Student Grade Prediction Using C4.5 Decision Tree

Submitted by:
Bala Harimani
Scott Maresh

Department of Computer Science
Wayne State University
Detroit, MI 48201, USA

December 10th, 2020

Submitted to Dr. Suzan Arslanturk
in partial fulfillment of the requirements for
CSC 5800: Intelligent Systems: Algorithms and Tools

Overview of the Problem

A student's ability to perform well in school as a child is an important factor in determining how successful they will be later life. Students who earn higher grades in school will, generally speaking, have more opportunities as adults than those who earn lower grades. Teachers and school administrators want to help students perform well in their formative years to put them in a strong position to do well later in life. This project aims to predict which students will earn lower marks in the early stages of the semester with the help of algorithms that we have learned about in this class so that teachers and school staff can intervene early and hopefully help them perform better in school. One such algorithm that will be employed is the C4.5 decision tree. We will also try other algorithms for comparison.

Dataset Overview, Preprocessing, and Visualization

Dataset Overview

The dataset we analyzed was found on Kaggle and was originally collected from a learning multi-agent learning management system known as Kalboard 360. The dataset contains records for 480 students. These records are classified by how well the student performed with respect to their overall grade and are categorized as low-level (L) if their score is less than 70, middle-level (M) if they scored at least 70 but less than 90, or has high-level (H) if they scored 90 or higher. [1,2,3]

Below is a table of the 16 attributes of each record:

Attribute Title	Description	Possible Values
Gender	Gender of the student	1. Male 2. Female
Nationality	Nationality of the student	1. Kuwait 2. Lebanon 3. Egypt 4. SaudiArabia 5. USA 6. Jordan 7. Venezuela 8. Iran 9. Tunis 10. Morocco 11. Syria 12. Palestime 13. Iran 14. Lybia
Place of Birth	Nation of student's birth	1. Kuwait 2. Lebanon 3. Egypt 4. SaudiArabia 5. USA 6. Jordan 7. Venezuela 8. Iran 9. Tunis 10. Morocco

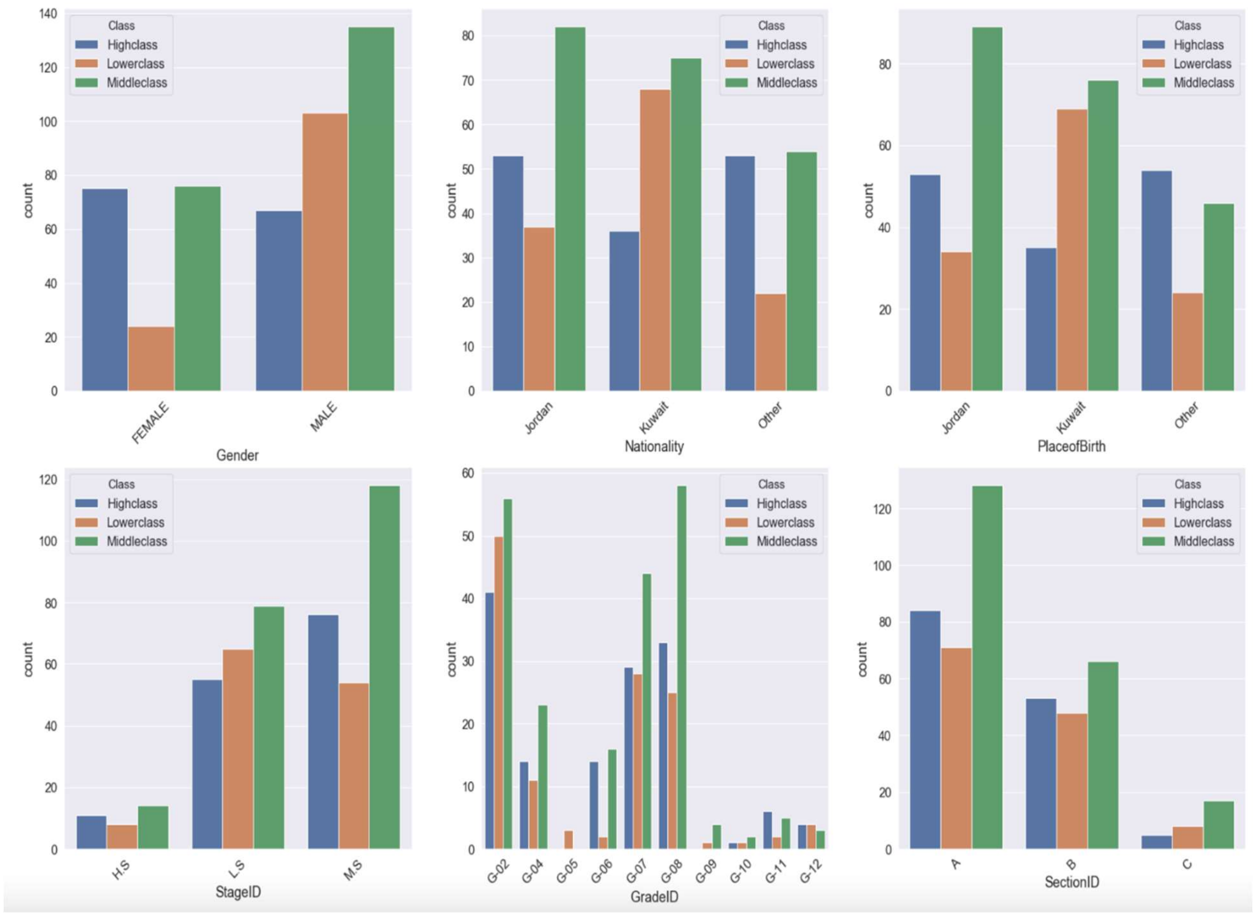
		11. Syria 12. Palestime 13. Iran 14. Lybia
Educational Stage	Educational level to which student belongs	1. LowerLevel 2. MiddleSchool 3. HighSchool
Grade Level	Grade to which student belongs	1. G-01 2. G-02 3. G-03 4. G-04 5. G-05 6. G-06 7. G-07 8. G-08 9. G-09 10. G-10 11. G-11 12. G-12
Section ID	Classroom to which student belongs	1. A 2. B 3. C
Topic	Course topic	1. English 2. Spanish 3. French 4. Arabic 5. IT 6. Math 7. Chemistry 8. Biology 9. Science 10. History 11. Quran 12. Geology
Semester	Semester of the school year	1. First 2. Second
Parent Responsible for Student	Parent responsible for student's education	1. Mom 2. Father
Raised Hand	How many times student raised their hand in class	Numeric values: 0-100
Visited Resources	How many times student visited course content online	Numeric values: 0-100
Viewing Announcements	How many times student checked new announcements	Numeric values: 0-100
Discussion Groups	Number of time student participated in discussion groups	Numeric values: 0-100
Parent Answering Survey	Whether the parent answered survey provided by the school	1. Yes 2. No
Parent School Satisfaction	Whether the parent is satisfied with the school	1. Yes 2. No
Student Absence Days	Number of days the student was absent	1. Above-7 2. Under-7

