**Sports Fantasy Application using ML Approach**

Prabhakar K

Professor

Dept of CSE

CMR University

Goutham Ganesh

Shanbhag

Student

CMR University

gouthamganesh.20bcs@cmr.edu.in

Joginder Prathap Singh N

Student

CMR University

joginder.20bcs@cmr.edu.in

Jomi Thomas

Student

CMR University

Jomi.20bcs@cmr.edu.in

K Harsh Mohan

Student

CMR University

harsh.20bcs@cmr.edu.in

1. **ABSTRACT:**

**The intersection of cricket and machine learning (ML) has given rise to a new era in fantasy sports, empowering enthusiasts to engage with the game on a more analytical and strategic level. This research presents a novel cricket fantasy application enriched with ML algorithms, focusing on the prediction of player performances and the optimization of fantasy teams. The core of the proposed application lies in the implementation of a Random Forest Regression algorithm to predict player scores based on historical performance data. Leveraging features such as previous averages, strike rates, and opponent averages, the model aims to provide accurate and dynamic predictions that adapt to the ever-changing cricketing landscape. To enhance user experience and engagement, the application incorporates a recommendation system driven by collaborative filtering. This system employs ML algorithms to analyze user preferences, historical team selections, and player interactions, ultimately suggesting optimal team compositions tailored to individual preferences, risk tolerance, and strategic objectives.**.

**Keywords**

### Random Forest Regression algorithm ,Decision Tree Algorithm ,K Means Algorithm ,KNN Algorithm ,Euclidean Distance ,Manhattan Distance ,Minkowski Distance

# INTRODUCTION

Cricket, often regarded as a game of uncertainties and strategic brilliance, has witnessed a remarkable transformation in recent years with the infusion of cutting-edge technologies. One such evolution is the integration of machine learning (ML) into cricket fantasy applications, presenting an exciting prospect for enthusiasts to engage with the sport on a more immersive and analytical level. This research embarks on a comprehensive exploration of the amalgamation of ML techniques into cricket fantasy platforms, with a primary emphasis on refining team composition strategies and dynamically predicting player performances in the context of cricket.

Cricket fantasy applications have become a cornerstone for fans looking to extend their participation beyond the stadium or television screen. Traditional approaches to fantasy team formation have relied on historical statistics, player averages, and recent performances. However, the incorporation of ML introduces a paradigm shift, allowing for the development of predictive models that can analyze a myriad of data points, including player statistics, match conditions, and evolving game scenarios.

The central objective of this research is to develop an ML-enhanced cricket fantasy application that not only enhances the accuracy of player performance predictions but also adapts dynamically to the nuances of cricket, which is inherently influenced by variables such as pitch conditions, player form, and match format. Leveraging supervised and unsupervised learning algorithms, the application aims to provide users with insights into player contributions, adaptability to different formats (Test, One Day, T20), and the ability to predict match-changing moments.

Beyond predictive analytics, the research addresses the user-centric aspect of cricket fantasy applications. A recommendation system is proposed, employing collaborative filtering and reinforcement learning algorithms to suggest optimal team compositions based on individual user preferences, risk tolerance, and strategic inclinations. This personalized approach seeks to elevate user engagement and satisfaction, catering to the diverse preferences of cricket enthusiasts.

The integration of ML in cricket fantasy applications introduces ethical considerations that warrant careful examination. Transparency, fairness, and responsible data handling emerge as pivotal elements, and this paper navigates these ethical challenges while proposing guidelines for the ethical deployment of ML in the context of cricket fantasy.

