CSE3999 - Technical Answers for Real World Problems (TARP)

Project Report

GENETIC ALGORITHM TO ENHANCE STUDENT PERFORMANCE ANALYSIS

By

18BCE1223	Sumit Ajmera
18BCE1031	Arpit Gupta
18BCE1087	Aadhitya Swarnesh
18BCE1226	Divvyam R Agarwal
18BCE1321	Roshni Nawaz

B.Tech. Computer Science and Engineering

Submitted to

Dr. Sajidha S

School of Computer Science and Engineering

Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

June 2021

DECLARATION

I hereby declare that the report titled "**GENETIC ALGORITHM TO ENHANCE STUDENT PERFORMANCE ANALYSIS"** submitted by me to VIT Chennai is a record of bona-fide work undertaken by me under the supervision of **Dr. Sajidha S**, School of Computer Science and Engineering, Vellore Institute of Technology, Chennai.

Signature of the Candidate

(To be (digital)signed by the student)

(Type the name of the student)

Reg. No. XXXXXXX
<Times New Roman, Font 12, Bold>

CERTIFICATE

Certified that this project report entitled "**GENETIC ALGORITHM TO ENHANCE STUDENT PERFORMANCE ANALYSIS"** is a bonafide work of **SUMIT AJMERA (18BCE1223), AADHITYA SWARNESH (18BCE1087), ARPIT GUPTA (18BCE1031), DIVVYAM R AGARWAL (18BCE1226), ROSHNI NAWAZ (18BCE1321)** and they carried out the Project work under my supervision and guidance for CSE3999 - Technical Answers for Real World Problems (TARP).

Dr. Sajidha S
SCOPE, VIT Chennai

ACKNOWLEDGEMENT

We wish to express our sincere thanks and deep sense of gratitude to our project guide, **Dr. Sajidha S**, School of Computer Science and Engineering for his/her consistent encouragement and valuable guidance offered to us throughout the course of the project work.

We are extremely grateful to **Dr. R. Jagadeesh Kannan**, Dean and **Dr. S. Geetha**, Associate Dean, School of Computer Science and Engineering (SCOPE), Vellore Institute of Technology, Chennai, for extending the facilities of the School towards our project and for their unstinting support.

We express our thanks to **Dr. Justus S,** Head of the Department, B.Tech. Computer Science and Engineering for his support throughout the course of this project.

We also take this opportunity to thank all the faculty of the School for their support and their wisdom imparted to us throughout the courses.

We thank our parents, family, and friends for bearing with us throughout the course of our project and for the opportunity they provided us in undergoing this course in such a prestigious institution.

18BCE1223 Sumit Ajmera
18BCE1031 Arpit Gupta
18BCE1087 Aadhitya Swarnesh
18BCE1226 Divvyam R Agarwal
18BCE1321 Roshni Nawaz

ABSTRACT

The saying — “Data is the new Oil” has been popular for the past decade, and this has brought about a situation where the amount of data available now is incomprehensible by humans. We rely on machines to help us comprehend this data, leading to the rise of Machine Learning. The field of Educational Data Mining is no exception, we have details of student’s behaviour and performance over the past few years. The aim here is to put this data to use, helping the students overcome their hurdles and emerge with successful careers. In this paper, we suggest an improvement over the existing Machine Learning models used in student’s performance prediction with the application of Genetic Algorithms as a Feature Selector combined with appropriate encoding schemes. We have proposed a three tier architecture with the Genetic Algorithm and One Hot encoding as the backbone. We have combined this with 7 standard Classification Algorithms (C4.5, SVC, L-GBM, etc) which have showcased an exceptional improvement in predicting the student’s final performance when Genetic Feature selection is applied.

CONTENTS

1. Introduction

1.1 **Objective**

It is always an upperhand for people who possess a general amount of knowledge in different fields. This system will take into consideration different aspects of a student's performance and personality. It will analyze the understanding and mentality of the person in various fields based on the entries. The student performance will be predicted on the basis of multiple semester marks, their involvement in different clubs and extracurricular activities. We will also try to incorporate a personalized result after analysis to motivate the students and to get an accurate model to forecast the students overall performance.

1.2 **Problem Statement**

Today's Students reshape our tomorrow, this being said it is our responsibility to guide them well until they reach a stage where they can analyse the situations and make wise decisions themselves. One important part in this process is played by the education system which prepares them to face the world. A balanced primary education is a must for every person to have a bright future. Today we have students who spend almost the first two decades of their lives just to get through one exam. On the other hand, due to the lack of proper nurturing at an early stage, students fall prey to harmful things. It becomes the responsibility of the primary education system to ensure that they excel in both having a well versed knowledge of their subjects, and psychologically by maintaining a balance between their education and the extra-curricular activities that they engage in, the hobbies they develop etc.

1.3 **Introduction**

Educational Data Mining and Analytics (EDM and EDA) are budding fields in today's Big Data era. Until now students have been reliant on only their examination marks as a measure of their performance, and thus many tend to just concentrate on their studies just before their examinations. But the increasing reliance of other fields on data analytics and machine learning has prompted us to explore new possibilities in the field of education. After the advent of computers ever since the 90's its usage is widespread, and the performance of students have been recorded on disks for a long time now. With this huge quantity of data, the possibilities are endless. For example, it can provide teachers, an general idea regarding a particular student before conducting any examinations which might help them to evaluate the student better. It might help the students to gain an understanding of how efficiently they can spend their time to shine in their subjects as well as their extracurricular activities. Web-based systems routinely collect vast quantities of data on user patterns, and data mining methods can be applied to these databases[1]. Students in classrooms tend to learn more by sharing their experiences and knowledge[2].