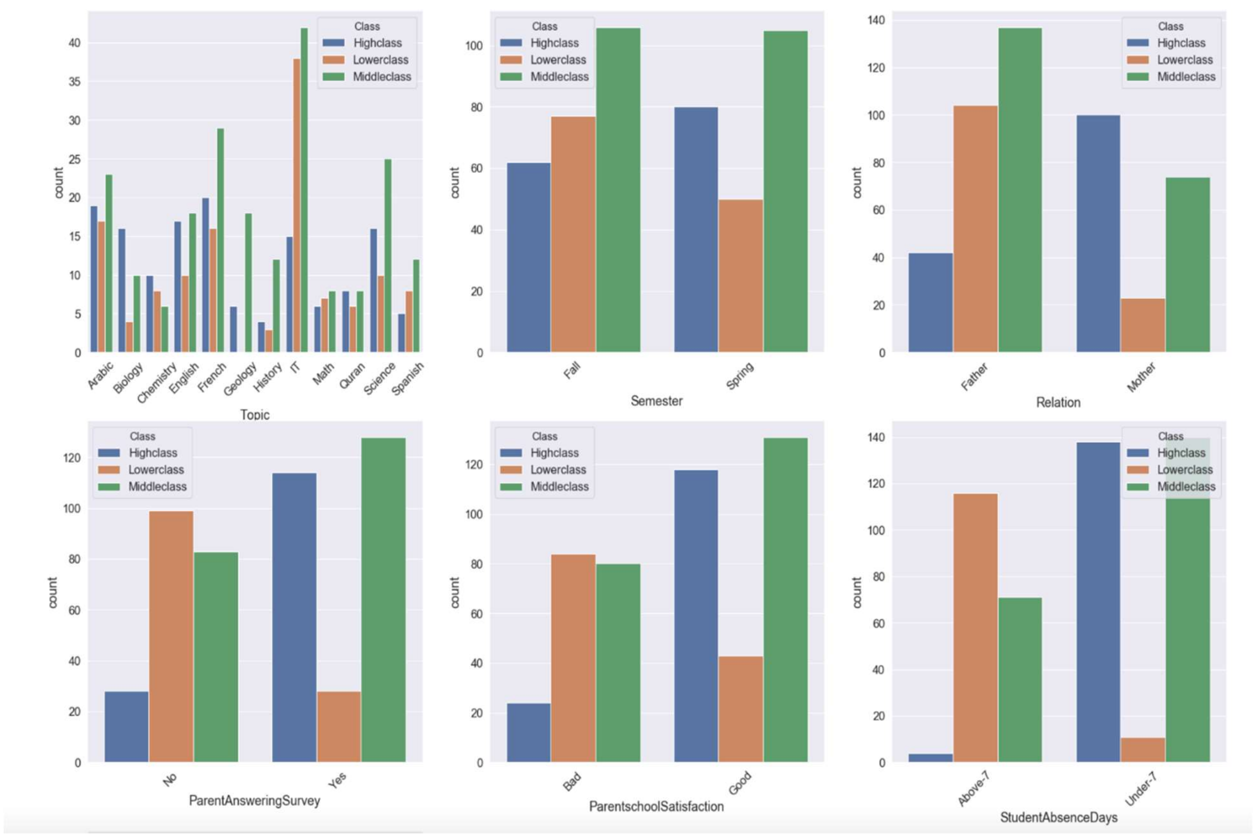
Dataset Preprocessing

Fortunately, the dataset was not missing any values, so the preprocessing step was very easy and straightforward. However, there were a few steps we took. In the “nationality” and “place of birth” variables, there were many possible values, but the majority had values of “Kuwait” and “Jordan” with very few in all other categories. Because there was such little data, we replaced all other values with “other” to make the data a little more straightforward and prevent the classifiers from over-fitting the data. Additionally, we removed the “grade levels” attribute because the data was redundant as the “educational stage” attribute already contains this data. Additionally, we removed the “topic” attribute as there were many possible values objects could take and some of it was too sparse for use for a classifier.

Dataset Visualization

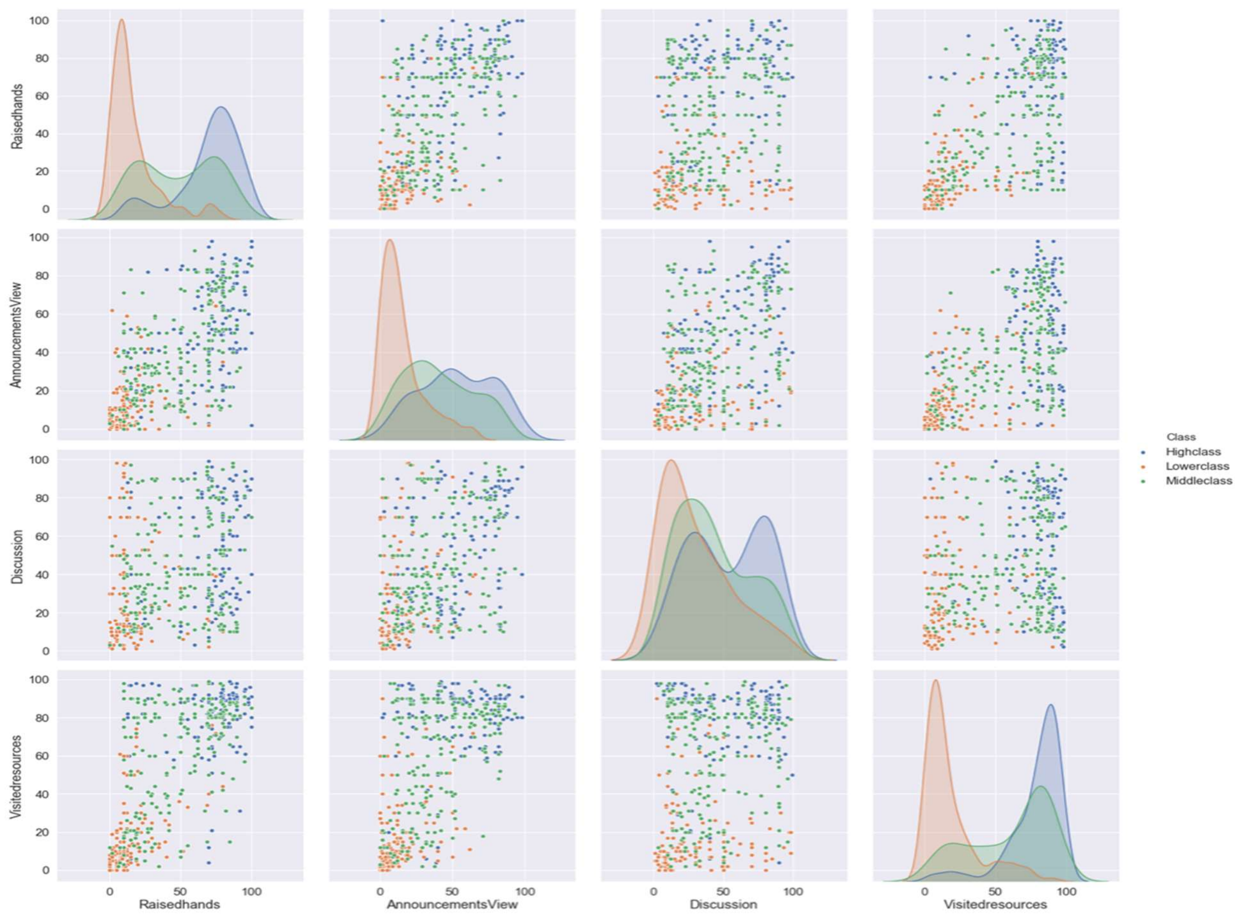
Before creating algorithms, we decided to create some graphs to get a better idea about the relationships between the different attributes and the classes. Below we created histograms for the categorical data:





The gender histogram shows that female students tend to perform better than male students. Students from Jordan are majorly in the middle class. The nationality and place of birth histograms show similar results, with those from Jordan tend to outperform those from Kuwait. Students with mothers who were more involved with their education seemed to perform much better than those with fathers who were more involved. It also seems as though students with parents who answered the survey and those with parents who were more satisfied with the school outperformed those who with parents who did not, but care should be taken when looking for a causal relationship, as it is possible that parents may be more likely to be unsatisfied with the school if their child is not performing well. The most striking subplot is that of the number of days absent, as those students who were absent less frequently seemed to heavily outperform those that were absent more frequently.

