Student Evaluation

UML501 Machine Learning Project Report

Submitted by: Manik Garg (101803079)

BE Third Year, 3CO10

Submitted to: Dr. Sumit Kumar

Introduction

Student academic performance is the most critical indication of educational advancement in any country. Essentially, students' academic achievement is influenced by gender, age, teaching staff, and students' learning. Predicting student academic success has gained a great deal of interest in education. In other words, student performance refers to the extent to which students achieve both immediate and long-term learning objectives [1]. Excellent academic record is an essential factor for a high-quality university based on its rankings. As a result, its ranking improves when an institution has a strong track record and academic achievements. From the student's perspective, maintaining outstanding academic performance increases the possibilities of securing employment, as excellent academic achievement is one of the primary aspects evaluated by employers.

Machine learning is a subfield of artificial intelligence (AI). The goal of machine learning generally is to understand the structure of data and fit that data into models that can be understood and utilized by people. Although machine learning is a field within computer science, it differs from traditional computational approaches. In traditional computing, algorithms are sets of explicitly programmed instructions used by computers to calculate or problem solve. Machine learning algorithms instead allow for computers to train on data inputs and use statistical analysis in order to output values that fall within a specific range. Because of this, machine learning facilitates computers in building models from sample data in order to automate decision-making processes based on data inputs. Any technology user today has benefitted from machine learning. Facial recognition technology allows social media platforms to help users tag and share photos of friends. Optical character recognition (OCR) technology converts images of text into movable type. Recommendation engines, powered by machine learning, suggest what movies or television shows to watch next based on user preferences. Self-driving cars that rely on machine learning to navigate may soon be available to consumers. Predicting and analyzing student performance are critical to assisting educators in recognizing students' weaknesses while helping them improve their grades. Likewise, students can improve their learning activities, and administrators can improve their operations. The timely prediction of student performance allows educators to identify low-performing individuals and intervene early in the learning process to apply the necessary interventions. ML is a novel approach with numerous applications that can make predictions on data. ML techniques in educational data mining aim to model and detect meaningful hidden patterns and useable information from educational contexts. Moreover, in the academic field, the ML approaches are applied to large datasets to represent a wide range of student characteristics as data points. These strategies can benefit various fields by achieving various goals, including extracting patterns, predicting behavior, or identifying trends, which allow educators to deliver the most effective methods for learning and to track and monitor the students' progress.

A number of reviews pertaining to not only the diverse factors like personal, socio-economic, psychological and other environmental variables that influence the performance of students but also the models that have been used for the performance prediction are available in the literature. Mika suggested that insufficient skills in basic mathematics caused 20 problems for any students who pursued in engineering. The research objectives are to identity the factors that contribute to the overall performance of students and to analyze any correlation between the abilities of students at entry point to the overall academic performance. Those factors can then be used as inputs to predict the overall performance achievement of students.The individual differences of each student, such aspersonality, motivation, self-efficacy, intelligence and self-control, have a close relationshipto his/her performance, so in this research, all these differences were covered by choosingthe proper features.

Background

To answer the question on the relationship between SET and academic achievement, a series of revision and meta-analytical studies have been carried out.As early as the seventies, many researchers analyzed the association between SET and student achievement. However, as Kulik and McKeachie (1975) pointed out, “the most impressive thing about studies relating class achievement to class ratings of instructors is the inconsistency of the results” (p. 235).

Cohen (1981) performed the first meta-analysis based on 68 multisection studies, in which various equivalent sections/classes follow the same outline and the same or equivalent assessments; each section is instructed by a different professor, and these professors are evaluated using students’ evaluation of teaching ratings. Cohen’s (1981) results indicated that SET scores correlated moderately with academic achievement ($r = 0.43$), concluding that these results support the validity of SET as a measure of teaching effectiveness. However, recent studies have questioned some aspects of Cohen’s (1981) meta-analysis, referring to the repeatable search strategy followed by Cohen or the sample size of sections on which Cohen’s meta-analysis studies are based, with as few as five sections (Uttl et al., 2017).

The primary objective of Feldman’s (1989) meta-analysis was to extend Cohen’s analysis of the correlation between several specific dimensions of the evaluation of the teacher’s instruction. The four dimensions most correlated with academic achievement were, in this order, preparation and organization, clarity and understandableness, perceived outcome, and teacher’s stimulation of interest in the course and its subject matter. Feldman’s (1989) results showed that the correlation between preparation and organization, the dimension most strongly correlated with academic achievement, ranged from 0.36 to 0.57. However, this meta-analysis did not account for the size of individual studies, so the moderate to high correlations may be an artifact of small-study effects.

The objectives of Clayson’s (2009) meta-analysis were to address situational questions and methodological questions. Criteria for including studies were related to college instruction, data based on multiple sections of the same course discipline, a measure of learning common across sections, a learning measure based on actual testing results and not on student perception, and SET conducted before the students took their final exam. Overall, 17 articles were included, containing 42 studies and 1,115 sections. Considering the situational dependence of previous meta-analysis on educational and/or psychological disciplines, studies were coded according to the subject matter of study.

The most extensive revision work on the relationship between the results of SET and their academic achievement is the one recently carried out by Uttl et al. (2017). On the one hand, they reanalyzed the previous meta-analyses of Cohen (1981); Feldman (1989), and Clayson (2009); on the other hand, they updated the previous meta-analyses of SET/achievement correlations included in multisection studies to date.

The present study aimed to check the relationships between SET and academic achievement, starting from the knowledge offered by previous studies. This study is carried out in a different context to most previous works. It is based in the South American country Ecuador and analyzes SET in the National Polytechnic School—a higher education institution for the study of technical subjects, such as engineering, architecture, and biotechnology. If the association between SET and academic performance is situational and not applicable to all academic disciplines, appearing stronger in studies in the field of education and the liberal arts and less in other areas such as business classes (Clayson, 2009), it seems

necessary to carry out newstudies, focusing on technical areas different to previous studies where there are fewer studies on the subject.