Genetic algorithm's rise in the field of machine learning is unprecedented. This heuristic based feature selection algorithm helps increase the accuracy of machine learning techniques and is used in many fields such as healthcare, economy, agriculture etc. There are a few researches which have associated GA with educational data mining[3].

Educational domains have a set of challenging problems for machine learning researchers since education systems offer complex information such as students information, class and schedule information, admission and registration, alumni information, etc. The motivation of this paper is to predict students performance-based on historical data using a hybrid machine learning approach.

In this paper we provide a new version of a performance prediction algorithm using genetic algorithms for feature selections in performance enhancement. They help in the process of feature selection and proper encoding of the dataset helps in fine tuning the dataset. The student performance will be predicted on the basis of past semester marks, their involvement in different clubs and extracurricular activities, etc. The main purpose of using feature selection techniques is to reduce the amount of redundancy and make maximum utilization of the subset of relevant features while maintaining high accuracy. All this is done without losing important information. We believe that genetic algorithms will increase the performance of the classical machine learning algorithm. In this paper we will also prove that proper pre-processing of the dataset can lead to better results.

In this paper, we predict the student grades using 7 different algorithms and compare them: Decision tree classifier, Bernoulli Naive Bayes Classifier, K- nearest neighbour classifier, Light GBM classifier, Random Forest classifier, Gaussian Naive Bayes classifier, Linear Support Vector classifier. The results are obtained as a 2 level grade encoding(Pass/Fail) and 5 level grade encoding(0-4). The dataset applied contains 33 different features out of which random combinations of 20 features are selected and iterated over 38 times (crossover) which helps us get refined features and helps boost the accuracy.

2. Literature Survey

There have been earlier works done in the analysis of student performances using various data mining techniques. In this pandemic stricken era, most classes are covered online. This combined with a proven fact that being consistent in online discussion is directly correlated to a student's grade in the exam[4], requires a fundamental shift in the way we approach this problem. Geographic background, psychographic characteristics also contribute to student performances, leading to students from urban areas performing better than students in rural areas[5].

Research has shown that subjective well-being and personality development directly contribute to student performance. Females have better personality development than males[6][7]. In several types of research, people have used several Machine learning algorithms for better prediction of student performances based on several factors. Hina gull et al[8] in their paper have created four classification models using a decision tree, Naïve Bayes, Logistic Regression, and Artificial Neural Network. They got the highest accuracy of 77.04% using an artificial neural network model.

Sachin bhoite et al[9] used machine learning, collaborative filtering, recommender systems, and artificial neural networks. They concluded the most effective and widely used technique for performance analysis was supervised learning. Predicting student chances of bad performance in any course in the early phase will always be beneficial and the scope of improvement can be increased. Raheela asif et al[10] have used several machine learning algorithms and data mining techniques to predict the student performance based on pre-admission marks and marks in the first, and second year. Boolean datasets produce better results. Enughwure et al[11] used logical operators like "+" and "&" for finding distinctive keywords during the search which lead to better results.

Student performance is not only dependent on marks, but on several other factors that contribute to overall performance. Ching-chieh kiu et al[12] used multilayer perceptrons, decision trees, random forest, and naïve Bayes for making a model. They have concluded that the J48 decision tree provides better results than any other algorithms. Another study has developed a student management portal for determining the performance in a particular course which leads to a better approach for improvement. They have used various machine learning techniques for performance prediction[13]. Turabieh et al[14] have shown that the feature selection algorithm reduces the non-essential parameter in the dataset and therefore improves the overall prediction using data mining techniques.

Yossy et al[15] In their paper, have used seven classification models K-Nearest Neighboring with an accuracy of 86.52%, classification and regression tests with an accuracy of 86.08%, naïve Bayes with an accuracy of 84.78%, AdaBoost with an accuracy of 88.04 %, extra tree with an accuracy of 81.30%, Bernoulli naïve Bayes with an accuracy of 79.34%, random forest E with an accuracy of 87.82%, random forest G with an accuracy of 89.78%. They concluded random forest G is the best algorithm among the lot, when predicting student performance.

3 Requirements Specification

3.1 **Hardware Requirements**

- Laptop with above 4GB RAM

- Core i5/i7
- 8th/10thgen

3.2 **Software Requirements**

- Google Colab
- Libraries: numpy, pandas, sklearn, matplotlib

4 System Design

We have identified various machine learning techniques that can be applied for this scenario. The performance analysis can be seen as a classification problem where we consider various features in an education policy and predict the performance of the student. The main challenging task would be identifying the various features and finalising their weights.We have used genetic algorithms for this purpose. We would train our model considering many features and note the accuracy each model is producing with the set of features chosen. We have employed various pre-existing solutions and compared their performance relative to our model.

- We have proposed a three step architecture in order to better optimize the solution in hand.
- The Modules of our proposed solution as shown in figure 1 are :
  1. Data Pre-Processing
  2. Feature Selection By Genetic Algorithm
  3. Classification Models

4.1 **Data Pre-Processing**

The dataset chosen here regarding a Student Performance in Examination along with details on their background and activities. We have a total of 32 parameters which we use in our models. There are many parameters which have categorical data which hinders some ML models. We thus come up with ways to “encode” them, which essentially converts them into numerical values that are handled well by all ML models universally. Here we follow 2 approaches regarding encoding. First is a label encoding which essentially maps the unique values of a parameter to unique numerical values. We use this in the case of binary categorical values which is the case when the parameters have only 2 unique values. For example the parameters — "school", "gender", "address", "family size" and a few others are encoded using this method.