For the numerical attributes such as raised hands, viewed announcements, visited resources and discussion groups, we created scatterplots and used separate symbols to denote the separate classes in the figure below:



As to be expected, in most of these plots the blue dots representing high-performing students are concentrated in the upper-righthand corners and the orange dots representing the low-performing students are concentrated in the lower-lefthand corner. This seems to indicate that students that are more engaged with the learning process tend to out-perform those that are not, which is an unsurprising conclusion. The plots along the diagonal line seem to indicate that low-performing students are less engaged across the board, but it seems as though having high scores with respect to visiting online resources and participating in discussion are stronger predictors of high-performing students than having high scores for raising their hands and viewing announcements.

Algorithm Selection

C4.5 Decision Tree

As mentioned previously, the goal of this project is to create a C4.5 decision tree to make accurate predictions about student performance. The C4.5 algorithm is a commonly used technique to create a decision tree. It is an extension of the ID3 method of creating a decision tree, which is a more primitive method that uses entropy to determine the best splits at each node of the tree. [4]

Below is the equation for calculating entropy:

$$Entropy(t) = -\sum_{j} p(j|t) * log_2(p(j|t))$$

In this equation, p(j|t) is the relative frequency of class j at node t. The entropy is minimized (and equal to zero) if all records belong to the same class. This value is maximized if all records are equally distributed among all classes, a case in which the split gives no information to the user. [5]

When determining which split is best for a particular node, the following equation is used to calculate the information gain:

$$GAIN_{Split} = Entropy(p) - \left(\sum_{i} \frac{n_i}{n} Entropy(i)\right)$$

This equation measures the difference in the entropy of the unsplit measure node and the sum of all entropies of the children resulting from i partitions. This technique is powerful in that it helps to quantify how useful each split is in terms of giving the user information, but an important drawback is that it tends to lead to splits that result in a large number of small, pure partitions. In some models, this will likely lead to over-fitting. [5]

To overcome this, the algorithm employs what is known as the GainRATIO, as defined below:

$$GainRATIO_{Split} = \frac{GAIN_{Split} =}{-\sum_j \frac{n_i}{n} * log_2(\frac{n_i}{n})}$$

The term in the denominator is know as the “SplitINFO” term, and it applies a penalty for splitting the node into multiple partitions. [4]

The above concepts apply to both ID3 and C4.5, but C4.5 has a couple of important advantages over ID3. First, C4.5 can handle continuous numerical attributes whereas ID3 cannot. This is important when analyzing the “Raised hand,” “Viewing announcements,” “Visited resources,” and “Discussions groups” variables. This method of handling continuous numerical attributes involves performing splits at every possible instance that could produce different results for the algorithm. C4.5 also introduces a concept known as pruning. In the process of pruning, the error for the node is calculated before and after splitting, and in the case of the latter, there is a penalty constant that is multiplied by the number of splits and added to the error in order to increase it. If the increased error resulting from this penalty is greater than the original amount of error, the node is pruned. This process starts from the bottom of the decision tree and is propagated upward. [4,5]

RIPPER

In addition to the C4.5 algorithm, we wanted to use a rules-based classifier as well. There are two major classes of methods one could employ to create a rule set: indirect methods and direct methods. Indirect methods involve creating a decision tree and extracting rules from it. WEKA has a module that would allow us to do this based on a C4.5 decision tree known as PART. However, because the rule-based classifier starts with the creation of a C4.5 decision tree, the resulting algorithm would be incredibly similar to our main decision tree. [5]

Direct methods of producing rule sets, by contrast, start from an empty rule set, grow rules using a learn-one-rule function, remove training records covered by the rule, and repeat the process until a stopping criterion is met. [5]

One such method, known as the RIPPER method, uses FOIL gain to determine which rules should be added and in what order, as shown below:

$$Gain(R_0, R_1) = t * \left(log(\frac{p_1}{p_1 + n_1}) \; - \; log(\frac{p_0}{p_0 + n_0}) \right)$$

In the above equation, R_0 refers to the initial rule and R_1 refers to the rule after adding the conjunct, t refers to the number of instances covered by both rules, p_0/p_0 and n_0/n_1 refer to the number of positive and negative instances covered by the two rules, respectively. Rules are created until the gain is not significant. [5]

Pruning can also be applied to this technique in a manner similar to what was described in the previous section for C4.5. [5]

For each rule, the following expression is evaluated:

$$v = \frac{p \; - \; n}{p + n}$$

In this equation, p represents the positive examples covered by the rule and n represents the negative examples covered by the rule. Any rule that maximizes v is pruned. [5]

For a multi-class problem like the one we have analyzed, the RIPPER method is used to learn rules for the classes in ascending order – that is, rules for the smallest class are learned first. In each case, objects that fit the class being learned are considered positive and objects in all other classes are considered to be negative. [5]

Nearest Neighbor

In addition to implementing a decision tree and a rule-based classifier, we also wanted to implement a lazy learner. The nearest neighbor algorithm is one such classifier. This classifier uses a set of records, a distance metric to calculate the distance between data points, and a value of k, which is an integer that describes the number of nearest elements to retrieve with respect to an unknown record. The nearest neighbor classifier differs from the other classifiers discussed up until this point in that it does not actually produce a model to follow – hence the type "lazy learner." [5]

There are several different options for selecting the aforementioned distance metric. The most straightforward methods would be to simply calculate the Manhattan distance or the Euclidean distance for this metric. Once this metric is chosen, it is also possible to add distance weighting for the k nearest neighbor metric. If an odd number of nearest neighbors is selected, it is possible to implement a simple "majority vote," or to use this distance weighting to add more weight to closer data points. This weighting is implemented by using a 1/distance or 1/distance^2 transformations. [5]

K*

In addition to the nearest neighbor algorithm, we wanted to try applying another lazy learner algorithm. We found the KStar algorithm in WEKA and decided to learn more about it. K* is similar to the nearest neighbor classifier in that it uses a distance algorithm but is different in that it measures entropy to calculate the distance. This entropy is not the same entropy that is described in the section on the C4.5 decision tree. Instead, it is defined as the complexity of transforming one point into another. This complexity is converted into a probability that such a transformation could occur. When new a new point is classified, it is compared to all points in the data set using this entropic metric. For each point in each class in the training set, the probability of the transformation of the new point into the existing points is summed. The class that has the highest summed probability is considered to be the class of the new data point. This algorithm is quite complex, but it has a handful of advantages over the nearest neighbors algorithm in that it can handle symbolic, real-valued, and missing data, although that does not help with respect to this particular dataset. [6,7]