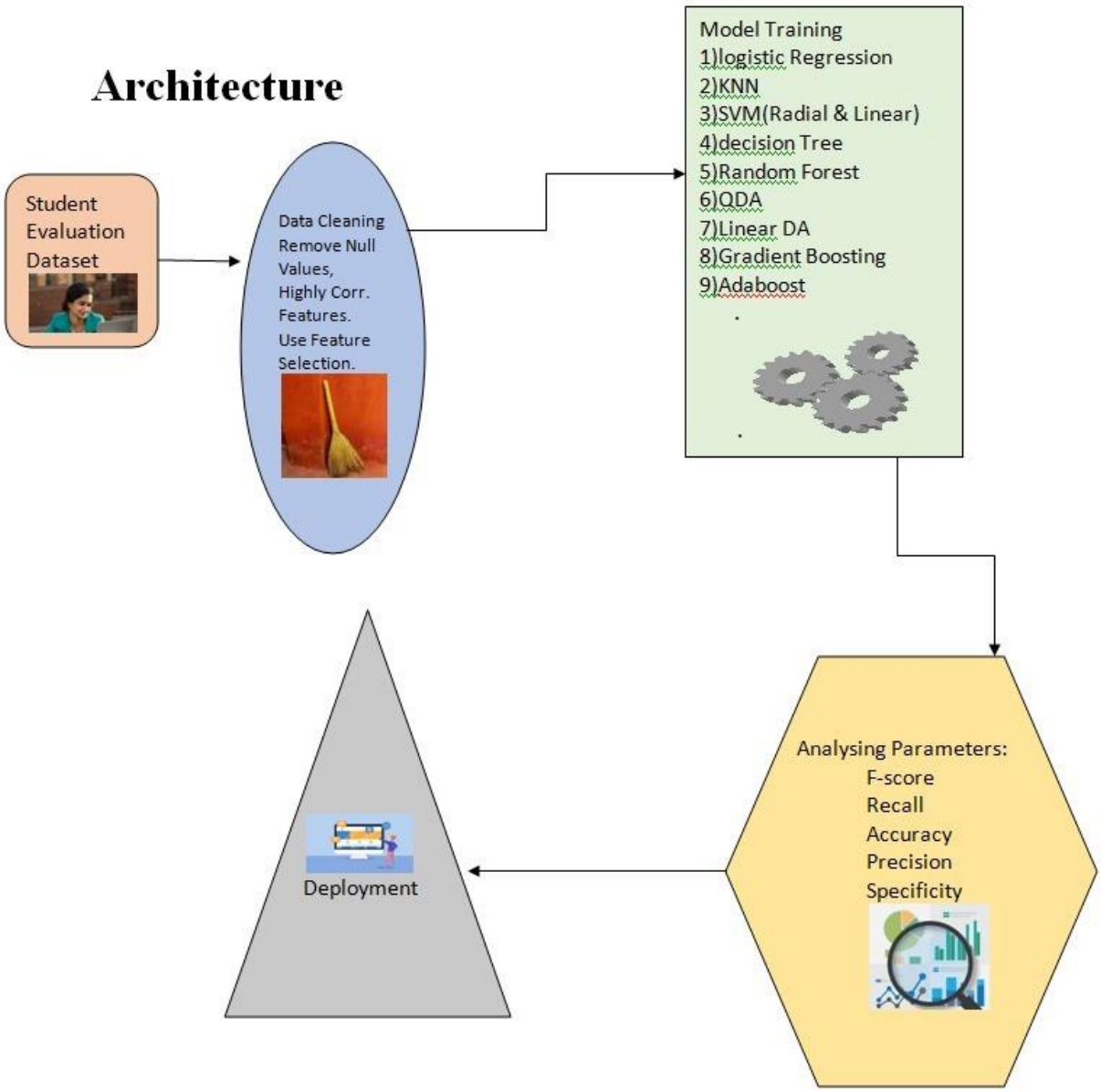


Fig 1: Architectural Diagram

Data Set Description

Problem Statement - Students has attended the multiple courses. At the end students provided the feedback on the course, instructor by answer the 30 Question. Given the feedback as a dataset, we have analyzed the data using various Clustering Techniques.

About Dataset-This dataset is based on an evaluation form filled out by students for different courses. It has different attributes including attendance, difficulty, score for each evaluation question, among others.The dataset has 5820 rows and 33 columns.

Description

	Feature	Explanation	Measurement	Range
2	Q1	hester course content, teaching method and evaluation system were provided at t	Likert-type (Categorical)	[1,5]
3	Q2	The course aims and objectives were clearly stated at the beginning of the period.	Likert-type (Categorical)	[1,5]
4	Q3	The course was worth the amount of credit assigned to it.	Likert-type (Categorical)	[1,5]
5	Q4	he course was taught according to the syllabus announced on the first day of clas	Likert-type (Categorical)	[1,5]
6	Q5	lass discussions, homework assignments, applications and studies were satisfa	Likert-type (Categorical)	[1,5]
7	Q6	The textbook and other courses resources were sufficient and up to date.	Likert-type (Categorical)	[1,5]
8	Q7	he course allowed field work, applications, laboratory, discussion and other studie	Likert-type (Categorical)	[1,5]
9	Q8	The quizzes, assignments, projects and exams contributed to helping the learning	Likert-type (Categorical)	[1,5]
10	Q9	greatly enjoyed the class and was eager to actively participate during the lectures	Likert-type (Categorical)	[1,5]
11	Q10	My initial expectations about the course were met at the end of the period or year.	Likert-type (Categorical)	[1,5]
12	Q11	The course was relevant and beneficial to my professional development.	Likert-type (Categorical)	[1,5]
13	Q12	The course helped me look at life and the world with a new perspective.	Likert-type (Categorical)	[1,5]
14	Q13	The Instructor's knowledge was relevant and up to date.	Likert-type (Categorical)	[1,5]
15	Q14	The Instructor came prepared for classes.	Likert-type (Categorical)	[1,5]
16	Q15	The Instructor taught in accordance with the announced lesson plan.	Likert-type (Categorical)	[1,5]
17	Q16	The Instructor was committed to the course and was understandable.	Likert-type (Categorical)	[1,5]
18	Q17	The Instructor arrived on time for classes.	Likert-type (Categorical)	[1,5]
19	Q18	The Instructor has a smooth and easy to follow delivery/speech.	Likert-type (Categorical)	[1,5]
20	Q19	The Instructor made effective use of class hours.	Likert-type (Categorical)	[1,5]
21	Q20	The Instructor explained the course and was eager to be helpful to students.	Likert-type (Categorical)	[1,5]
22	Q21	The Instructor demonstrated a positive approach to students.	Likert-type (Categorical)	[1,5]
23	Q22	The Instructor was open and respectful of the views of students about the course.	Likert-type (Categorical)	[1,5]
24	Q23	The Instructor encouraged participation in the course.	Likert-type (Categorical)	[1,5]
25	Q24	nstructor gave relevant homework assignments/projects, and helped/guided stud	Likert-type (Categorical)	[1,5]
26	Q25	Instructor responded to questions about the course inside and outside of the cou	Likert-type (Categorical)	[1,5]
27	Q26	ystem (midterm and final questions, projects, assignments, etc.) effectively mea	Likert-type (Categorical)	[1,5]
28	Q27	The Instructor provided solutions to exams and discussed them with students.	Likert-type (Categorical)	[1,5]
29	Q28	The Instructor treated all students in a right and objective manner.	Likert-type (Categorical)	[1,5]
30	instr	Unique Code of Instructor	Categorical	[1,3]
31	class	Course Code(descriptor)	Categorical	[1,13]
32	rerpeat	Number of times the student is taking this course	Numerical	[0,...]
33	attendance	Code of the level of attendance	Categorical	[0,4]
34	difficulty	Level of difficulty of Course as perceived by the student	Categorical	[1,5]

Preprocessing:

Preprocessing is not applied on the dataset. So we find correlation.Boxplot and count plot is also plotted.