In the case when a parameter does not follow a binary categorical value, we turn towards a more sophisticated **One Hot Encoding** Approach. When this is applied to a parameter, it creates a new binary parameter for every unique value. For example, the "MJob" parameter has unique values like "at_home", "services", etc. While encoding this we create new binary columns for each of these values like "MJob_at_home" which will have '1' when the "M_Job" has a value of "at_home", and '0' otherwise. The shortcoming of label encoding is that the model may assume the data to be in some order. For example, in the "M_Job" parameter if we replace "at_home" as 0, "health" as 1 and "service" as 2, the machine learning model can assume them to be correlated having an order as $0 < 1 < 2$. So, it may come to conclusions such as if the mother is a health worker, their children will perform better which becomes an anomaly. One hot encoding ensures that the ML model does not correlate a feature column to misunderstand its values and thereby prevent the degradation of the results.

We turn towards this approach only for categorical values of order greater than two. This is because One Hot Encoding consumes extra space. We felt that applying this approach for a binary system did not justify its usage of more disk space.

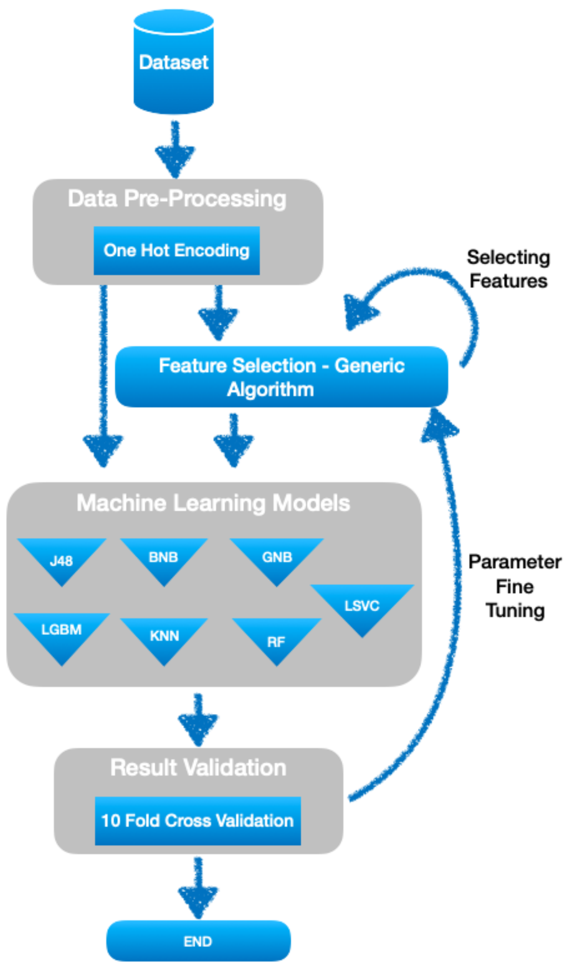


Fig 1 : Architecture Diagram

4.2. **Feature Selection By Genetic Algorithm**

Genetic algorithm is a well known feature selection technique employed for machine learning algorithms. It is inspired by Darwin's theory -- "survival of the fittest". But when we have more unique values, the chances increase for the model to misunderstand the labelled encoding as an ordinal relationship. Thus it becomes imperative to not compromise building a better model for disk space. Genetic algorithms have been applied in various domains to increase the model performances [2]. GA is a search based optimization technique that iteratively executes a five step process to get the best possible solution. The five steps involved in a Genetic algorithm cycle are - Initialization, Fitness function, Selection, Crossover and Mutation. These Methods used in genetic algorithms are explained in the below section one by one.

4.2.1 **Initial Population**

These are a set of randomly selected solutions (set of features). Each solution is called a chromosome and the pool of these chromosomes is called population as shown in Fig3. A chromosome is a collection of genes. A gene individually denotes the status

of a feature in the training set. The number of genes in a chromosome are equal to the number of features affecting the result variable. The size of the initial population is random. The more the initial population the better the results. Fig3 shows a pictorial representation of all the basic terms related to an initial population.

4.2.2 **Fitness function**

This function allocates a fitness score to each chromosome in a generation. A fitness score defines how fit the solution (chromosome) is to survive while competing with other chromosomes. In our proposed solution we have defined the accuracy performance metric achieved by the chromosome from the machine learning algorithm as the fitness score for the chromosome. It is the most important part of any genetic algorithm. It determines the rate of convergence and how the other hyperparameters to be fine tuned.

4.2.3 **Selection**

This process is analogous to Darwin’s theory of “survival of the fittest”. Only the fittest chromosomes are selected from the present generation and passed on to the next generation. The chromosomes with higher fitness scores have higher chances to be selected. The selected chromosomes are called parents and they take part in the

```
Given:
size: initial population size
num_feat : number of features
mut_rate : mutation rate
num_gen : number of generation
num_par : number of parents
```

```
// Generate 'initial_population' of length 'size'
// (Each 'chromosome' is of length 'num_feat')
curr_pop = initial_population
i = 0
while ( i < num_gen )
        // Fitness function
        for chromosome in curr_pop
                fitness_score(chromosome)   //store fitness score
        end for
        Selection(fitness_score, num_par)
        // select 'num_par' fittest members and remove the rest = Selection
        new_pop = Crossover(curr_pop)
        //cross parents to form '(size - num_par)' offsprings = Crossover
        // Mutation
        for chromosome in new_pop
                if (rand.random() < mut_rate)
                        mutate(chromosome)
                end if
        end for
        curr_pop = new_pop
end while
```