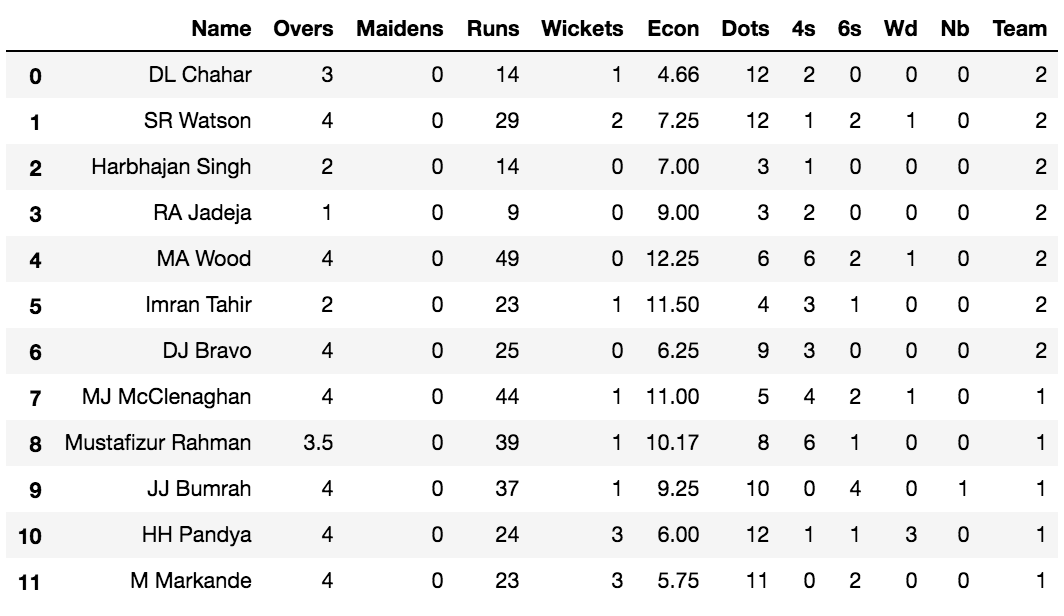
However, challenges persist in implementing ML models in a dynamic, real-time cricketing environment. This research discusses these challenges and offers insights into potential solutions, contributing to the broader discourse on the practical applications of ML in the intricate and ever-evolving landscape of cricket.

In summary, this research endeavors to advance our understanding of the synergy between ML and cricket fantasy, offering a glimpse into the future of data-driven and personalized cricket engagement. The ML-enhanced cricket fantasy application developed in this study serves as a prototype, illustrating the transformative potential of advanced analytics in reshaping the way cricket enthusiasts experience and interact with the game.

# APPROACH FOR ANALYSIS:

# 3.1 Data Scraping:

Data was extracted from ESPN webiste by running a scraping script in a justified manner



*fig 1:data scraping*

Dataset comprises of all the ODI matches from Jan 5, 1971, to Oct 29, 2017. A total of 3933 ODI match results were scrapped. The collected dataset was subjected to cleaning process where some of the matches were deleted from the analysis. Since it’s not possible to foresee the impact of nature on cricket, matches which either ended up in a tie/draw or interrupted by rain, were being removed from the dataset. Matches of special teams like World XI, Asia XI & Africa XI were also removed.

The dataset was also replicated two times by swapping the team positions i.e. a game between team 1: India and team 2: Sri Lanka was also replicated as team 1: Sri Lanka and team 2: India. For further making the dataset suitable for input to the various machine learning classifier models, the continuous dataset was converted into a categorical dataset, using dummy variables. Innings feature was determined by first translating Column:

*Margin* into Column: *Winner Innings* using:

Win by Wickets =*⇒* Winner Innings: 2

Win by Runs =*⇒* Winner Innings: 1

Further, Using Column:

*Winner* and the generated Column:

*Winner Innings*, the innings of each team per match were acquired. Venue feature was determined by using Column: *Winner* and scrapped dataframe from ESPN website which provided the names of cricket grounds in all countries. Combining both of these, Column: *Host Country* was generated, which was used to get venue of a match with respect to both the teams. The dataset was saved in comma separated format. A total of 7494 match records were used for the analytical study which was further divided into the testing and training data.