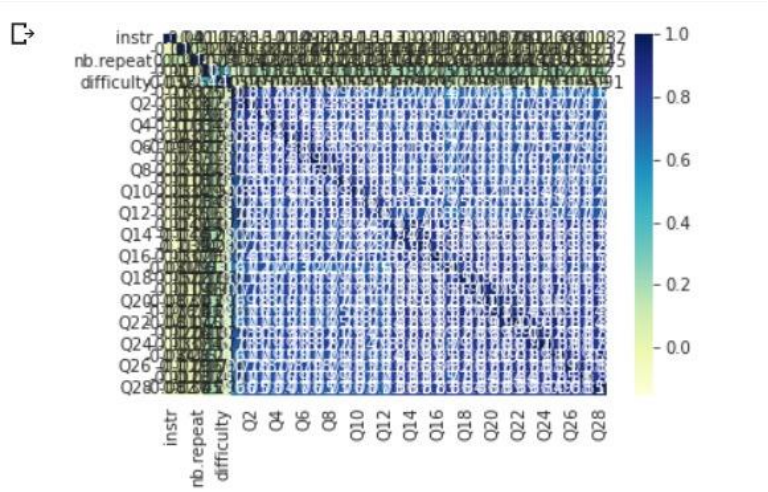


Fig2: Heatmap showing correlations among all the features of the dataset

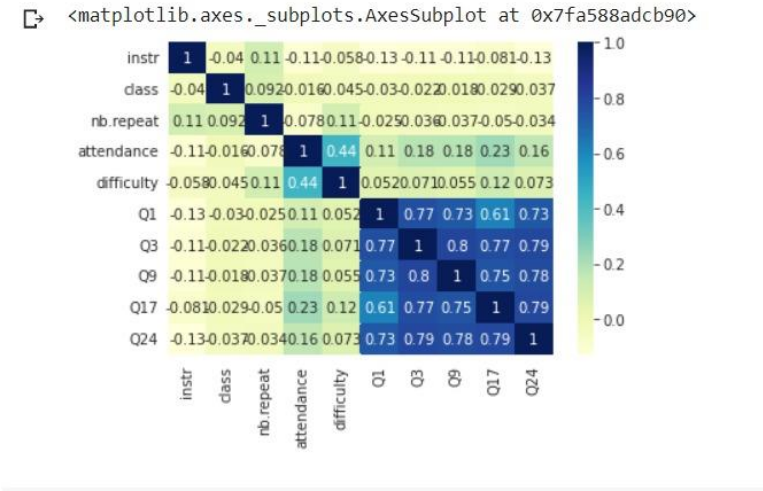


Fig3: Correlation among all the features of the dataset

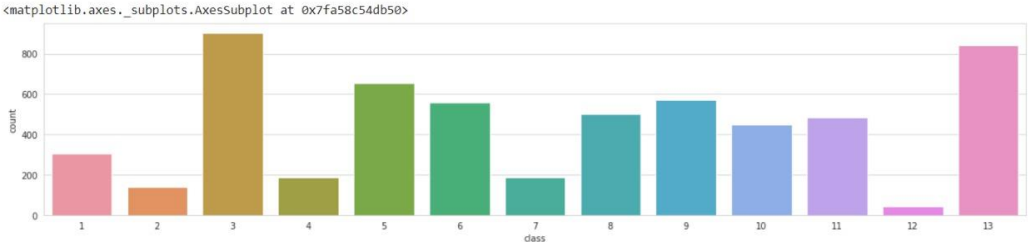


Fig4: Count Plot

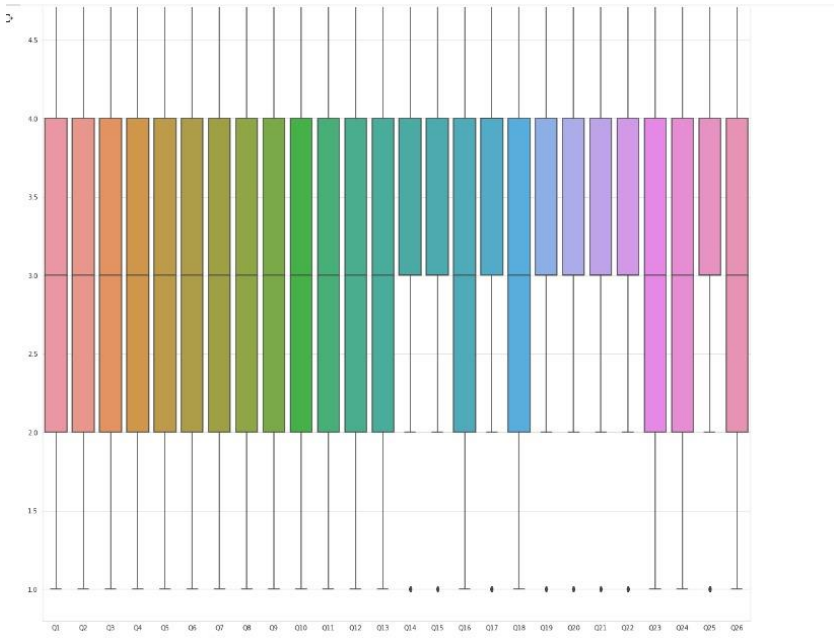


Fig5: Box Plot

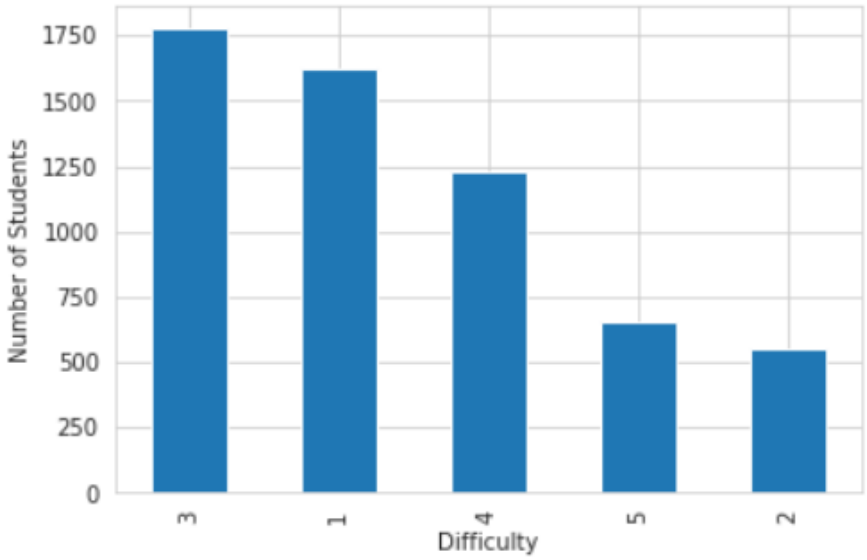


Fig6: Number of students vs Difficulty

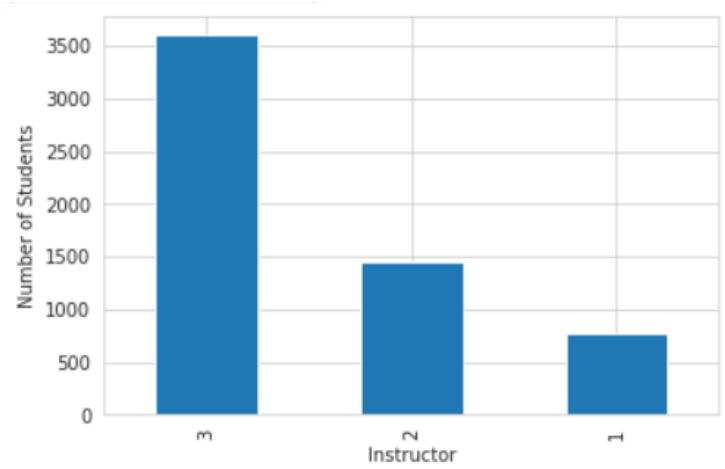


Fig7: Number of Students vs Instructor

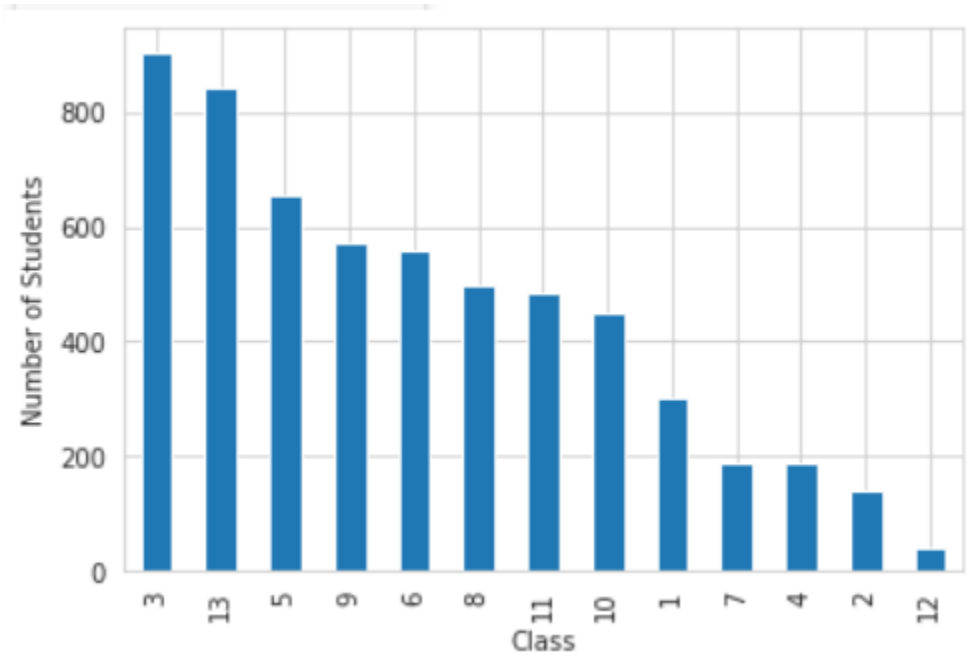


Fig8: Number of Students vs Class

Algorithms Used:

KNN CLASSIFIER

The k-nearest neighbors algorithm, also known as KNN or k-NN, is a non-parametric, supervised learning classifier, which uses proximity to make classifications or predictions about the grouping of an individual data point. While it can be used for either regression or classification problems, it is typically used as a classification algorithm, working off the assumption that similar points can be found near one another.

For classification problems, a class label is assigned on the basis of a majority vote—i.e., the label that is most frequently represented around a given data point is used. While this is technically considered "plurality voting", the term, "majority vote" is more commonly used in literature. The distinction between

these terminologies is that “majority voting” technically requires a majority of greater than 50%, which primarily works when there are only two categories. When you have multiple classes—e.g., four categories, you don’t necessarily need 50% of the vote to make a conclusion about a class; you could assign a class label with a vote of greater than 25%.