Fig 2 : Pseudo Code For Genetic Algorithm

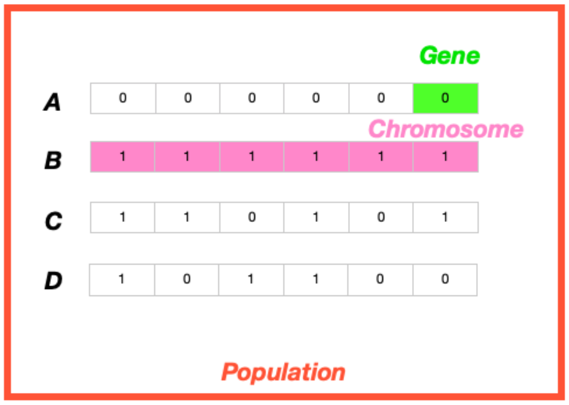


Fig 3 : Terms in Genetics

process of crossover to generate more offspring. The idea of selection is that the fittest chromosome will give us the gene that is required to produce a better solution.

4.2.4 **Crossover**

The idea of crossover is to produce a more fit solution from the fittest parents of the previous generation. In this phase the genes of two parents are exchanged between themselves to produce a more fit solution. We proceed with a predefined choice of the gene positions involved in the crossover. For example if we have a parent chromosome "A" represented by "1010100" and another parent chromosome "B" represented by " 1101111" , we have a new offspring "C" represented as "1101100" by exchanging three genes (2nd, 3rd and 4th) from the first parent with the second parent. The newer produced chromosomes along with the parent chromosomes from the previous generation together form the new generation.

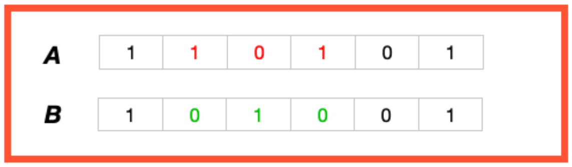


Fig 4 : Mutation

4.2.5 **Mutation**

Mutation is the process of alternating the genes in a chromosome to produce hopeful changes that can lead to a more fit chromosome. Mutation in GA is necessary as it produces diversity in the population and ensures that no premature convergence of the algorithm happens. It is largely governed by mutation rate. Mutation rate is a very small probabilistic measure that works as a threshold for a particular gene specifying whether it is to be mutated or not. The process of mutation is arbitrary. Mutation is necessary, as otherwise the population in a generation will become monotonous. Thus Mutation brings diversity among the population. Fig. 4 Shows an example of the mutation process. The mutation rate of our experimentation is mentioned in Table 1.

4.3 **Classification Models**

Classification is a technique where we categorize data into any given number of classes. We essentially identify or predict the class or category a particular data record may fall under. In Machine Learning, Classifier
Models come under Supervised learning, which means that these models get trained by looking at pre-classified data and learning patterns and trends from them. There are many classification Algorithms, here we have curated a set of 7 algorithms namely — J48 Decision Tree, Bernoulli Naive Bayes, K-Nearest Neighbor, Random Forest, Gaussian Naive Bayes, Linear Support Vector and Light GBM classifier models. Many of these models have been experimented upon in this field, and thus it seemed fruitful to improve these models.

A few of these Algorithms require parameters in order for them to function effectively. The kNN classifier is used with a value of k = 5. The Light GBM and SVC classifiers are relatively new to the field of Educational Data Mining, and have so far yielded the best results among the lot.

All the classifiers have been trained with a K-Fold Cross validation technique with the value of k = 10. This is more advanced than the traditional test-train split and is useful in the cases of a small dataset. Here we randomly rearrange the entire dataset and then split it into 'k' parts. We then use the 'k-1' parts as our training set. This ensures that every record has an equal probability of being included in the train or test set thereby reducing the chances of the model developing a bias.

During Classification, we have followed two strategies — a two class split and a five class split. The two class split results in the model predicting if the student passes or fails their final examination. On the other hand a five class split is a more detailed approach wherein the model classifies the student by matching them to one of 5 groups which can be co-related to the grades they would receive in the real world. We have demonstrated the results obtained during both these classification scenarios.

5 Implementation of System

5.1 Data Collection

We have used the public dataset which is easily available at [16]. It shows academic performance and personal details of students from two portuguese schools. The dataset consist of 33 features that are well defined in the table . The grading system ranges from 0 to 20.

The dataset has 649 samples and 33 features. The data recorded contains a student's performance in portuguese subject over a year which consists of three exams. The dataset features are mainly numeric but some features are labels which were converted to numeric features using suitable label encoding or one hot encoding. The whole features are described in Table 2. The dataset was selected as a reference to measure our accuracy score with the other works that have been done in the same field.

5.2 Experimental setup

All the experiments were carried out in the Google colaboratory environment. Python3 was the language of choice and all the models were implemented through sklearn library. There were four types of experiments that have taken place. Firstly with 2 Level Classification of grades and 5 Level Classification of grades. They further were branched into 2 sub-categories each, one with feature selection and other without feature selection. So end up with four different scenarios - 2-Level classification without feature selection, 2-Level classification with feature selection, 5-Level classification without feature selection ,and 5-Level classification with feature selection. Additional libraries used for data processing were numpy and pandas along with csv file processing tools. Matplotlib was used to plot any figures that support the result and feature analysis. The experimental setup values used in GA are shown in Table 1.

5.3 Performance Evaluation

We used the well known performance metric of accuracy for performance evaluation of classification techniques. Higher the accuracy means the better the machine learning model results. We calculated the accuracy using the confusion matrix. Accuracy is basically the ratio of total correctly predicted rows and the total number of rows in the dataset.