Naïve Bayes

We also decided to fit a naïve Bayes classifier to our data. Unlike many other classifiers, the naïve Bayes classifier assigns probabilities to each test data point belonging to a particular class and hence gives "soft" assignments. [5]

Bayes theorem, pictured below, give the framework for how this classifier works:

$$P(C|A) = \frac{P(A|C) * P(C)}{P(A)}$$

This equation states that the probability of C given A is equal to the probability of A given C multiplied by the probability of C divided by the probability of A. In the case a naïve Bayes classifier, A represents all values for all attributes of the test point and C represents the class for which we are assessing appropriateness. In instances in which models have continuous numerical values, it is necessary to discretize the range or perform two-way splits. [5]

The naïve Bayes classifier is useful in that it is robust to noise points. This is a major advantage over the nearest neighbor method. It also does well with handling missing values (as these values are ignored) and it is robust to irrelevant attributes. However, one major weakness is that it assumes independence between attributes. In our case, this assumption is almost certainly false. For example, students that raise their hands more frequently during class are probably also more likely to participate in discussion groups. [5]

Random Forest

The last model we wanted to explore with this project is the random forest classifier. This will be the second decision tree classifier that will be fitted to the data. Instead of simply creating one decision tree, this algorithm comes up with many – hence the name "random forest." These trees are made by constructing a bootstrap sample and using the sample to create a decision tree. In this tree, every internal node selects an attribute from a randomly-selected set and chooses the one attribute from this sent that shows a maximum reduction in an impurity measure. This process is repeated until every leaf is pure. [7]

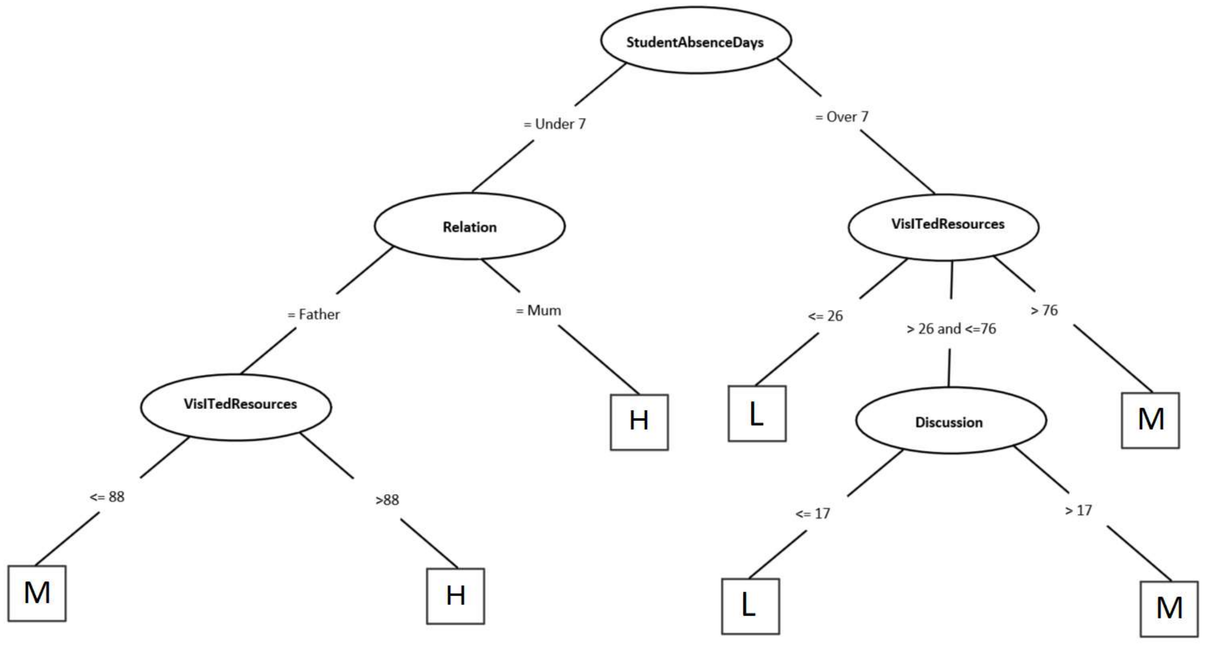
After this forest have been created, new instances are classified via "popular vote" among all decision trees. None of these trees are pruned. As a result, these trees have low bias but high variance. One strength of this method is that the trees lack correlation, as each tree chooses a different splitting criterion from its subset of attributes. [7]

Analysis Results and Comparison

Below are the results of all six algorithms. All were completed using WEKA and 10 fold cross-validation. Of utmost significance is the recall value of the L class, as we want a model that misses lower-performing students as infrequently as possible.

C4.5 Decision Tree

We implemented the C4.5 algorithm using the J48 classifier in WEKA. There are a number of settings that can be adjusted when running this algorithm, but chief among them are "confidenceFactor" and "minNumObj." The "confidenceFactor" variable is used in pruning, with lower values resulting in more aggressive pruning. The "minNumObj" variable dictates the minimum number of objects in a particular leaf. By setting minNumObj to 1 and confidence factor to 0.99, the classifier has an accuracy of 75% when using 10-fold cross-validation, but the resulting tree has 129 leaves. A tree this large may be accurate, but it would be cumbersome to use in practice and relying on some of the leaves that were produced with only a small number of values may be dubious. On the other hand, increasing the minNumObj value to 50 and decreasing the confidenceFactor value to 0.001 leads to a tree of four leaves that has an accuracy 69.4%. This is easier to use, but it is less accurate and does not make use of much of the data provided. Ultimately, we settled on a tree with a minNumObj value of 10 and confidenceFactor of 0.001. Below is the resulting tree:



Perhaps the most noteworthy feature of the resulting tree is that top branch involves the variable "StudentAbsenceDays." If the student as absent more than seven days during the semester, the resulting branches lead only to M and L. Otherwise, the resulting branches lead to M and H. Attendance seems to be a key factor in how well a student performs. It is also interested to note that students whose mother is responsible for education tend to do better than those in which the father is responsible. The tree also indicates that students that visit online resources related to their class and participate in discussion more frequently tend to get higher grades. None of this is surprising, but it is useful information nonetheless.

The overall accuracy of this model is 71.25%. Below is the confusion matrix:

		Classification		
		H	M	L
Actual	H	116	26	0
	M	60	132	19
	L	1	32	94

It is important to note that while there is a moderate amount of misclassification between the M and H pair and the L and H pair, there is only one instance of a student in the L class that was classified as H and zero cases of a student in the H class being classified as L.

Below is more detailed information regarding several parameters related to the model's performance:

Class	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area
H	0.180	0.655	0.817	0.727	0.602	0.846	0.607
M	0.216	0.695	0.626	0.658	0.416	0.733	0.680
L	0.054	0.832	0.740	0.783	0.714	0.924	0.760

There is a fair amount to unpack in the table above, but perhaps the most striking observation is the ROC area for the L class. An ROC area of 0.924 is very good, indicating that this this model, on average, strikes a very good balance between producing true positives and true negatives. However, the recall value of the L class leaves something to be desired. It is concerning that the model misclassifies one out of four students who struggle.