*•* Training Dataset Size: 5620

*•* Testing Dataset Size: 1874

**3.2 FEATURE ENGINEERING AND SELECTION :**

It is important to understand how certain features / variables drive the desired outcomes. The initial sets of features that make up the base datasets are depicted with values of ‘False’ in the ‘Engineered’ columns. We apply a unique procedure to the datasets for the problems to engineer some additional features that help drive the final predictions. For performance comparsion problems, the focus is on creating statistical metrics that are derived primarily from fantasy points such as average points per team, average points for a player, points variance, and minimum / maximum points. Similarly, for other feature problems, we add several statistical features that are based principally on points scored such as the average score difference for the favorite team / underdog team when they are the favorite / underdog and the average total points scored by the favorite / underdog team when they are home / away. Values of ‘True’ in the ‘Engineered’ columns signify the features added after the engineering procedures are applied for each problem. The value, *yi* , to be predicted for a given row is, for performance related problems, the fantasy points (numerical). For type other feature, *yi* indicates whether the underdog team wins the at giant stadium.

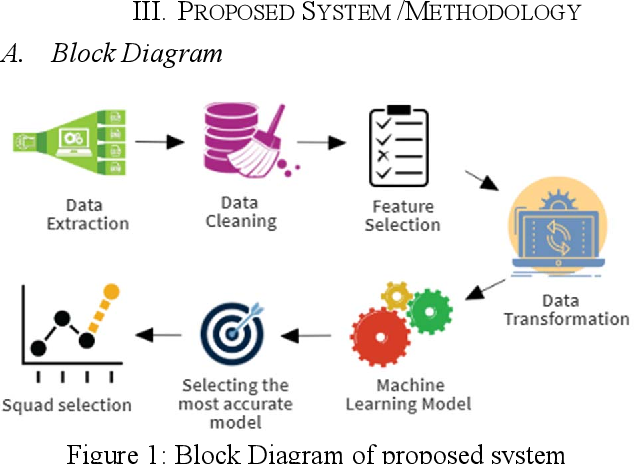
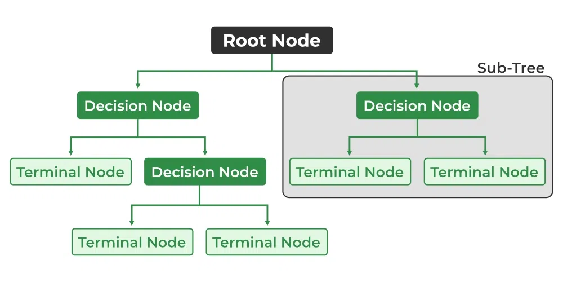


fig 2:process invovled

# Models :

## Decision Tree:

A decision tree is a flowchart-like [tree structure](https://www.geeksforgeeks.org/introduction-to-tree-data-structure-and-algorithm-tutorials/) where each internal node denotes the feature, branches denote the rules and the leaf nodes denote the result of the algorithm. It is a versatile [supervised machine-learning](https://www.geeksforgeeks.org/ml-types-learning-supervised-learning/) algorithm, which is used for both classification and regression problems. It is one of the very powerful algorithms. And it is also used in Random Forest to train on different subsets of training data, which makes random forest one of the most powerful algorithms in [machine learning](https://www.geeksforgeeks.org/machine-learning/).



*fig 3: Decision tree*

**Root Node:** It is the topmost node in the tree,  which represents the complete dataset. It is the starting point of the decision-making process.

**Decision/Internal Node:** A node that symbolizes a choice regarding an input feature. Branching off of internal nodes connects them to leaf nodes or other internal nodes.

**Leaf/Terminal Node:** A node without any child nodes that indicates a class label or a numerical value.

**Splitting:** The process of splitting a node into two or more sub-nodes using a split criterion and a selected feature.

**Branch/Sub-Tree:** A subsection of the decision tree starts at an internal node and ends at the leaf nodes.

**Parent Node:** The node that divides into one or more child nodes.

**Child Node:** The nodes that emerge when a parent node is split.

**Impurity:** A measurement of the target variable’s homogeneity in a subset of data. It refers to the degree of randomness or uncertainty in a set of examples. The **Gini index** and **entropy** are two commonly used impurity measurements in decision trees for classifications task

**Variance:** Variance measures how much the predicted and the target variables vary in different samples of a dataset. It is used for regression problems in decision trees.