Since the dataset is small there are chances of the dataset being biased. To solve this problem we have used the K-Fold Validation technique for result calculation. It is a less biased technique because it ensures every row to be a part of a test or train dataset.

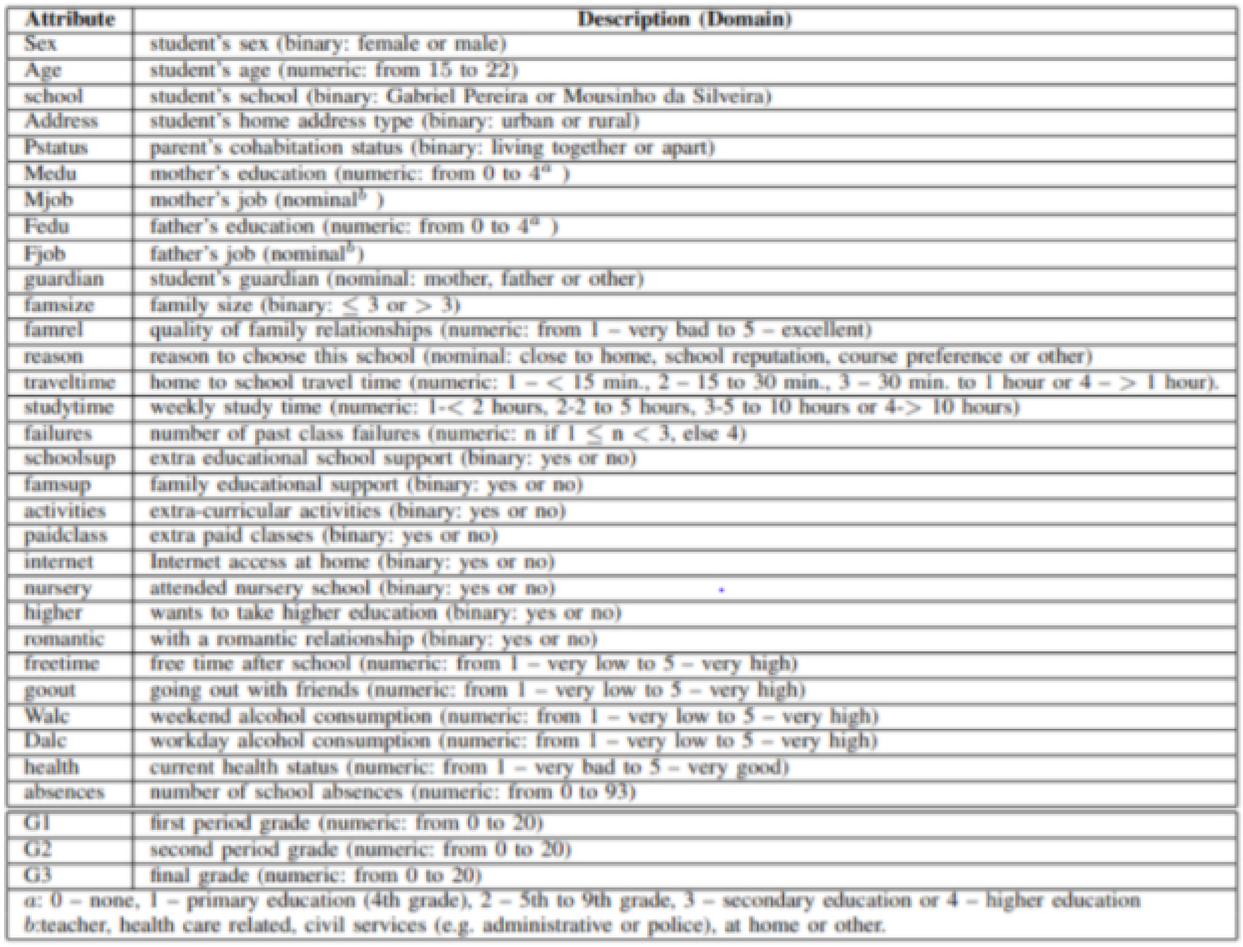
The steps involved in a K-Fold validation are as follows -

1. Split the dataset into K different folds of equal size. The choice of K is as per the requirement of data (usually ranges between 5-10), but a high value of K ensures less bias in training.
2. Train the model to fit on the K-1 folds and test the model accuracy using the remaining kth fold. Store the accuracy score.
3. Repeat this process until all the k folds serve as a test set for one time.
4. The average of all the accuracy scores determine the accuracy of the model.

For this implementation we have used the value of K to be decided by manually running the algorithms for different values of K from 5 to 10. We were getting the best results with K value set to 10. So for the whole experiment we have set K to 10 and obtained the results.

Hyperparameter	Value for experimentation
Mutation rate	0.1
Number of generations	38
Number of parents	10
Initial population size	10
Crossover	3-7 circular

Table 1 - Setting of Hyperparameters for experiments.