RIPPER

As with C4.5, we implemented the RIPPER algorithm using WEKA. The classifier used was JRIP. When running this algorithm, there are similar considerations when trying to come up with a model – we wanted a model that was not over-fitted to the data, but also accurate. In the interest of comparing the results of this model to those of C4.5, we wanted to choose parameters that would make the resulting ruleset similar to the aforementioned tree. To accomplish this, we increased a variable called minNo from 2.0 to 4.0. This variable dictates the minimum weight of the instances in a rule. Below is the resulting rule set:

(*VisITedResources<=27*) & (StudentAbsenceDays = Above-7) => Class=L

(ParentschoolSatisfaction = Bad) & (gender = M) and (raisedhands <= 12) => Class=L

(AnnouncementsView <= 10) & (Discussion <= 12) => Class=L

(StudentAbsenceDays = Above-7) & (ParentAnsweringSurvey = No) & (Discussion <= 17) => Class=L

(raisedhands >= 72) & (VisITedResources >= 89) => Class=H

(Relation = Mum) & (StudentAbsenceDays = Under-7) => Class=H

=> Class=M

The resulting rule set is a little more cumbersome to use than the C4.5 decision tree, but it reveals similar patterns with respect to how frequently the student is absent, how often they visit online resources, and how often they participate in discussion groups. Other variables that impact student performance include whether the parents are satisfied with the school, if the student views announcements, whether or not the parent answers a satisfaction survey, and how often the student raises their hand in class.

The overall accuracy of this model is 70.625%, which is comperable to that of the C4.5 model. Below is the confusion matrix:

		Classification		
		H	M	L
Actual	H	94	47	1
	M	46	144	21
	L	0	26	101

Even though the overall accuracy is comparable to the C4.5 decision tree, the confusion matrix reveals something interesting; this model is more accurate in identifying students in the M and L class and less accurate in identifying students in the H class. Since the purpose of this modelling is to identify students who are struggling and help them, this is an important advantage.

Below is a more detailed breakdown of the accuracy sorted by class:

Class	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area
H	0.136	0.671	0.622	0.667	0.528	0.832	0.618
M	0.271	0.644	0.682	0.673	0.410	0.732	0.636
L	0.062	0.821	0.795	0.808	0.741	0.917	0.772

This table seems to confirm the observation noted from the confusion matrix. The recall for the L class is slightly higher than that of the C4.5 model. This indicates that this model is less likely to miss a struggling student. Obviously, it would be ideal to increase the recall of the L class even higher if possible.

Nearest Neighbor

We implemented the nearest neighbor algorithm using the IBk module in WEKA. We selected 5 to be the value of K as we wanted there to be some consensus when assigning points and we also used 1/distance weighting.

One weakness of this method of classification is that it does not produce a model that can be followed. This is not a problem with respect to implementation, but a model can be informative to the user (for example, the C4.5 tree starts with the StudentAbsenceDays variable, indicating to the user that the number of days the student is absent is important in determining if a child may struggle).

The accuracy of the model is 72.08%. Below is the confusion matrix:

		Classification		
		H	M	L
Actual	H	98	43	1
	M	46	139	26
	L	2	16	109

The confusion matrix indicates that the nearest neighbor model does better with respect to identifying L and M class students than the C4.5 decision tree, but it underperforms with respect to identifying H class students.

Below is a more detailed breakdown of the accuracy sorted by class:

Class	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area
H	0.142	0.671	0.690	0.681	0.544	0.877	0.754
M	0.219	0.702	0.659	0.680	0.443	0.778	0.689
L	0.076	0.801	0.858	0.829	0.765	0.948	0.858

Thus far, this model has the best recall value for the L class. This is perhaps the most valuable statistic for comparing these models, but it is jarring to note how large of a drop in the recall for the H class this model has compared to the C4.5 tree (a drop of ~10%). This would be important to note if the educators would like to identify high-performing students instead of lower-performing ones.

K*

We decided to also try the K* algorithm as a second lazy learner algorithm to compare against the nearest neighbor algorithm. This was implemented via the KStar module in WEKA. The globalBlend value was kept at the default value of 20. The resulting accuracy of the model is 73.13%, which is good relative to the other models and slightly higher than that of the nearest neighbor model. Below is the confusion matrix:

		Classification		
		H	M	L
Actual	H	103	37	2
	M	40	141	30
	L	1	19	107

As they are both lazy learner models, it is not surprising that the confusion matrix for K* is similar to that of the nearest neighbor model, doing better with respect to the M/L classes and doing worse with respect to the H class. Below is a more detailed breakdown of the accuracy for the three different classes:

Class	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area
H	0.121	0.715	0.725	0.720	0.602	0.904	0.769
M	0.208	0.716	0.668	0.691	0.464	0.806	0.774
L	0.091	0.770	0.843	0.843	0.731	0.948	0.851

While these values are slightly different than those of the nearest neighbor model, the resulting profile is very similar.

Naïve Bayes

In order to create a naïve Bayes classifier, we used WEKA's NaiveBayes module. This module is very straightforward to use with few options that have any impact on the actual results on the resulting classifier itself, largely due to how straightforward the naïve Bayes classifier is. Like some of the other classifiers discussed thus far, this module does not actually create a tree or model, as it just calculates probabilities and determines if each point follows predicted behavior assuming independence. This model has the lowest accuracy, being only 68.3%. Below is the confusion matrix:

		Classification		
		H	M	L
Actual	H	107	32	3
	M	59	110	42
	L	0	16	111

Although the previously-mentioned accuracy is lower than the other models, it is important to note that most of this error is derived from misclassifications of students that belong to the M class and that the model is stronger with respect to correctly classifying students in the H and L classes. Below is a more detailed breakdown of the accuracy sorted by class:

Class	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area
H	0.175	0.645	0.754	0.695	0.556	0.884	0.763
M	0.178	0.696	0.521	0.596	0.362	0.785	0.694
L	0.127	0.712	0.874	0.784	0.703	0.950	0.872

One of the primary values we have been comparing between all of these models, the L class recall, is very high relative to the other models. However, it is important to know that the model is less precise for this class, indicating that there are more false positives when using this model compared to the others. This would be important to keep in mind if the cost to help struggling students is high. As mentioned previously, it is also important to note that the F-measure of the M class is quite low, indicating that model's precision and recall are poor for this class.

Random Forest

We implemented the random forest algorithm using the RandomForest module in WEKA. There was an option to limit the depth of the created decision trees, but we decided not to place a limit on the depth as the random forest algorithm uses multiple unpruned trees that vote on the final classification of an instance.

WEKA produces too many trees to be included in this report, but the accuracy of the model was 81.7%, which is the best accuracy of all classifiers. The confusion matrix is below:

		Classification		
		H	M	L
Actual	H	110	32	0
	M	23	174	14
	L	0	19	108

As expected with the accuracy being as high as it is, the confusion matrix shows that most instances are classified correctly.

Below is a more detailed breakdown of the accuracy sorted by class:

Class	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area
H	0.068	0.827	0.775	0.800	0.721	0.936	0.864
M	0.190	0.773	0.825	0.798	0.632	0.878	0.832
L	0.04	0.885	0.850	0.867	0.821	0.970	0.901

Across the board, most of these values are the best of all models discussed in this paper. Of note, the recall for the L class is very high, indicating that only 15% of lower performing students would be missed by this model. The model is also more precise in detecting students in the H and M classes, which would give the user more confidence when determining how well a student is likely to perform.