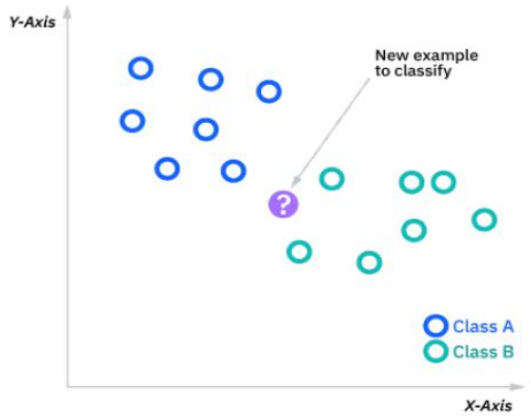


Fig9: Depicting KNN classifier

The goal of the k-nearest neighbor algorithm is to identify the nearest neighbors of a given query point, so that we can assign a class label to that point. In order to do this, KNN has a few requirements:

Determine distance metrics-

Euclidean distance (p=2):

$$d(x,y) = \sqrt{\sum_{i=1}^{n}(y_i - x_i)^2}$$

Manhattan distance (p=1):

$$d(x,y) = \left(\sum_{i=1}^{m}\left|x_i - y_i\right|\right)$$

Minkowski distance:

$$\text{Minkowski Distance} = \left(\sum_{i=1}^{n}\left|x_i - y_i\right|\right)^{1/p}$$

Hamming distance:

$$\text{Hamming Distance} = D_H = \left(\sum_{i=1}^{k}\left|x_i - y_i\right|\right)$$

$$\begin{array}{ll} x=y & D=0 \\ x\neq y & D\neq 1 \end{array}$$

The following code is an example of how to create and predict with a KNN model:

```
from sklearn.neighbors import KNeighborsClassifier
model_name = 'K-Nearest Neighbor Classifier'
knnClassifier = KNeighborsClassifier(n_neighbors = 5, metric = 'minkowski', p=2)
knn_model = Pipeline(steps= [('preprocessor', preprocessorForFeatures), ('classifier' , knnClassifier)])
knn_model.fit(X_train, y_train)
y_pred = knn_model.predict(X_test)
```

Advantages and disadvantages of the KNN algorithm:

Advantages:

-**Easy to implement**:Given the algorithm's simplicity and accuracy, it is one of the first classifiers that a new data scientist will learn.

- **Adapts easily**: As new training samples are added, the algorithm adjusts to account for any new data since all training data is stored into memory.

- **Few hyperparameters**: KNN only requires a k value and a distance metric, which is low when compared to other machine learning algorithms.

Disadvantages:

- **Does not scale well**: Since KNN is a lazy algorithm, it takes up more memory and data storage compared to other classifiers. This can be costly from both a time and money perspective. More memory and storage will drive up business expenses and more data can take longer to compute. While different data structures, such as Ball-Tree, have been created to address the computational inefficiencies, a different classifier may be ideal depending on the business problem.

- **Curse of dimensionality**: The KNN algorithm tends to fall victim to the curse of dimensionality, which means that it doesn't perform well with high-dimensional data inputs. This is sometimes also referred to as thepeaking phenomenon(PDF, 340 MB) (link resides outside of ibm.com), where after the algorithm attains the optimal number of features, additional features increase the amount of classification errors, especially when the sample size is smaller.

- **Prone to overfitting**: Due to the "curse of dimensionality", KNN is also more prone to overfitting. While feature selection and dimensionality reduction techniques are leveraged to prevent this from occurring, the value of k can also impact the model's behavior. Lower values of k can overfit the data, whereas higher values of k tend to "smooth out" the prediction values since it is averaging the values over a greater area, or neighborhood. However, if the value of k is too high, then it can underfit the data.

LINEAR SVM

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.

The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine. Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyperplane:

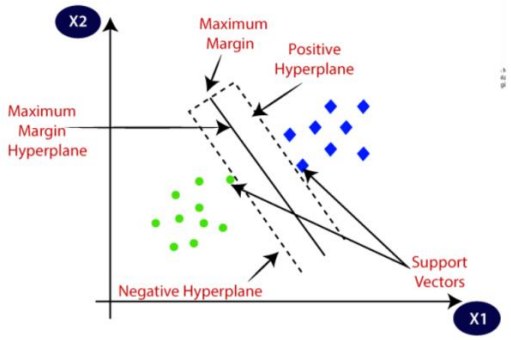


Fig10: Depicting SVM

Example: SVM can be understood with the example that we have used in the KNN classifier. Suppose we see a strange cat that also has some features of dogs, so if we want a model that can accurately identify whether it is a cat or dog, so such a model can be created by using the SVM algorithm. We will first train our model with lots of images of cats and dogs so that it can learn about different features of cats and dogs, and then we test it with this strange creature. So as support vector creates a decision boundary between these two data (cat and dog) and choose extreme cases (support vectors), it will see the extreme case of cat and dog. On the basis of the support vectors, it will classify it as a cat.

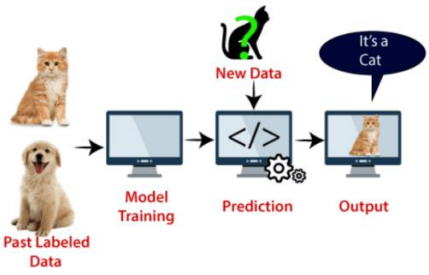


Fig11

SVM can be of two types:

- **Linear SVM:** Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.

- **Non-linear SVM:** Non-Linear SVM is used for non-linearly separated data, which means if a dataset cannot be classified by using a straight line, then such data is termed as non-linear data and classifier used is called as Non-linear SVM classifier.

Hyperplane and Support Vectors in the SVM algorithm:

Hyperplane: There can be multiple lines/decision boundaries to segregate the classes in n-dimensional space, but we need to find out the best decision boundary that helps to classify the data points. This best boundary is known as the hyperplane of SVM.

The dimensions of the hyperplane depend on the features present in the dataset, which means if there are 2 features (as shown in image), then hyperplane will be a straight line. And if there are 3 features, then hyperplane will be a 2-dimension plane.

We always create a hyperplane that has a maximum margin, which means the maximum distance between the data points.

Support Vectors:

The data points or vectors that are the closest to the hyperplane and which affect the position of the hyperplane are termed as Support Vector. Since these vectors support the hyperplane, hence called a Support vector.

Linear SVM:

The working of the SVM algorithm can be understood by using an example. Suppose we have a dataset that has two tags (green and blue), and the dataset has two features x1 and x2. We want a classifier that can classify the pair(x1, x2) of coordinates in either green or blue. Consider the below image:

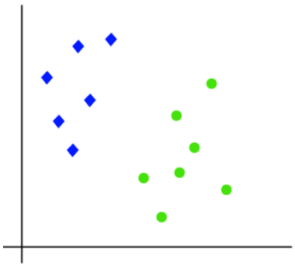


Fig12

So as it is 2-d space so by just using a straight line, we can easily separate these two classes. But there can be multiple lines that can separate these classes. Consider the below image:

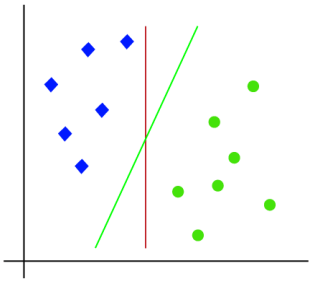


Fig13

Hence, the SVM algorithm helps to find the best line or decision boundary; this best boundary or region is called as a **hyperplane**. SVM algorithm finds the closest point of the lines from both the classes. These points are called support vectors. The distance between the vectors and the hyperplane is called as **margin**. And the goal of SVM is to maximize this margin. The **hyperplane** with maximum margin is called the **optimal hyperplane**.

