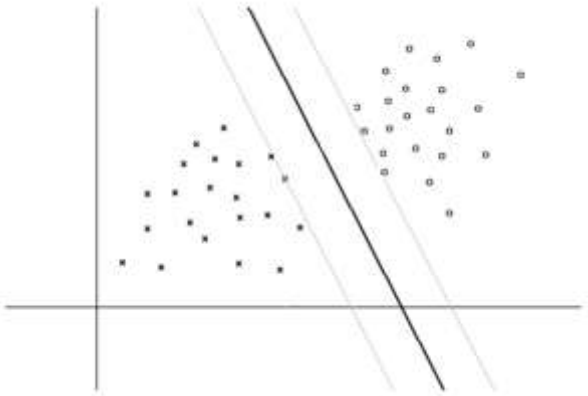

Fig .2 Linear Hyperplane in SVM [3]

The hyperplane with equation,

$$f(x)= w^T(x)+w_0$$

sections the features into two classes.

$f(x) \geq 1$, $\forall\ x \in$ Class 1

$f(x) \leq -1$, $\forall\ x \in$ Class 2

The hyperplanes need not necessarily be linear. There can be features which best classify with a non-linear hyperplane. SVM uses kernels to build these hyperplanes. Kernels are algorithms which analyse patterns based on the training data set and given conditions of classification. It forms the basis of SVM. [2]

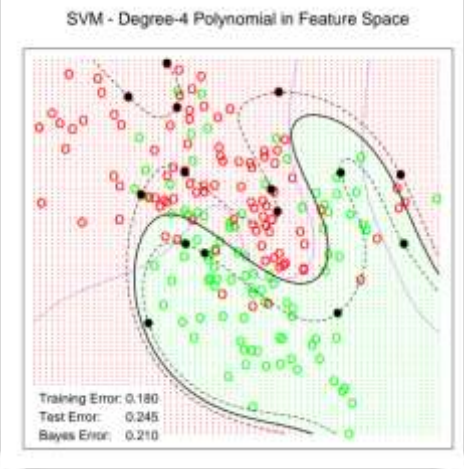

Fig .3 Non-linear hyperplane in SVM. [5]

3.3 Hybrid Algorithm

One of the major drawbacks of SVM is its inability to perform fast computation with large amount of data. SVM is the most efficient when it works with small quantities. In our case, a manager will be reviewed by all the employees of the company, employees of partner companies and customers (if required). The following algorithm is laid down on an assumption that the organisation is big enough to collect several feedback forms contributing to a large amount of data. Because of such a large quantity of data items, SVM will become slow. To overcome this we hybrid SVM with decision tree. In SVMDT we use SVM mechanism only on data points which are more crucial. For less crucial data points, we apply a much faster univariate decision. This makes the system speed up since SVM is performed only on few, selected data points.

We have used **Project Management Evaluation Form (by Client)** by **University of California San Francisco** as our basis of testing. The hybrid algorithm is performed on the textual feedback at the end of the form : **Were there any lessons that would be useful for future projects?**

Fig .4 Feedback form

1. For the training set, we conducted a short survey and got 104 data sets. Out of which 80% was used as the training data and the remaining 20% was used as the testing data.
   The following python code segregates training data and test data:
   TrainX, TestX, TrainY, TestY =
   Partition(x, label, testSize = 0.2, randomState=0)
2. Let us consider an SVM decision function,

$$f(x)= w^T(x)+w_0$$

   It is a linear function that separates data points into two sections.
   $f(x) \geq 1$, $\forall\ x \in$ Class 1
   $f(x) \leq -1$, $\forall\ x \in$ Class 2
3. As shown in figure 4, z represents the distance of the closest data point (in both the classes) to the hyperplane.

$$z= 1/||w||$$

We will now define a parameter to identify the measure of cruciality of data points, C(x). C(x) will be a value, lower for data points closer to the hyperplane and higher for data points that are farther. To set a limit to this cruciality measure, we define another parameter d(x) which acts as a threshold.