Attribute	Description (Domain)
Sex	student's sex (binary: female or male)
Age	student's age (numeric: from 15 to 22)
school	student's school (binary: Gabriel Pereira or Mousinho da Silveira)
Address	student's home address type (binary: urban or rural)
Pstatus	parent's cohabitation status (binary: living together or apart)
Medu	mother's education (numeric: from 0 to 4[a])
Mjob	mother's job (nominal[b])
Fedu	father's education (numeric: from 0 to 4[a])
Fjob	father's job (nominal[b])
guardian	student's guardian (nominal: mother, father or other)
famsize	family size (binary: ≤ 3 or > 3)
famrel	quality of family relationships (numeric: from 1 – very bad to 5 – excellent)
reason	reason to choose this school (nominal: close to home, school reputation, course preference or other)
traveltime	home to school travel time (numeric: 1 – < 15 min., 2 – 15 to 30 min., 3 – 30 min. to 1 hour or 4 – > 1 hour).
studytime	weekly study time (numeric: 1-< 2 hours, 2-2 to 5 hours, 3-5 to 10 hours or 4-> 10 hours)
failures	number of past class failures (numeric: n if $1 \leq n < 3$, else 4)
schoolsup	extra educational school support (binary: yes or no)
famsup	family educational support (binary: yes or no)
activities	extra-curricular activities (binary: yes or no)
paidclass	extra paid classes (binary: yes or no)
internet	Internet access at home (binary: yes or no)
nursery	attended nursery school (binary: yes or no)
higher	wants to take higher education (binary: yes or no)
romantic	with a romantic relationship (binary: yes or no)
freetime	free time after school (numeric: from 1 – very low to 5 – very high)
goout	going out with friends (numeric: from 1 – very low to 5 – very high)
Walc	weekend alcohol consumption (numeric: from 1 – very low to 5 – very high)
Dalc	workday alcohol consumption (numeric: from 1 – very low to 5 – very high)
health	current health status (numeric: from 1 – very bad to 5 – very good)
absences	number of school absences (numeric: from 0 to 93)
G1	first period grade (numeric: from 0 to 20)
G2	second period grade (numeric: from 0 to 20)
G3	final grade (numeric: from 0 to 20)
a: 0 – none, 1 – primary education (4th grade), 2 – 5th to 9th grade, 3 – secondary education or 4 – higher education *b*:teacher, health care related, civil services (e.g. administrative or police), at home or other.	

Table 2 - Dataset Features Description

6. Results and Discussion

In this paper we have made a comparative study on how usage of genetic algorithms and one hot encoding enhance the accuracy of classification algorithms. The experiment was conducted using seven of traditional classification models - K Nearest Neighbour, Gaussian Naive Bayes, Bernoulli Naive Bayes, J48 Decision Tree, Random Forest, LGBM Classifier and Linear SVC. We trained our classification algorithms with the dataset mentioned in the above section and noted the accuracy obtained after K-Fold Validation.

All the experimental results are verified by repetitive execution and complete verification by all the authors. For the result finding in table 3 we have trained a dataset with 2 level classification of grades on two different models. One with use of encoding and other without use of encoding. There is no Genetic algorithm used for this experimentation. For the rest of the experimental findings we have used dataset with encoding.

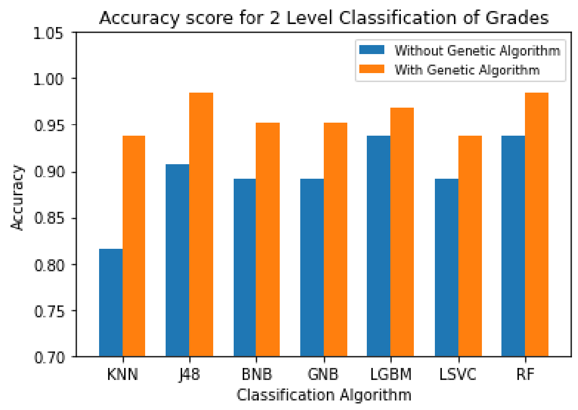


Fig 5: Accuracy scores for 2-Level Classification of grades

Figure 4 shows the bar graph for the accuracy of various machine learning techniques on 2 LevelClassification of grades in the dataset. The orange bars indicate the performance of the algorithm when trained with a genetic algorithm and the blue bars indicate the performance without a genetic algorithm.

Figure 5 shows the same bar graph with 5 level Classification of grades. In both the figures it is quite evident that the algorithms work better with features selected from genetic algorithms.

Table 3 proves the claim that one hot encoding is better than the normal label encoding for non binary non numerical values. It can be seen that KNN works best when we do not use one hot encoding which is due to the better performance of KNN on correlated datasets. Apart from KNN one hot encoding produced an increase in accuracy for at

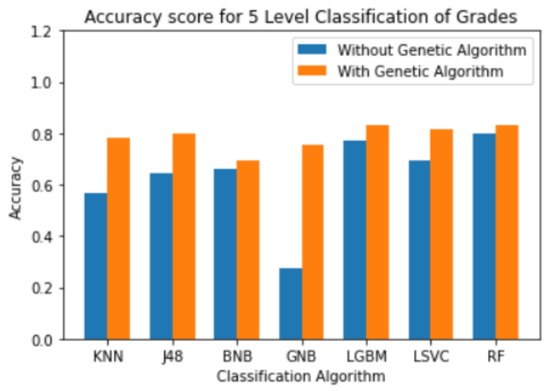
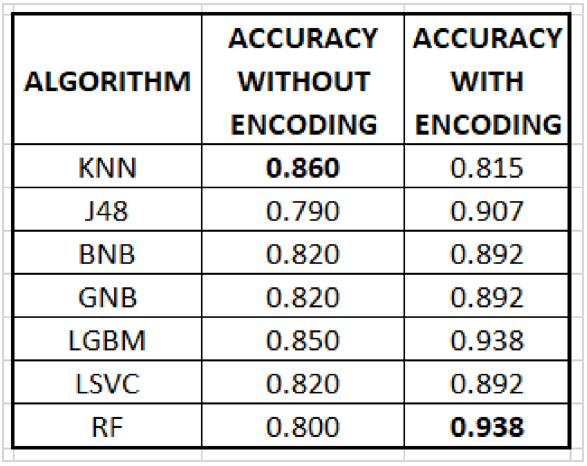