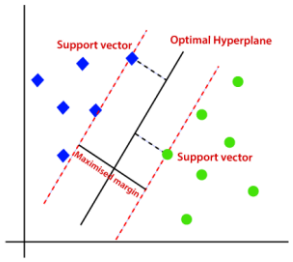


Fig14

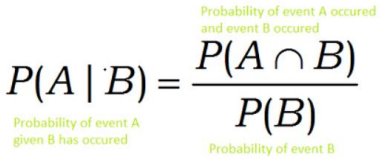
NAÏVE BAYES CLASSIFIER

This model is easy to build and is mostly used for large datasets. It is a probabilistic machine learning model that is used for classification problems. The core of the classifier depends on the Bayes theorem with an assumption of independence among predictors. That means changing the value of a feature doesn't change the value of another feature.It is called Naive because of the assumption that 2 variables are independent when they may not be. In a real-world scenario, there is hardly any situation where the features are independent. Since it is a probabilistic approach, the predictions can be made really quick. It can be used for both binary and multi-class classification problems.

Conditional Probability for Naive Bayes:

Conditional probability is defined as the likelihood of an event or outcome occurring, based on the occurrence of a previous event or outcome. Conditional probability is calculated by multiplying the probability of the preceding event by the updated probability of the succeeding, or conditional, event.

Mathematically, the conditional probability of event A given event B has already happened is given by:



Bayes Rule

Now we are prepared to state one of the most useful results in conditional probability: Bayes' Rule.Bayes' theorem which was given by Thomas Bayes, a British Mathematician, in 1763 provides a means for calculating the probability of an event given some information.Mathematically Bayes theorem can be stated as:

$$P(B|A) = \frac{P(A|B)P(B)}{P(A)}.$$

Basically, we are trying to find the probability of event A, given event B is true.

Here P(B) is called prior probability which means it is the probability of an event before the evidenceP(B|A) is called the posterior probability i.e., Probability of an event after the evidence is seen.Bayes rule provides us with the formula for the probability of Y given some feature X. In real-world problems, we hardly find any case where there is only one feature.When the features are independent, we can extend Bayes' rule to what is called Naive Bayes which assumes that the features are independent that means changing the value of one feature doesn't influence the values of other variables and this is why we call this algorithm "*NAIVE*".

When there are multiple X variables, we simplify it by assuming that X's are independent, so

$$P(Y = k|X) \ = \ \frac{P(X|Y = k) * P(Y = k)}{P(X)}$$

For n number of X, the formula becomes **Naive Bayes**:

$$P(Y = k|X1, X2 \ldots . Xn) = \frac{P(X1|Y = k) * P(X2|Y = k) \ldots \ldots * P(Xn|Y = k) * P(Y = k)}{P(X1) * P(X2) \ldots . * P(Xn)}$$

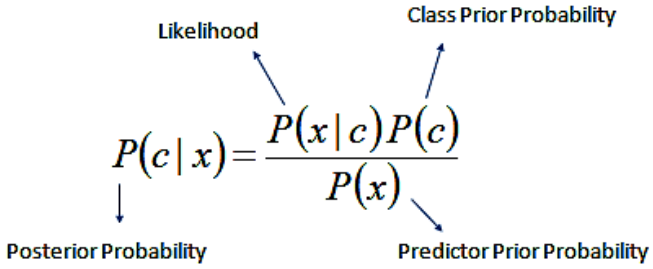
Which can be expressed as:

$$P(Y=k|X1,X2....Xn) = \frac{P(Y)\prod_{i=1}^{n} P(Xi|Y)}{P(X1)*P(X2)....*P(Xn)}$$

Since the denominator is constant here so we can remove it. It's purely your choice if you want to remove it or not. Removing the denominator will help you save time and calculations.

$$P(Y=k|X1,X2....Xn) \propto P(Y)\prod_{i=1}^{n} P(Xi|Y)$$

This formula can also be understood as:



$$P(c \mid X) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

There are a whole lot of formulas mentioned here but worry not we will try to understand all this with the help of an example.

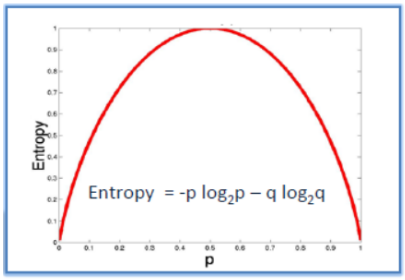
Decision Tree Using Information Gain

Decision tree builds classification or regression models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes. A decision node (e.g., Outlook) has two or more branches (e.g., Sunny, Overcast and Rainy). Leaf node (e.g., Play)

represents a classification or decision. The topmost decision node in a tree which corresponds to the best predictor called root node. Decision trees can handle both categorical and numerical data. The core algorithm for building decision trees called ID3 by J. R. Quinlan which employs a top-down, greedy search through the space of possible branches with no backtracking. ID3 uses Entropy and Information Gain to construct a decision tree. In ZeroR model there is no predictor, in OneR model we try to find the single best predictor, naive Bayesian includes all predictors using Bayes' rule and the independence assumptions between predictors but decision tree includes all predictors with the dependence assumptions between predictors.

Entropy

A decision tree is built top-down from a root node and involves partitioning the data into subsets that contain instances with similar values (homogenous). ID3 algorithm uses entropy to calculate the homogeneity of a sample. If the sample is completely homogeneous the entropy is zero and if the sample is an equally divided it has entropy of one.



Entropy = $-0.5 \log_2 0.5 - 0.5 \log_2 0.5 = 1$

To build a decision tree, we need to calculate two types of entropy using frequency tables as follows:

a) Entropy using the frequency table of one attribute:

$$E(S) = \sum_{i=1}^{c} - p_i \log_2 p_i$$

b) Entropy using the frequency table of two attributes:

$$E(T,X) = \sum_{c \in X} P(c)E(c)$$

Information Gain

The information gain is based on the decrease in entropy after a dataset is split on an attribute. Constructing a decision tree is all about finding attribute that returns the highest information gain (i.e., the most homogeneous branches).

Step 1: Calculate entropy of the target.

Step 2: The dataset is then split on the different attributes. The entropy for each branch is calculated. Then it is added proportionally, to get total entropy for the split. The resulting entropy is subtracted from the entropy before the split. The result is the Information Gain, or decrease in entropy.

$$Gain(T,X) = Entropy(T) - Entropy(T,X)$$

Step 3: Choose attribute with the largest information gain as the decision node, divide the dataset by its branches and repeat the same process on every branch.

Step 4a: A branch with entropy of 0 is a leaf node.

Step 4b: A branch with entropy more than 0 needs further splitting.

Step 5: The ID3 algorithm is run recursively on the non-leaf branches, until all data is classified.

Decision Tree to Decision Rules

A decision tree can easily be transformed to a set of rules by mapping from the root node to the leaf nodes one by one.

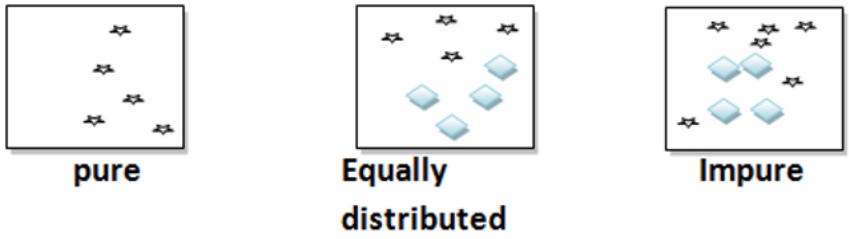
Decision Trees - Issues

- Working with continuous attributes (binning)

- Avoiding overfitting
- Super Attributes (attributes with many unique values)
- Working with missing values

Decision tree using Gini Index

In Decision Tree, the major challenge is to identify the attribute of the root node in each level. This process is known as attribute selection. We have two popular attribute selection measures:
The Gini impurity measure is one of the methods used in decision tree algorithms to decide the optimal split from a root node and subsequent splits.Gini index is also known as Gini impurity.
Gini index calculates the amount of probability of a specific feature that is classified incorrectly when selected randomly.
If all the elements are linked with a single class then it is called pure.
It ranges from 0-1
0 = all elements
1 = Randomly distributed
0.5 = equally distributed
It means an attribute with a lower Gini index should be preferred.



Equation of Gini index:

$$Gini = 1 - \sum_{i=1}^{n} (p_i)^2$$

where *pi is* the probability of an object being classified to a specific class.