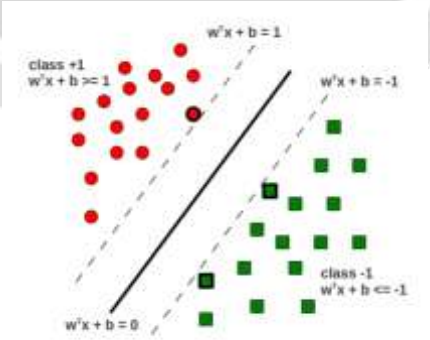

Fig .5 d(x)=Maximum margin from $w^T(x)+b=0$ to $w^T(x)+b=1$ or $w^T(x)+b=-1$. [4]

4. Data points with $C(x) \leq d(x)$ will be identified as a crucial test data and will go through SVM for testing whereas data points with $C(x) \geq d(x)$ will be identified as less crucial and will go through a univariate decision.

5. After these predictions we train the decision tree with three classes:
   $f(x)<0 \cap C(x) \geq d(x)$, $\forall x \in$ Class 1
   $f(x)>0 \cap C(x) \geq d(x)$, $\forall x \in$ Class 2
   $f(x)>0 \cap C(x) \leq d(x) + f(x)<0 \cap C(x) \leq d(x)$, $\forall x \in$ Class 3

6. Once the data points get segregated into appropriate classes, we save the file of each class and perform univariate decision on less crucial data to identify if the data points (Here, a word in a document) is positive or negative. Accordingly we conclude feedback to be positive or negative.
7. For the file with crucial data points we loop back to step 4.

The figure below depicts the hybrid algorithm through a flowchart.

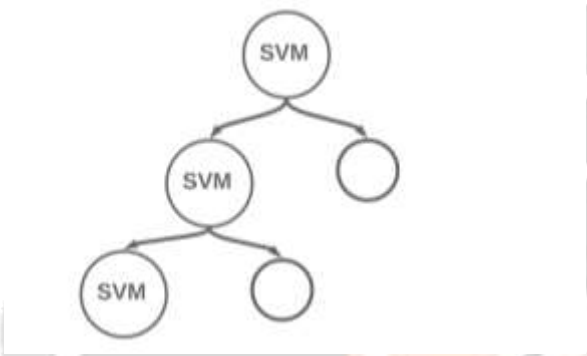

Fig .6 SVM/DT Flowchart. Empty nodes denote univariate operations

3.4 Naive Baye's Algorithm

The Naive Baye's Algorithm is used on the section of the form which inputs rating (1 to 5) as its feedback i.e, 1a to 1n and 2a to 2h. This algorithm is used to allot a sub-department to a management employee based on their feedback. The sub-departments include: Logistics, Web-development, Finance and Sponsorship. We conducted a survey and got 26 data sets for each sub-department. We used 100% of the data sets as training data and performed a test on a new sample data set given below. Refer to **Figure .4** for criterias.

Criteria	1a	1b	1c	1d	1e	1f	1g	1h	1i	1j	1k	1l	1m	1n
Rating	2	4	5	2	1	3	1	4	3	2	5	4	4	2

Criteria	2a	2b	2c	2d	2e	2f	2g	2h
Rating	3	4	4	1	3	5	3	2

To find final sub- department allotment:
L: Logistics, WD: Web-development, F: Finance, S: Sponsorship