Fig 6: Accuracy scores for 5-Level Classification of grades

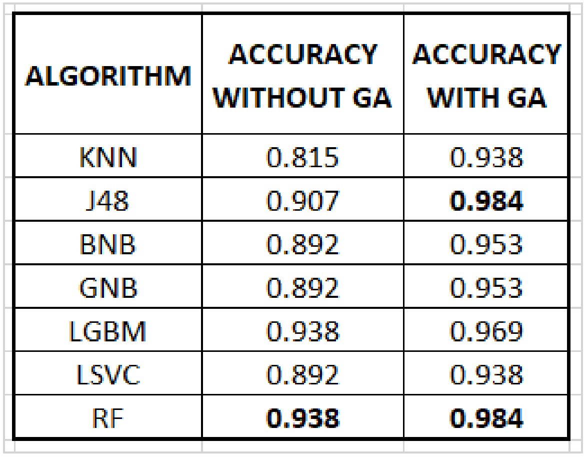
least 7% for all the classification algorithms with a spectacular increase of 13.8% for random forest.

Table 4 shows the results obtained when the dataset with grades encoded in 2 Levels (Pass or Fail) is trained with classification algorithms with GA and without feature selection from GA. From the data it is evident that there is a substantial growth in accuracy with the help of GA. Random Forest is the best data mining technique for student performance prediction. It produces an accuracy of 98.4% which is much higher than the previous ML models discussed in the related works section. Overall every model produced an increase in accuracy in the range of 3~12%.



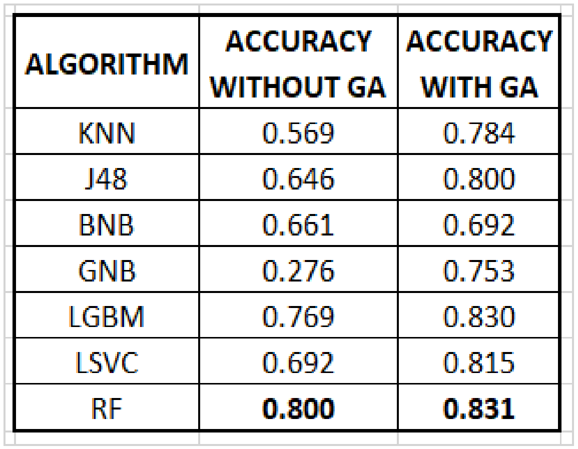
ALGORITHM	ACCURACY WITHOUT ENCODING	ACCURACY WITH ENCODING
KNN	**0.860**	0.815
J48	0.790	0.907
BNB	0.820	0.892
GNB	0.820	0.892
LGBM	0.850	0.938
LSVC	0.820	0.892
RF	0.800	**0.938**

Table 3 - Accuracy Comparison for dataset with and without encoding for 2 Level grade classification



ALGORITHM	ACCURACY WITHOUT GA	ACCURACY WITH GA
KNN	0.815	0.938
J48	0.907	**0.984**
BNB	0.892	0.953
GNB	0.892	0.953
LGBM	0.938	0.969
LSVC	0.892	0.938
RF	**0.938**	**0.984**

Table 4 - Accuracy Comparison for with and without GA for 2 Level Grade classification.