Conclusion

As we mentioned previously, one of our primary focuses was to minimize the recall of the L class, as we wanted to have as few false negatives as possible so that students who need help will not be missed. In this respect, the best model is the naïve Bayes classifier with a recall of 87.4%, followed closely by the nearest neighbor classifier with a recall of 85.8% and the random forest classifier with a recall of 85.0%. Going by this metric alone, the naïve Bayes classifier is the best choice. It is important to note that school district's goal with respect to this modelling could have a drastic effect on which classifier would be preferrable, especially since the naïve Bayes classifier was the least accurate model overall.

If the goal of the district is to accurately predict which students fall into each of the tree classes, the random trees model best, as it is the most accurate overall. If the school district wanted to implement a solution in which high-performing student are paired with low-performing students, the precision of the models with respect to identifying high-performing students would then become much more important, as it is important to ensure that the number of "false positives" with respect to identifying these high-performing students is minimized. Inaccurately identifying high-performing students would result in the low-performing students not getting the help that they need. The naïve Bayes classifier is actually the worst model in this respect, as the precision for identifying high-performing students is the lowest at 64.5%. The best model for implementing this solution is the random forest classifier, as it has the highest precision for identifying high-performing students (82.7%) and still have high recall for identifying low-performing students.

It is unfortunate that the primary analysis method explored in this paper, the C4.5 decision tree, performed the worst with respect to minimizing the recall of identifying low-performing students. That being said, it is important to note that this is not an indictment of this method. It may work much better on other datasets and it does have a few advantages. The chief advantage it haves over most of the other classifiers listed here is that it produces a model that is informative for educators. Additionally, the model had the second-highest f-measure for the high-performing student class (0.727), losing only to the random-forest classifier (0.800).

Works Cited

Amrieh, E. A., Hamtini, T., & Aljarah, I. (2016). Mining Educational Data to Predict Student's academic Performance using Ensemble Methods. International Journal of Database Theory and Application, 9(8), 119-136. [1]

Amrieh, E. A., Hamtini, T., & Aljarah, I. (2015, November). Preprocessing and analyzing educational data set using X-API for improving student's performance. In Applied Electrical Engineering and Computing Technologies (AEECT), 2015 IEEE Jordan Conference on (pp. 1-5). IEEE. [2]

"Students' Academic Performance Dataset." *Kaggle.* Web. 21 Nov 2020. <https://www.kaggle.com/aljarah/xAPI-Edu-Data> [3]

"Building Classification Models: ID3 and C4.5." *College of Science and Technology Temple University.* Web. 27 Nov 2020. <https://cis.temple.edu/~giorgio/cis587/readings/id3-c45.htmlData> [4]

Arslanturk, S. (2020). *CSC 5800 Lectures and Notes*. Wayne State University. [5]

"K*: An Instance-based Learner Using an Entropic Distance Measure." *The University of Qaikato.* Web. 25 Nov 2020. <https://www.cs.waikato.ac.nz/ml/publications/1995/Cleary95-KStar.pdf> [6]

Vijayarani, S & Muthulakshmi, M. (2012). Comparative Analysis of Bayes and Lazy Classification Algorithms. International Journal of Advanced Research in Computer and Communication Engineering, 2(8), 3118-3124. [7]

Tan, P, Steinbach, M, Kumar, V, & Karpatne, A. *Introduction to Data Mining*. United States: Pearson, 2019. Print. [8]

Appendix

C4.5 Decision Tree WEKA Output

```
=== Run information ===

Scheme:      weka.classifiers.trees.J48 -C 0.001 -M 10
Relation:    Other Data Rmove Topic Grade ID
Instances:   480
Attributes:  15
             gender
             NationalITy
             PlaceofBirth
             StageID
             SectionID
             Semester
             Relation
             raisedhands
             VisITedResources
             AnnouncementsView
             Discussion
             ParentAnsweringSurvey
             ParentschoolSatisfaction
             StudentAbsenceDays
             Class
Test mode:   10-fold cross-validation

=== Classifier model (full training set) ===

J48 pruned tree
------------------

StudentAbsenceDays = Under-7
|   Relation = Father
|   |   VisITedResources <= 88: M (109.0/27.0)
|   |   VisITedResources > 88: H (36.0/12.0)
|   Relation = Mum: H (144.0/47.0)
StudentAbsenceDays = Above-7
|   VisITedResources <= 26: L (103.0/11.0)
|   VisITedResources > 26
|   |   VisITedResources <= 76
|   |   |   Discussion <= 17: L (15.0/2.0)
|   |   |   Discussion > 17: M (40.0/13.0)
|   |   VisITedResources > 76: M (33.0/1.0)

Number of Leaves  :     7

Size of the tree :      13


Time taken to build model: 0 seconds
```

```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         342               71.25   %
Incorrectly Classified Instances       138               28.75   %
Kappa statistic                          0.5608
Mean absolute error                      0.257
Root mean squared error                  0.373
Relative absolute error                 59.3663 %
Root relative squared error             80.169  %
Total Number of Instances              480

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.626    0.216    0.695      0.626    0.658      0.416    0.733     0.680     M
                 0.817    0.180    0.655      0.817    0.727      0.602    0.846     0.607     H
                 0.740    0.054    0.832      0.740    0.783      0.714    0.924     0.760     L
Weighted Avg.    0.713    0.162    0.719      0.713    0.712      0.550    0.817     0.680

=== Confusion Matrix ===

   a   b   c   <-- classified as
 132  60  19 |   a = M
  26 116   0 |   b = H
  32   1  94 |   c = L
```

RIPPER WEKA Output

```
=== Run information ===

Scheme:       weka.classifiers.rules.JRip -F 3 -N 4.0 -O 2 -S 1
Relation:     Other Data Rmove Topic Grade ID
Instances:    480
Attributes:   15
              gender
              NationalITy
              PlaceofBirth
              StageID
              SectionID
              Semester
              Relation
              raisedhands
              VisITedResources
              AnnouncementsView
              Discussion
              ParentAnsweringSurvey
              ParentschoolSatisfaction
```

```
            StudentAbsenceDays
            Class
Test mode:    10-fold cross-validation

=== Classifier model (full training set) ===

JRIP rules:
===========

(VisITedResources <= 27) and (StudentAbsenceDays = Above-7) => Class=L (105.0/12.0)
(ParentschoolSatisfaction = Bad) and (gender = M) and (raisedhands <= 12) => Class=L (13.0/1.0)
(AnnouncementsView <= 10) and (Discussion <= 12) => Class=L (7.0/2.0)
(StudentAbsenceDays = Above-7) and (ParentAnsweringSurvey = No) and (Discussion <= 17) =>
Class=L (9.0/1.0)
(raisedhands >= 72) and (VisITedResources >= 89) => Class=H (42.0/4.0)
(Relation = Mum) and (StudentAbsenceDays = Under-7) => Class=H (122.0/45.0)
 => Class=M (182.0/36.0)

Number of Rules : 7


Time taken to build model: 0.01 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         339               70.625  %
Incorrectly Classified Instances       141               29.375  %
Kappa statistic                          0.5461
Mean absolute error                      0.2549
Root mean squared error                  0.3865
Relative absolute error                 58.8823 %
Root relative squared error             83.0845 %
Total Number of Instances              480

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.682    0.271    0.664      0.682    0.673      0.410    0.732     0.636     M
                 0.662    0.136    0.671      0.662    0.667      0.528    0.832     0.618     H
                 0.795    0.062    0.821      0.795    0.808      0.741    0.917     0.772     L
Weighted Avg.    0.706    0.176    0.708      0.706    0.707      0.532    0.810     0.667

=== Confusion Matrix ===

   a   b   c   <-- classified as
 144  46  21 |   a = M
  47  94   1 |   b = H
  26   0 101 |   c = L
```