**Mean squared error, Mean Absolute Error, friedman\_mse, or Half Poisson deviance** are used to measure the variance for the regression tasks in the decision tree.

**Information Gain:** Information gain is a measure of the reduction in impurity achieved by splitting a dataset on a particular feature in a decision tree. The splitting criterion is determined by the feature that offers the greatest information gain, It is used to determine the most informative feature to split on at each node of the tree, with the goal of creating pure subsets

**Pruning**: The process of removing branches from the tree that do not provide any additional information or lead to overfitting.

## K Means:

[Unsupervised Machine Learning](https://www.geeksforgeeks.org/supervised-unsupervised-learning/) is the process of teaching a computer to use unlabeled, unclassified data and enabling the algorithm to operate on that data without supervision. Without any previous data training, the machine’s job in this case is to organize unsorted data according to parallels, patterns, and variations.

The goal of [clustering](https://www.geeksforgeeks.org/clustering-in-machine-learning/) is to divide the population or set of data points into a number of groups so that the data points within each group are more comparable to one another and different from the data points within the other groups. It is essentially a grouping of things based on how similar and different they are to one another.

We are given a data set of items, with certain features, and values for these features (like a vector). The task is to categorize those items into groups. To achieve this, we will use the K-means algorithm

*fig 4: K Means Clustring*

## KNN algorithm:

K-Nearest Neighbours is one of the most basic yet essential classification algorithms in Machine Learning. It belongs to the [supervised learning](https://www.geeksforgeeks.org/supervised-unsupervised-learning/) domain and finds intense application in pattern recognition, [data mining](https://www.geeksforgeeks.org/data-mining/), and intrusion detection.

It is widely disposable in real-life scenarios since it is non-parametric, meaning, it does not make any underlying assumptions about the distribution of data (as opposed to other algorithms such as GMM, which assume a [Gaussian distribution](https://www.geeksforgeeks.org/mathematics-probability-distributions-set-3-normal-distribution/) of the given data). We are given some prior data (also called training data), which classifies coordinates into groups identified by an attribute

### **Euclidean Distance**

This is nothing but the cartesian distance between the two points which are in the plane/hyperplane. [Euclidean distance](https://www.geeksforgeeks.org/calculate-the-euclidean-distance-using-numpy/) can also be visualized as the length of the straight line that joins the two points which are into consideration. This metric helps us calculate the net displacement done between the two states of an object.

Rendered by QuickLaTeX.com

### **Manhattan Distance**

[Manhattan Distance](https://www.geeksforgeeks.org/how-to-calculate-manhattan-distance-in-r/) metric is generally used when we are interested in the total distance traveled by the object instead of the displacement. This metric is calculated by summing the absolute difference between the coordinates of the points in n-dimensions.

Rendered by QuickLaTeX.com

### **Minkowski Distance**

We can say that the Euclidean, as well as the Manhattan distance, are special cases of the [Minkowski distance](https://www.geeksforgeeks.org/minkowski-distance-python/).

Rendered by QuickLaTeX.com

From the formula above we can say that when p = 2 then it is the same as the formula for the Euclidean distance and when p = 1 then we obtain the formula for the Manhattan distance.

**4.3.3 Advantages of the KNN Algorithm**

**Easy to implement** as the complexity of the algorithm is not that high.

**Adapts Easily** – As per the working of the KNN algorithm it stores all the data in memory storage and hence whenever a new example or data point is added then the algorithm adjusts itself as per that new example and has its contribution to the future predictions as well.