How the tree is splitted?
1. Target is the decision node.
2. It is subdivided into the parent node
3. Parent node is divided into child node basing on the value of how many 1 or 0 in parent node. These are again divided into leaf node based on target=1 & target=0
(Leaf node is end node, it cannot be further divided)
Now calculate the Gini index and will find the one which factor decision is made.
The factor which gives the least Gini index is the winner, i.e,.based on that the decision tree is built.

RANDOM FOREST CLASSIFIER

A random forest is a machine learning technique that's used to solve regression and classification problems. It utilizes ensemble learning, which is a technique that combines many classifiers to provide solutions to complex problems.A random forest algorithm consists of many decision trees. The 'forest' generated by the random forest algorithm is trained through bagging or bootstrap aggregating. Bagging is an ensemble meta-algorithm that improves the accuracy of machine learning algorithms.The (random forest) algorithm establishes the outcome based on the predictions of the decision trees. It predicts by taking the average or mean of the output from various trees. Increasing the number of trees increases the precision of the outcome.

Decision trees are the building blocks of a random forest algorithm. A decision tree is a decision support technique that forms a tree-like structure. An overview of decision trees will help us understand how random forest algorithms work.A decision tree consists of three components: decision nodes, leaf nodes, and a root node. A decision tree algorithm divides a training dataset into branches, which further segregate into other branches. This sequence continues until a leaf node is attained. The leaf node cannot be segregated further.

The nodes in the decision tree represent attributes that are used for predicting the outcome. Decision nodes provide a link to the leaves. The following diagram shows the three types of nodes in a decision tree.

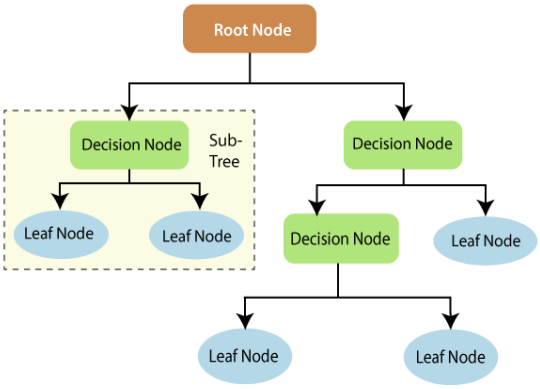


Fig15

The information theory can provide more information on how decision trees work. Entropy and information gain are the building blocks of decision trees. An overview of these fundamental concepts will improve our understanding of how decision trees are built.

Entropy is a metric for calculating uncertainty. Information gain is a measure of how uncertainty in the target variable is reduced, given a set of independent variables.

The information gain concept involves using independent variables (features) to gain information about a target variable (class). The entropy of the target variable (Y) and the conditional entropy of Y (given X) are used to estimate the information gain. In this case, the conditional entropy is subtracted from the entropy of Y.

Information gain is used in the training of decision trees. It helps in reducing uncertainty in these trees. A high information gain means that a high degree of uncertainty (information entropy) has been removed. Entropy and information gain are important in splitting branches, which is an important activity in the construction of decision trees.

Let's take a simple example of how a decision tree works. Suppose we want to predict if a customer will purchase a mobile phone or not. The features of the phone form the basis of his decision. This analysis can be presented in a decision tree diagram.

The root node and decision nodes of the decision represent the features of the phone mentioned above. The leaf node represents the final output, either *buying* or *not buying*. The main features that determine the choice include the price, internal storage, and Random Access Memory (RAM). The decision tree will appear as follows.

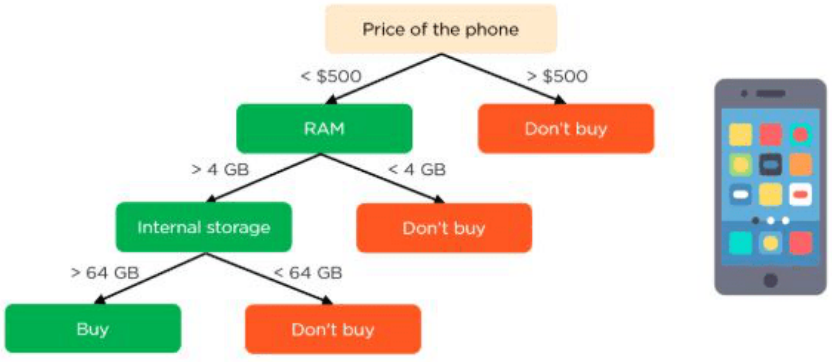


Fig16

Applying decision trees in random forest

The main difference between the decision tree algorithm and the random forest algorithm is that establishing root nodes and segregating nodes is done randomly in the latter. The random forest employs the bagging method to generate the required prediction.

Bagging involves using different samples of data (training data) rather than just one sample. A training dataset comprises observations and features that are used for making predictions. The decision trees produce different outputs, depending on the training data fed to the random forest algorithm. These outputs will be ranked, and the highest will be selected as the final output.

Our first example can still be used to explain how random forests work. Instead of having a single decision tree, the random forest will have many decision trees. Let's assume we have only four decision trees. In this case, the training data comprising the phone's observations and features will be divided into four root nodes.

The root nodes could represent four features that could influence the customer's choice (price, internal storage, camera, and RAM). The random forest will split the nodes by selecting features randomly. The final prediction will be selected based on the outcome of the four trees.

The outcome chosen by most decision trees will be the final choice. If three trees predict *buying*, and one tree predicts *not buying*, then the final prediction will be *buying*. In this case, it's predicted that the customer will buy the phone.

Classification in random forests

Classification in random forests employs an ensemble methodology to attain the outcome. The training data is fed to train various decision trees. This dataset consists of observations and features that will be selected randomly during the splitting of nodes.A rain forest system relies on various decision trees. Every decision tree consists of decision nodes, leaf nodes, and a root node. The leaf node of each tree is the final output produced by that specific decision tree. The selection of the final output follows the majority-voting system. In this case, the output chosen by the majority of the decision trees becomes the final output of the rain forest system. The diagram below shows a simple random forest classifier.

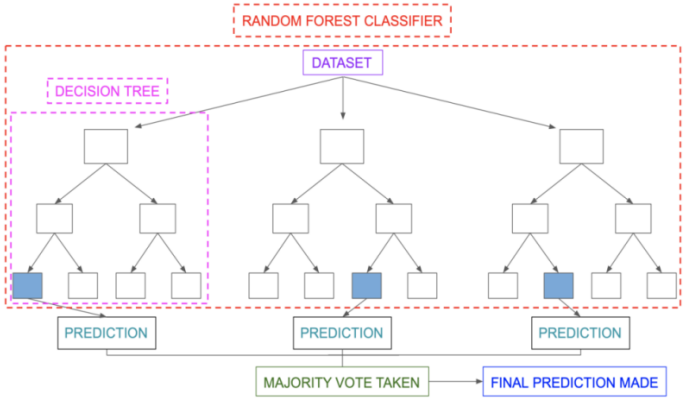


Fig17

Let’s take an example of a training dataset consisting of various fruits such as bananas, apples, pineapples, and mangoes. The random forest classifier divides this dataset into subsets. These subsets are given to every decision tree in the random forest system. Each decision tree produces its specific output. For example, the prediction for trees 1 and 2 is *apple*.Another decision tree (n) has predicted *banana* as the outcome. The random forest classifier collects the majority voting to provide the final prediction. The majority of the decision trees have chosen *apple* as their prediction. This makes the classifier choose *apple* as the final prediction.

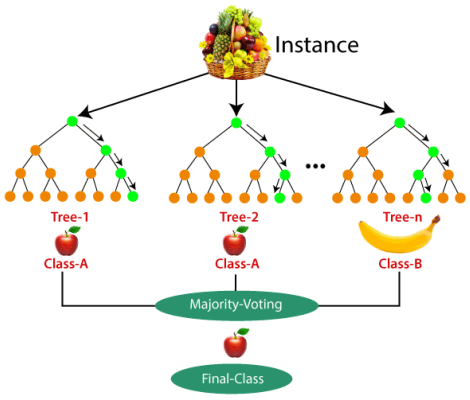


Fig18

Result

In results section, we have created this following table by running the below mentioned classifiers.