Nearest Neighbor WEKA Output

```
=== Run information ===

Scheme:       weka.classifiers.lazy.IBk -K 5 -W 0 -I -A "weka.core.neighboursearch.LinearNNSearch -A
\"weka.core.EuclideanDistance -R first-last\""
Relation:     Other Data Rmove Topic Grade ID
Instances:    480
Attributes:   15
              gender
              NationalITy
              PlaceofBirth
              StageID
              SectionID
              Semester
              Relation
              raisedhands
              VisITedResources
              AnnouncementsView
              Discussion
              ParentAnsweringSurvey
              ParentschoolSatisfaction
              StudentAbsenceDays
              Class
Test mode:    10-fold cross-validation

=== Classifier model (full training set) ===

IB1 instance-based classifier
using 5 inverse-distance-weighted nearest neighbour(s) for classification


Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         346               72.0833 %
Incorrectly Classified Instances       134               27.9167 %
Kappa statistic                          0.573
Mean absolute error                      0.2382
Root mean squared error                  0.3652
Relative absolute error                 55.0176 %
Root relative squared error             78.4952 %
Total Number of Instances              480

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.659    0.219    0.702      0.659    0.680      0.443    0.778     0.689     M
                 0.690    0.142    0.671      0.690    0.681      0.544    0.877     0.754     H
                 0.858    0.076    0.801      0.858    0.829      0.765    0.948     0.858     L
Weighted Avg.    0.721    0.159    0.719      0.721    0.719      0.558    0.852     0.753

=== Confusion Matrix ===

   a   b   c   <-- classified as
 139  46  26 |   a = M
  43  98   1 |   b = H
  16   2 109 |   c = L
```

K* WEKA Output

```
=== Run information ===

Scheme:       weka.classifiers.lazy.KStar -B 20 -M a
Relation:     Other Data Rmove Topic Grade ID
Instances:    480
Attributes:   15
              gender
              NationalITy
              PlaceofBirth
              StageID
              SectionID
              Semester
              Relation
              raisedhands
              VisITedResources
              AnnouncementsView
              Discussion
              ParentAnsweringSurvey
              ParentschoolSatisfaction
              StudentAbsenceDays
              Class
Test mode:    10-fold cross-validation

=== Classifier model (full training set) ===

KStar Beta Verion (0.1b).
Copyright (c) 1995-97 by Len Trigg (trigg@cs.waikato.ac.nz).
Java port to Weka by Abdelaziz Mahoui (am14@cs.waikato.ac.nz).

KStar options : -B 20 -M a

Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         351               73.125  %
Incorrectly Classified Instances       129               26.875  %
```

```
Kappa statistic                          0.5892
Mean absolute error                      0.196
Root mean squared error                  0.3728
Relative absolute error                 45.2642 %
Root relative squared error             80.1316 %
Total Number of Instances              480

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.668    0.208    0.716      0.668    0.691      0.464    0.806     0.774     M
                 0.725    0.121    0.715      0.725    0.720      0.602    0.904     0.769     H
                 0.843    0.091    0.770      0.843    0.805      0.731    0.948     0.851     L
Weighted Avg.    0.731    0.151    0.730      0.731    0.730      0.576    0.873     0.793

=== Confusion Matrix ===

   a   b   c   <-- classified as
 141  40  30 |   a = M
  37 103   2 |   b = H
  19   1 107 |   c = L
```

Naïve Bayes WEKA Output

```
=== Run information ===

Scheme:       weka.classifiers.bayes.NaiveBayes
Relation:     Other Data Rmove Topic Grade ID
Instances:    480
Attributes:   15
              gender
              NationalITy
              PlaceofBirth
              StageID
              SectionID
              Semester
              Relation
              raisedhands
              VisITedResources
              AnnouncementsView
              Discussion
              ParentAnsweringSurvey
              ParentschoolSatisfaction
              StudentAbsenceDays
              Class
Test mode:    10-fold cross-validation

=== Classifier model (full training set) ===

Naive Bayes Classifier
```

```
                    Class
Attribute                  M      H      L
                   (0.44)  (0.3)  (0.27)
===============================================================
gender
  F                   77.0    76.0    25.0
  M                  136.0    68.0   104.0
  [total]            213.0   144.0   129.0

NationalITy
  Other                55.0    54.0    23.0
  Jordan               83.0    54.0    38.0
  KW                   76.0    37.0    69.0
  [total]            214.0   145.0   130.0

PlaceofBirth
  Other                47.0    55.0    25.0
  Jordan               90.0    54.0    35.0
  KuwaIT                77.0    36.0    70.0
  [total]            214.0   145.0   130.0

StageID
  MiddleSchool            119.0    77.0    55.0
  lowerlevel             80.0    56.0    66.0
  HighSchool              15.0    12.0     9.0
  [total]            214.0   145.0   130.0

SectionID
  A                  129.0    85.0    72.0
  B                   67.0    54.0    49.0
  C                   18.0     6.0     9.0
  [total]            214.0   145.0   130.0

Semester
  F                  107.0    63.0    78.0
  S                  106.0    81.0    51.0
  [total]            213.0   144.0   129.0

Relation
  Father               138.0    43.0   105.0
  Mum                   75.0   101.0    24.0
  [total]            213.0   144.0   129.0

raisedhands
  mean               48.9088 70.3182 16.8562
  std. dev.          26.8908 22.5134 17.1462
  weight sum             211     142     127
  precision           1.2346  1.2346  1.2346

VisITedResources
```

```
  mean                 60.5954 78.7262   18.31
  std. dev.            28.0917 19.2617 19.0818
  weight sum               211     142     127
  precision              1.125   1.125   1.125

AnnouncementsView
  mean                 40.9788 53.3868 15.6548
  std. dev.            23.9845 24.9186 15.2338
  weight sum               211     142     127
  precision             1.1264  1.1264  1.1264

Discussion
  mean                 43.7788 53.6526 30.7968
  std. dev.            26.0678 27.1279 25.6083
  weight sum               211     142     127
  precision             1.1011  1.1011  1.1011

ParentAnsweringSurvey
  Yes                   129.0   115.0    29.0
  No                     84.0    29.0   100.0
  [total]               213.0   144.0   129.0

ParentschoolSatisfaction
  Good                  132.0   119.0    44.0
  Bad                    81.0    25.0    85.0
  [total]               213.0   144.0   129.0

StudentAbsenceDays
  Under-7               141.0   139.0    12.0
  Above-7                72.0     5.0   117.0
  [total]               213.0   144.0   129.0



Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         328               68.3333 %
Incorrectly Classified Instances       152               31.6667 %
Kappa statistic                          0.5252
Mean absolute error                      0.221
Root mean squared error                  0.3889
Relative absolute error                 51.0578 %
Root relative squared error             83.6053 %
Total Number of Instances              480

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.521    0.178    0.696     0.521    0.596     0.362    0.785     0.694     M
                 0.754    0.175    0.645     0.754    0.695     0.556    0.884     0.763     H
                 0.874    0.127    0.712     0.874    0.784     0.703    0.950     0.872     L
Weighted Avg.    0.683    0.164    0.685     0.683    0.675     0.510    0.858     0.762

=== Confusion Matrix ===

   a   b   c   <-- classified as
 110  59  42 |   a = M
  32 107   3 |   b = H
  16   0 111 |   c = L
```