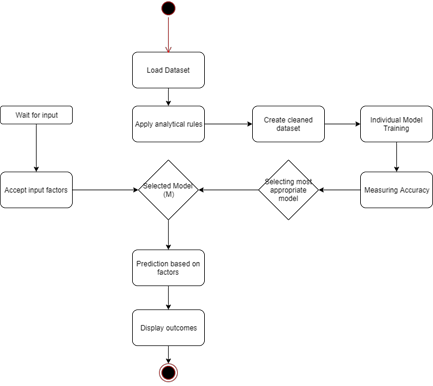
**Few Hyperparameters** – The only parameters which are required in the training of a KNN algorithm are the value of k and the choice of the distance metric which we would like to choose from our evaluation metric.

## 4.3.4 Disadvantages of the KNN Algorithm

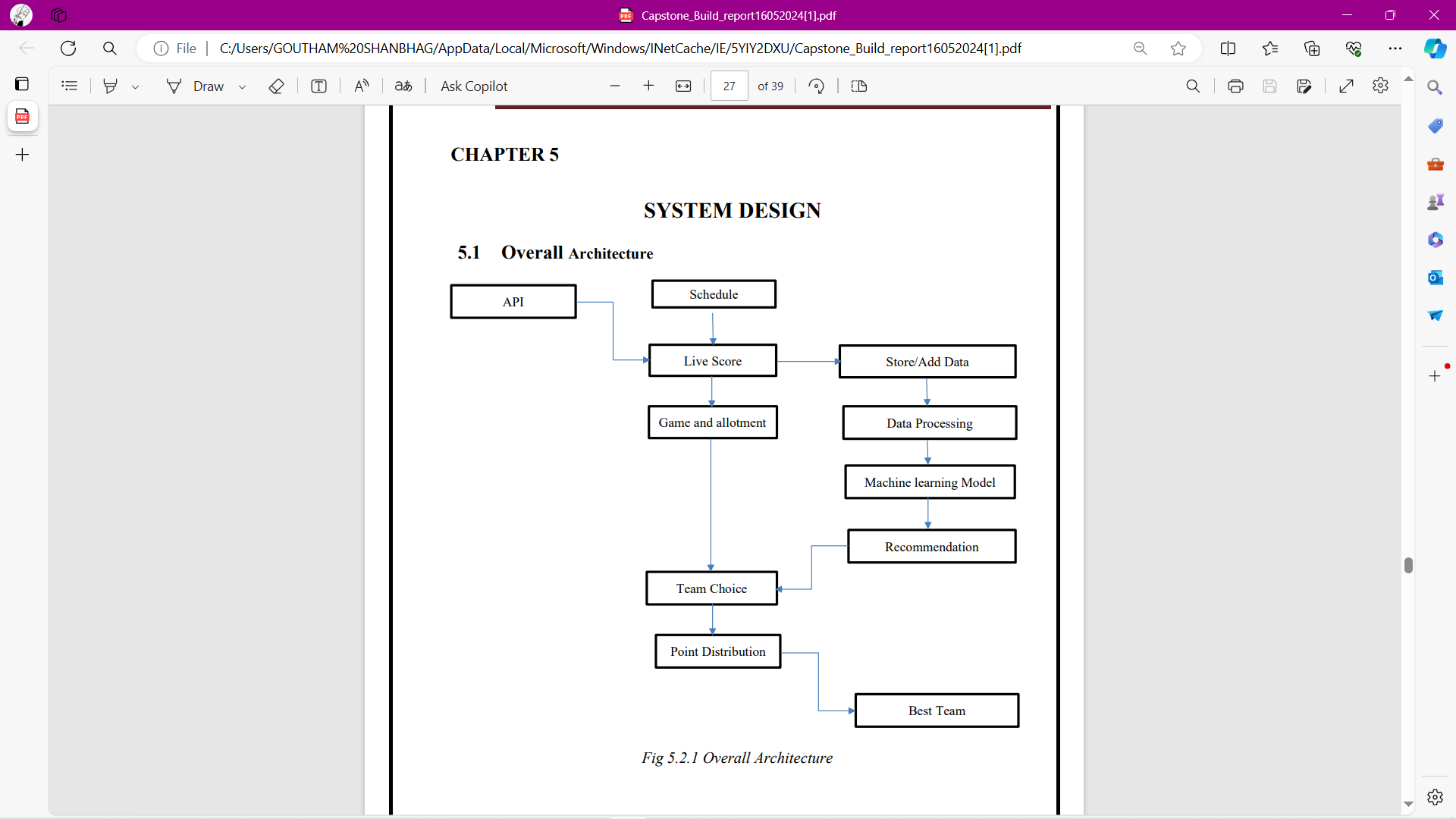
**Does not scale** – As we have heard about this that the KNN algorithm is also considered a Lazy Algorithm. The main significance of this term is that this takes lots of computing power as well as data storage. This makes this algorithm both time-consuming and resource exhausting.