1. Probability of sub-department = Logistics when 1a=1 is P(1a=1 | sys=L)
   Similarly, we calculate for 1a=2, 1a=3, 1a=4, 1a=5
   P(1a=2 | sys=L)
   P(1a=3 | sys=L)
   P(1a=4 | sys=L)
   P(1a=5 | sys=L)
   Similarly, we calculate probabilities for all possible criteria ratings when the department is Logistics.
   We follow the same pattern for remaining three departments.
   P(1a=1 | sys=WD)
   P(1a=2 | sys=WD)
   P(1a=3 | sys=WD)
   P(1a=4 | sys=WD)
   P(1a=5 | sys=WD)
2. After probability calculation of all the criteria with every possible rating and sub-department, we use Naive Baye's algorithm to calculate the probability of test data set to be of a particular sub-department. The department with highest probability is assigned to the manager with respect to the test form.
   argmax P(sys=L) P(1a=2 | sys=L) P(1b=4 | sys=L) P(1c=5 | sys=L) P(1d=2 | sys=L)... P(1n=2 | sys=L) P(2a=3 | sys=L)...P(2h=2 | sys=L)
   Vs.
   argmax P(sys=WD) P(1a=2 | sys=WD) P(1b=4 | sys=WD) P(1c=5 | sys=WD) P(1d=2 | sys=WD)... P(1n=2 | sys=WD) P(2a=3 | sys=WD)...P(2h=2 | sys=WD)
   Vs.
   argmax P(sys=F) P(1a=2 | sys=F) P(1b=4 | sys=F) P(1c=5 | sys=F) P(1d=2 | sys=F)... P(1n=2 | sys=F) P(2a=3 | sys=F)...P(2h=2 | sys=F)
   Vs.
   argmax P(sys=S) P(1a=2 | sys=S) P(1b=4 | sys=S) P(1c=5 | sys=S) P(1d=2 | sys=S)... P(1n=2 | sys=S) P(2a=3 | sys=S)...P(2h=2 | sys=S)
3. The maximum number of sub-department assigned to a manager accounting to all feedback forms can be said to be the most suitable sub-department for that manager.

3.4 Simple Average Calculation for Final Rating

The final rating of a manager is calculated as follows:

1. Calculate the average rating of a form.
   If textual feedback is positive,
   $$\textit{Average rating of individual form} = \Sigma(\textit{ratings of all criterias} + 5) / \textit{number of criterias} + 1$$
   If textual feedback is negative,
   $$\textit{Average rating of individual form} = \Sigma(\textit{ratings of all criterias} - 5) / \textit{number of criterias} + 1$$
2. After the calculation of the average rating of each feedback form that was filled for the manager to be evaluated, we calculate the mean of these values to get the final rating.
   $$\textit{Final rating} = \Sigma\ (\textit{Average rating of individual form}) / \textit{Total number of forms}$$
3. Following is the python code for the above algorithm

```
Import statistics
final=0
#y is the number of feedback forms
for i in xrange(1,y)
#rating_list[y] consists of ratings of all   #criteria and the rating(+5 or -5) of textual #review
    avg=statistics.mean(rating_list[y])
    final=final+avg
#n is the number of forms
final_avg=final/n
```

4 Conclusion

This paper discusses methods to analyse feedback about the managers of an organisation. The proposed methods were SVM/DT and Naive Baye's. SVM/DT mainly focused on increasing the speed of computation and Naive Baye's was implemented because it is easier to work on and requires less training data making work fast and less prone to errors. Future enhancements can be made on the hybrid model of SVM/DT to output multiclass results as current model can only give binary outputs.

5 Reference

[1] Decision Tree Learning Mitchell, Chapter 3 CptS 570 Machine Learning, School of EECS, Washington State University
[2] CS229 Lecture notes, Andrew Ng, Stanford University
[3] The SVM Approach, NPTEL, Lecture 13, Module 9
[4] Learning Maximum-Margin Hyperplanes: Support Vector Machines. Piyush Rai, Machine Learning (CS771A) , Aug 24, 2016, IIT Kharagpur.
[5] Support Vector Machines, Based on ESL (chapter 12) and papers by Vladimir Vapnik+Isabel Guyon, Trevor Hastie, Saharon Rosset, Ji Zhu, Rob Tibshirani, Stanford University.