		Accuracy	Precision	Recall	F1 score	Sensitivity	Specificity	Error rate	TPR	FPR
1.	Logistic regression with regularization	0.73797251, 0.7628866 , 0.85652921, 0.65292096, 0.85395189	0.57894737, 0.24698795, 0. , 0.3507014 , 1.	0.74673629 , 0.21354167 , 0. , 0.68627451 , 0.00584795	0.65222349, 0.22905028, 0. , 0.46419098, 0.01162791	0.74673629, 0.21354167, 0. , 0.68627451, 0.00584795	0.73367478, 0.87139918, 0.996004 , 0.64356436, 1.	0.26202749, 0.2371134 , 0.14347079, 0.34707904, 0.14604811	0.74673629 , 0.21354167 , 0. , 0.68627451 , 0.00584795	0.26632522 , 0.12860082 , 0.003996 , 0.35643564 , 0.
2.	KNeighborsClassifier	0.76546392, 0.74140893, 0.80412371, 0.70876289, 0.81185567	0.61720243 , 0.27678571, 0.23129252, 0.31967213, 0.27380952	0.7513089 , 0.30845771 , 0.22818792 , 0.31075697 , 0.12707182	0.67768595, 0.29176471, 0.22972973, 0.31515152, 0.17358491	0.75130889 , 0.30845771, 0.22818792, 0.31075697, 0.12707182	0.77237852, 0.8317757 , 0.88866995, 0.81818182, 0.93794507	0.23453608, 0.25859107, 0.19587629, 0.29123711, 0.18814433	0.7513089 , 0.30845771 , 0.22818792 , 0.31075697 , 0.12707182	0.22762148, 0.1682243 , 0.11133005, 0.18181818, 0.06205493
3.	Linear SVM	0.79467354, 0.78865979, 0.87199313, 0.6073	0.66512702, 0.33082707, 0.5 , 0.32718121,	0.7539267 , 0.21890547 , 0.006	0.70674847, 0.26347305, 0.01324503, 0.4604	0.7539267 , 0.21890547 , 0.00671141, 0.7768	0.81457801, 0.90758048, 0.99901478, 0.5607	0.20532646,0.21134021, 0.12800687,0.39261168,	0.7539267 , 0.21890547 , 0.006	0.18542199 , 0.09241952 , 0.000

		8832, 0.8445 0172	0	71141 , 0.776 89243 , 0.	4864, 0	9243, 0.	8861, 1.	0.15549 828	71141 , 0.776 89243 , 0.	98522 , 0.439 21139 , 0.
4.	Naive Bayes	0.7534 3643, 0.7809 2784, 0.8539 5189, 0.6262 8866, 0.8307 5601	0.6030 3688, 0.3223 6842, 0.08 , 0.3181 8182, 0.1	0.727 74869 , 0.243 78109 , 0.013 42282 , 0.641 43426 , 0.011 04972	0.6595 4923, 0.2776 204 , 0.0229 8851, 0.4253 6328, 0.0199 005	0.7277 4869, 0.2437 8109, 0.0134 2282, 0.6414 3426, 0.0110 4972	0.7659 8465, 0.8930 4258, 0.9773 399 , 0.6221 2486, 0.9816 8871	0.2465 6357, 0.2190 7216, 0.1460 4811, 0.3737 1134, 0.1692 4399	0.727 74869 , 0.243 78109 , 0.013 42282 , 0.641 43426 , 0.011 04972	0.234 01535 , 0.106 95742 , 0.022 6601 , 0.377 87514 , 0.018 31129
5.										
6.	Decision Tree(info rmation gain)	0.7534 3643, 0.7809 2784, 0.8539 5189, 0.6262 8866, 0.8307 5601	0.6030 3688, 0.3223 6842, 0.08 , 0.3181 8182, 0.1	0.727 74869 , 0.243 78109 , 0.013 42282 , 0.641 43426 , 0.011 04972	0.6595 4923, 0.2776 204 , 0.0229 8851, 0.4253 6328, 0.0199 005	0.7277 4869, 0.2437 8109, 0.0134 2282, 0.6414 3426, 0.0110 4972	0.7659 8465, 0.8930 4258, 0.9773 399 , 0.6221 2486, 0.9816 8871	0.2465 6357, 0.2190 7216, 0.1460 4811, 0.3737 1134, 0.1692 4399	0.727 74869 , 0.243 78109 , 0.013 42282 , 0.641 43426 , 0.011 04972	0.234 01535 , 0.106 95742 , 0.022 6601 , 0.377 87514 , 0.018 31129
7.	Decision Tree(gini index)	0.8451 3173, 0.7573 8832, 0.8613 9748, 0.6263 4593, 0.8554 4101	0.7761 0819, 0.3439 5604, 1. , 0.3315 4034, 0.5	0.732 10489 , 0.403 87097 , 0.001 65017 , 0.719 74522 , 0.061 80666	0.7534 6462, 0.3715 1335, 0.0032 9489, 0.4539 6719, 0.1100 141	0.7321 0489, 0.4038 7097, 0.0016 5017, 0.7197 4522, 0.0618 0666	0.8991 1984, 0.8337 0474, 1. , 0.6006 4271, 0.9895 5544	0.1548 6827, 0.2426 1168, 0.1386 0252, 0.3736 5407, 0.1445 5899	0.732 10489 , 0.403 87097 , 0.001 65017 , 0.719 74522 , 0.061 80666	0.100 88016 , 0.166 29526 , 0. , 0.399 35729 , 0.010 44456
8.	Random Forest	0.7929 5533, 0.8238 8316, 0.8719 9313, 0.5687 2852,	0.6613 2723, 0.4705 8824, 0, 0.3095 5994, 0	0.756 5445 , 0.159 20398 , 0. , 0.812	0.7057 3871, 0.2379 1822, 0, 0.4483 5165, 0	0.7565 445 , 0.1592 0398, 0. , 0.8127 49 , 0.	0.8107 4169, 0.9626 1682, 1. , 0.5016 4294, 1.	0.2070 4467, 0.1761 1684, 0.1280 0687, 0.4312 7148,	0.756 5445 , 0.159 20398 , 0. , 0.812	0.189 25831 , 0.037 38318 , 0. , 0.498

		0.8445 0172		749 , 0.				0.1554 9828	749 , 0.	35706 , 0.

In results section, we have created this following table by running the below mentioned classifiers.

		Accuracy	Precision	Recall	F1 score	Sensitivity	Specificity	Error rate	TPR	FPR
1.	Logistic regression with regularization	0.432	0.4348	0.3406	0.2712	0.3406	0.8487	0.568	0.3406	0.1508
2.	KNeighborsClassifier	0.416	0.391	0.416	0.398	0.416	0.85	0.584	0.416	0.187
3.	Linear SVM	0.454	0.4555	0.3512	0.2885	0.3512	0.8563	0.546	0.3512	0.1435
4.	Naive Bayes	0.423	0.2846	0.3273	0.2808	0.3273	0.8480	0.577	0.3273	0.1515
5.	Decision Tree(information gain)	0.423	0.2846	0.3273	0.2808	0.3273	0.8480	0.577	0.3273	0.1515
6.	Decision Tree(gini index)	0.473	0.5904	0.387	0.3386	0.387	0.864	0.549	0.387	0.675
7.	Random Forest	0.451	0.480	0.576	0.464	0.576	0.758	0.549	0.576	0.242

Confusion Matrix:

KNN:

```
array([[287,  32,  19,  32,  12],
       [ 53,  62,  25,  43,  18],
       [ 26,  37,  34,  42,  10],
       [ 53,  58,  41,  78,  21],
       [ 46,  35,  28,  49,  23]])
```

Random Forest

```
array([[289,  24,   0,  69,   0],
       [ 40,  32,   0, 129,   0],
       [ 26,   2,   0, 121,   0],
       [ 41,   6,   0, 204,   0],
       [ 41,   4,   0, 136,   0]])
```

Naïve Bayes

```
array([[278,  34,   8,  54,   8],
       [ 50,  49,   7,  88,   7],
       [ 31,  20,   2,  95,   1],
       [ 54,  31,   3, 161,   2],
       [ 48,  18,   5, 108,   2]])
```

Decision Tree using Information Gain

```
array([[278,  34,   8,  54,   8],
       [ 50,  49,   7,  88,   7],
       [ 31,  20,   2,  95,   1],
       [ 54,  31,   3, 161,   2],
       [ 48,  18,   5, 108,   2]])
```