**Curse of Dimensionality** – There is a term known as the peaking phenomenon according to this the KNN algorithm is affected by the [curse of dimensionality](https://www.geeksforgeeks.org/videos/curse-of-dimensionality-in-machine-learning/) which implies the algorithm faces a hard time classifying the data points properly when the dimensionality is too high.

**Prone to Overfitting** – As the algorithm is affected due to the curse of dimensionality it is prone to the problem of overfitting as well. Hence generally [feature selection](https://www.geeksforgeeks.org/feature-selection-techniques-in-machine-learning/) as well as [dimensionality reduction](https://www.geeksforgeeks.org/dimensionality-reduction/) techniques are applied to deal with this problem.



*fig 5:Work flow of the algorithm*



*fig 6:**Model Diagram*

# Results and Disscusson:

***Performance Measures***

To evaluate performance we use confusion matrix of predicted score. Values of True Positive(TP), True Negative(TN), False Positive(FP), False Negative(FN) are filled in confusion matrix.

**Confusion matrix:**

|  |  |
| --- | --- |
|  | *Predicted* |
| *Actual* |  | *Negative* | *Positive* |
| *Negative* | *True Negative* | *False Positive* |
| *Positive* | *False Negative* | *True Positive* |

 **Accuracy Score**:

Comparison of the expected outcomes with the predicted outcomes of the system for a given input dataset is done. Higher the rate of actual true labels in testing data matching with corresponding set of predicted labels, higher will be the accuracy For measuring the success of the prediction, we use precision-recall index.

 **Precision Score:**

This is defined as the number of True Positives (Tp) divided by the sum of True Positives and False Positives

(Fp)

**References:**

1. Bowman, M., Debray, S. K., and Peterson, L. L. 1993. Reasoning about naming systems. .
2. Ding, W. and Marchionini, G. 1997 A Study on Video Browsing Strategies. Technical Report. University of Maryland at College Park.
3. Fröhlich, B. and Plate, J. 2000. The cubic mouse: a new device for three-dimensional input. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems
4. Tavel, P. 2007 Modeling and Simulation Design. AK Peters Ltd.
5. Sannella, M. J. 1994 Constraint Satisfaction and Debugging for Interactive User Interfaces. DoctoralThesis. UMI Order Number: UMI Order No. GAX95-09398., University of Washington.
6. Forman, G. 2003. An extensive empirical study of feature selection metrics for text classification. J. Mach. Learn. Res. 3 (Mar. 2003), 1289-1305.
7. Brown, L. D., Hua, H., and Gao, C. 2003. A widget framework for augmented interaction in SCAPE.
8. Y.T. Yu, M.F. Lau, "A comparison of MC/DC, MUMCUT and several other coverage criteria for logical decisions", Journal of Systems and Software, 2005, in press.
9. Spector, A. Z. 1989. Achieving application requirements. In Distributed Systems, S. Mullender

**Best value: 1 and Worst value: 0.**

**Recall Score:** This is defined as the number of True Positives (Tp) divided by the sum of True Positives and False Negatives

(Fn).

**Accuracy**: Total accuracy can be easily evaluated using confusion matrix. Ratio of sum of True Positives(Tp) and True

Negative(Tn) to that of sum of True Positives(Tp) +True Negative(Tn)+False Positives(Fp) and False Negative(Fn)