Random Forest WEKA Output

```
=== Run information ===

Scheme:       weka.classifiers.trees.RandomForest -P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1
Relation:     Other Data Rmove Topic Grade ID
Instances:    480
Attributes:   15
              gender
              NationalITy
              PlaceofBirth
              StageID
              SectionID
              Semester
              Relation
              raisedhands
              VisITedResources
              AnnouncementsView
              Discussion
              ParentAnsweringSurvey
              ParentschoolSatisfaction
              StudentAbsenceDays
              Class
Test mode:    10-fold cross-validation

=== Classifier model (full training set) ===

RandomForest

Bagging with 100 iterations and base learner

weka.classifiers.trees.RandomTree -K 0 -M 1.0 -V 0.001 -S 1 -do-not-check-capabilities

Time taken to build model: 0.24 seconds
```

```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         392               81.6667 %
Incorrectly Classified Instances        88               18.3333 %
Kappa statistic                          0.7156
Mean absolute error                      0.222
Root mean squared error                  0.3136
Relative absolute error                 51.2836 %
Root relative squared error             67.4027 %
Total Number of Instances              480

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.825    0.190    0.773      0.825    0.798      0.632    0.878     0.832     M
                 0.775    0.068    0.827      0.775    0.800      0.721    0.936     0.864     H
                 0.850    0.040    0.885      0.850    0.867      0.821    0.970     0.901     L
Weighted Avg.    0.817    0.114    0.819      0.817    0.817      0.708    0.919     0.860

=== Confusion Matrix ===

   a   b   c   <-- classified as
 174  23  14 |   a = M
  32 110   0 |   b = H
  19   0 108 |   c = L
```

Python Code for Plots

```
{
 "cells": [
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {},
   "outputs": [],
   "source": [
    "import pandas as pd\n",
    "import numpy as np\n",
    "import seaborn as sns \n",
    "import matplotlib as mpl\n",
    "import matplotlib.pyplot as plt\n",
    "%matplotlib inline "
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {},
   "outputs": [],
   "source": [
    "df = pd.read_csv('Schoolgrade.csv',sep=',')"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
 "metadata": {},
 "outputs": [],
 "source": [
  "df.head()"
 ]
},
{
 "cell_type": "code",
 "execution_count": null,
 "metadata": {},
 "outputs": [],
 "source": [
  "df.info()"
 ]
},
{
 "cell_type": "code",
 "execution_count": null,
 "metadata": {},
 "outputs": [],
 "source": [
  "df['gender']=df.gender.astype('category')"
 ]
},
{
 "cell_type": "code",
 "execution_count": null,
 "metadata": {},
 "outputs": [],
 "source": [
  "df['Nationality']=df.Nationality.astype('category')"
 ]
},
{
 "cell_type": "code",
 "execution_count": null,
 "metadata": {},
 "outputs": [],
 "source": [
  "df['PlaceofBirth']=df.PlaceofBirth.astype('category')"
 ]
},
{
 "cell_type": "code",
 "execution_count": null,
 "metadata": {},
 "outputs": [],
 "source": [
  "df['StageID']=df.StageID.astype('category')"
 ]
},
{
 "cell_type": "code",
 "execution_count": null,
 "metadata": {},
 "outputs": [],
 "source": [
  "df['GradeID']=df.GradeID.astype('category')"
 ]
},
{
 "cell_type": "code",
 "execution_count": null,
 "metadata": {},
 "outputs": [],
 "source": [
  "df['SectionID']=df.SectionID.astype('category')"
 ]
},
{
 "cell_type": "code",
 "execution_count": null,
 "metadata": {},
 "outputs": [],
 "source": [
  "df['Topic']=df.Topic.astype('category')"
 ]
},
{
 "cell_type": "code",
 "execution_count": null,
 "metadata": {},
 "outputs": [],
 "source": [
  "df['Semester']=df.Semester.astype('category')"
 ]
},
{
 "cell_type": "code",
 "execution_count": null,
 "metadata": {},
 "outputs": [],
 "source": [
  "df['Relation']=df.Relation.astype('category')"
 ]
},
{
 "cell_type": "code",
 "execution_count": null,
 "metadata": {},
 "outputs": [],
 "source": [
  "df['ParentAnsweringSurvey']=df.ParentAnsweringSurvey.astype('category')"
 ]
},
{
 "cell_type": "code",
 "execution_count": null,
 "metadata": {},
 "outputs": [],
 "source": [
  "df['ParentschoolSatisfaction']=df.ParentschoolSatisfaction.astype('category')"
 ]
},
{
 "cell_type": "code",
 "execution_count": null,
 "metadata": {},
 "outputs": [],
 "source": [
  "df['Class']=df.Class.astype('category')"
 ]
},
{
 "cell_type": "code",
 "execution_count": null,
 "metadata": {},
 "outputs": [],
 "source": [
  "df['StudentAbsenceDays']=df.StudentAbsenceDays.astype('category')"
 ]
},
{
 "cell_type": "code",
 "execution_count": null,
 "metadata": {},
 "outputs": [],
 "source": [
  "df.info()"
 ]
```

```
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {},
   "outputs": [],
   "source": [
    "plt.figure(figsize=(10,4))\n",
    "plt.subplot(1,2,1)\n",
    "sns.countplot('ParentAnsweringSurvey', hue='ParentschoolSatisfaction',data=df)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {},
   "outputs": [],
   "source": [
    "df.columns"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {},
   "outputs": [],
   "source": [
    "feature=['gender', 'Nationality', 'PlaceofBirth', 'StageID', 'GradeID',\n",
    "       'SectionID', 'Topic', 'Semester', 'Relation','ParentAnsweringSurvey', 'ParentschoolSatisfaction',\n",
    "       'StudentAbsenceDays','Class']"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {},
   "outputs": [],
   "source": [
    "list(enumerate(feature))"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {
    "scrolled": true
   },
   "outputs": [],
   "source": [
    "plt.figure(figsize=(30,60))\n",
    "plt.rcParams[\"axes.labelsize\"] = 15\n",
    "sns.set(font_scale = 1.5)\n",
    "for i in enumerate(feature):\n",
    "    plt.subplot(6,3,i[0]+1)\n",
    "    sns.countplot(i[1], hue='Class',data=df)\n",
    "    plt.xticks(rotation=45)\n",
    "    \n",
    "plt.savefig('test.pdf')"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {},
   "outputs": [],
   "source": [
    "plt.rcParams[\"axes.labelsize\"] = 20\n",
    "sns.set(font_scale = 1.5)\n",
    "g= sns.pairplot(df[['Raisedhands','AnnouncementsView','Discussion','Class','Visitedresources']],height=5,hue='Class')\n",
    "plt.show()\n"
   ]
  }
 ],
 "metadata": {
  "kernelspec": {
   "display_name": "Python 3",
   "language": "python",
   "name": "python3"
  },
  "language_info": {
   "codemirror_mode": {
    "name": "ipython",
    "version": 3
   },
   "file_extension": ".py",
   "mimetype": "text/x-python",
   "name": "python",
   "nbconvert_exporter": "python",
   "pygments_lexer": "ipython3",
   "version": "3.7.6"
  }
 },
 "nbformat": 4,
 "nbformat_minor": 4
}
```