Linear SVM

```
array([[288,  40,   0,  54,   0],
       [ 51,  44,   1, 105,   0],
       [ 21,  12,   1, 115,   0],
       [ 37,  19,   0, 195,   0],
       [ 36,  18,   0, 127,   0]])
```

Logistic Regression with Regularisation

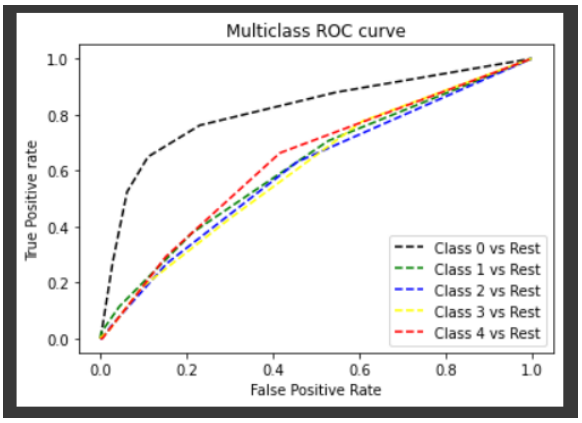
```
array([[286,  45,   0,  52,   0],
       [ 71,  41,   3,  77,   0],
       [ 48,  28,   0,  87,   0],
       [ 56,  23,   1, 175,   0],
       [ 33,  29,   0, 108,   1]])
```

Decision Tree using Gini Index

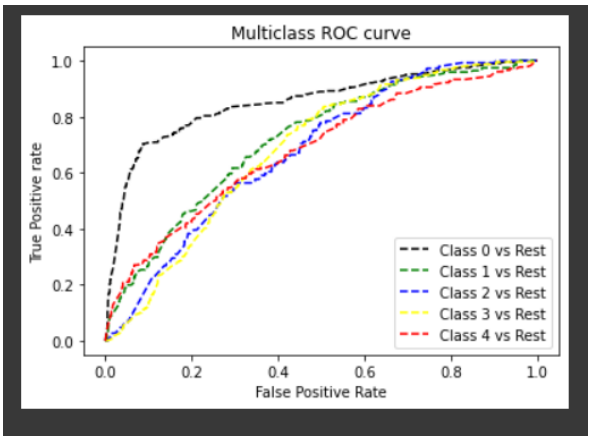
```
array([[1033,  176,    0,  192,   10],
       [  83,  313,    0,  372,    7],
       [  56,  151,    1,  389,    9],
       [  88,  163,    0,  678,   13],
       [  71,  107,    0,  414,   39]])
```

ROC Curves:

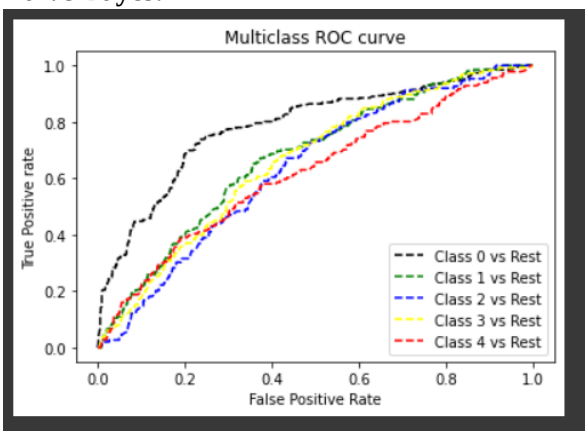
KNN:



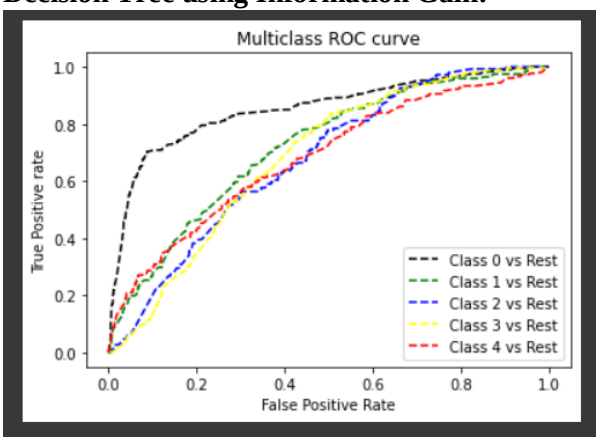
Random Forest:



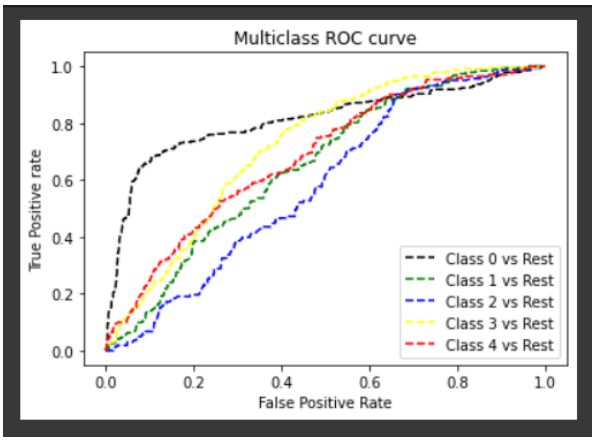
Naïve Bayes:



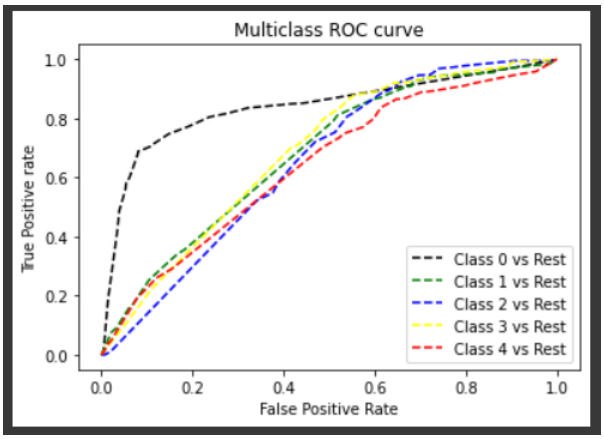
Decision Tree using Information Gain:



Logistic Regression with Regularisation:



Decision Tree using Gini Index:



Conclusion

The major purpose of this study was to determine theattendance of a student based on the questions answered by each student. Many different algorithms were applied to predict the result, but the algorithm that most appropriately predicted the correct values is linear SVM. Unlike logistic regression and other algorithms, SVMs are designed to generate more complex decision boundaries. An LS-SVM with a simple linear kernel corresponds to a linear decision boundary. Instead of a linear kernel, more complex kernel functions, such as the commonly used RBF kernel, can be chosen. The reason why SVMs work well with high dimensional data is that they are automatically regularized and regularization is a way to prevent overfitting with high dimensional data. The accuracy of the model using linear SVM is approximately 87%, which means that if a student answers all the questions about the instructor and courses then this model may predict the attendance of the student. However, a word of caution should be noted. The data in this study were gathered from just one course in oneuniversity. So, we are not generalizing the results to other courses, lecturers, or universities across the nation. Plus, the accuracy of the data provided by the respondents depends on their honesty as well as their understanding of the

questions asked. The respondents' skills of evaluating based on the criteria are different which is due to theirpersonal judgment of the scale used in the questions.Lastly, limited time and cost are the constraint faced when conducting this study where only two classes are selected and thus, the result may not as accurate and reliable.

References

https://archive.ics.uci.edu/ml/datasets/Turkiye+Student+Evaluation

https://www.hindawi.com/journals/cin/2022/4151487/

https://link.springer.com/article/10.1007/BF00991617

https://scholar.google.com/scholar_lookup?title=Faculty%20ratings%20and%20student%20grades%3A%20A%20university-wide%20multiple%20regression%20analysis&journal=Journal%20of%20Educational%20Psychology&volume=68&pages=573-577&publication_year=1976&author=Brown%2CD.%20L.

https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

https://towardsdatascience.com/logistic-regression-using-python-sklearn-numpy-mnist-handwriting-recognition-matplotlib-a6b31e2b166a