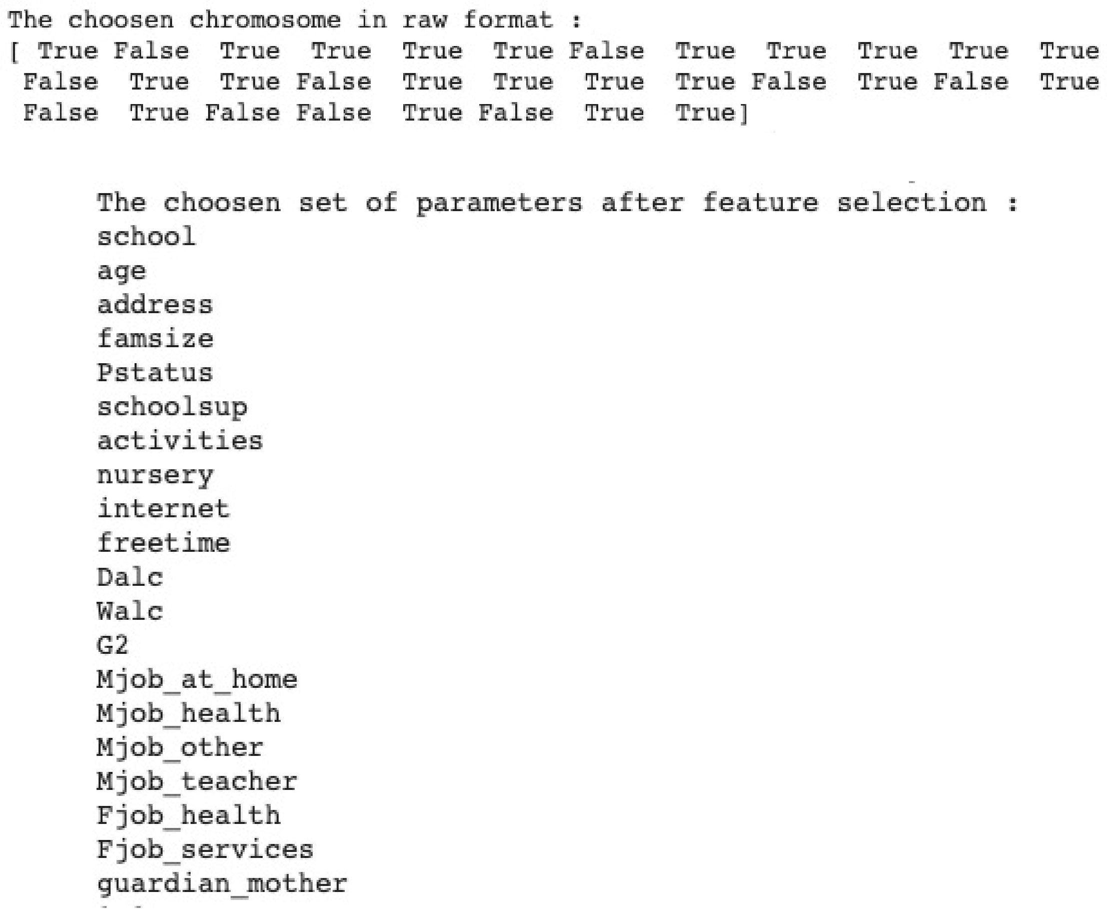


ALGORITHM	ACCURACY WITHOUT GA	ACCURACY WITH GA
KNN	0.569	0.784
J48	0.646	0.800
BNB	0.661	0.692
GNB	0.276	0.753
LGBM	0.769	0.830
LSVC	0.692	0.815
RF	**0.800**	**0.831**

Table 5 - Accuracy Comparison for with and without GA for 5 Level Grade Classification.

The above conclusion can be further proved by the result produced in Table 5. It shows the accuracy produced by various ML algorithms when we encode the grades in 5 Levels (0 - 4). We can see that while the accuracy is lower than the 2 Level encoding, it still shows an increase when we apply a genetic algorithm.

The figure 7 denotes a sample output of the features that the genetic algorithm gives as a set of trimmed features that it determines to be the best combination which when trained using the model would produce the best results. In these diagrams part (a) denotes the raw output of the genetic algorithm which is an array of boolean true/false values which denote whether a particular feature is selected or not. The part (b) gives a more intuitive representation, by displaying the actual list of parameters selected by this algorithm as the optimal set of features.



```
The choosen chromosome in raw format :
[ True False  True  True  True  True False  True  True  True  True  True
 False  True  True False  True  True  True  True False  True False  True
 False  True False False  True False  True  True]
```

```
The choosen set of parameters after feature selection :
school
age
address
famsize
Pstatus
schoolsup
activities
nursery
internet
freetime
Dalc
Walc
G2
Mjob_at_home
Mjob_health
Mjob_other
Mjob_teacher
Fjob_health
Fjob_services
guardian_mother
```

Figure 7 : (a) and (b) denoting the output features of an iteration of the Genetic Algorithm

7. Conclusion and Future Work

In this paper we have proposed a genetic algorithm along with encoding to predict the student performance for a known public dataset as our benchmark. Through the results we can conclude that the encoding technique that we proposed produces better results than normal encoding. Also we have concluded that the use of genetic algorithms for feature selection will boost the performance for the ML models. We have tested our models for various scenarios and all the results are verified around 10 times.

In the future, we can try building a generic model for Indian education policy for students and also we believe that the accuracy of the model can be further increased if we use outlier removal through clustering technique. We can also try our own ensemble clustering and compare the different results that we will get.

8. REFERENCES

[1] Bidgoli, B.M., Kashy, D., Kortemeyer, G. and Punch, W.F., 2003, November. Predicting student performance: An Application of data mining methods with the educational web-based system LON-CAPA. In *Proceedings of ASEE/IEEE frontiers in education conference*.

[2] Sukstrienwong, A., 2017. A Genetic-algorithm Approach for Balancing Learning Styles and Academic Attributes in Heterogeneous Grouping of Students. *International Journal of Emerging Technologies in Learning*, *12*(3).

[3] Katoch, S., Chauhan, S.S. and Kumar, V., 2020. A review on genetic algorithm: past, present, and future. *Multimedia Tools and Applications*, pp.1-36.

[4] S. Voghoei, N. Hashemi Tonekaboni, D. Yazdansepas and H. R. Arabnia, "University Online Courses: Correlation between Students' Participation Rate and Academic Performance," *2019 International Conference on Computational Science and Computational*

Intelligence (CSCI), 2019, pp. 772-777, doi: 10.1109/CSCI49370.2019.00147.

[5] T. N. T. Yaakub, W. R. W. Ahmad, Y. Husaini and N. Burham, "Influence Factors in Academic Performance among Electronics Engineering Student: Geographic Background, Mathematics Grade and Psycographic Characteristics," *2018 IEEE 10th International Conference on Engineering Education (ICEED)*, 2018, pp. 30-33, doi: 10.1109/ICEED.2018.8626963.

[6] J. Li and T. Wang, "The Relationship Between Subjective Well-Being and Academic Achievement of Primary and Middle School Students," 2019 10th International Conference on Information Technology in Medicine and Education (ITME), 2019, pp. 550-554, doi: 10.1109/ITME.2019.00130.

[7] Amos, P.I. and Hassan, Z., 2017. International Journal of Education, Learning and Training Quality of Teaching and its Influence on Student Satisfaction and Intention to Continue with Institution. *Int. J. Educ. Learn. Train.*, *2*, pp.1-11.

[8] H. Gull, M. Saqib, S. Z. Iqbal and S. Saeed, "Improving Learning Experience of Students by Early Prediction of Student Performance using Machine Learning," 2020 IEEE International Conference for Innovation in Technology (INOCON), 2020, pp. 1-4, doi: 10.1109/INOCON50539.2020.9298266.

[9] Bhoite, S. and Pathare, S., Students Academic Performance Analysis using Data Mining Techniques in Higher Education: A case study of college.

[10] Asif, R., Merceron, A. and Pathan, M.K., 2014. Predicting student academic performance at degree level: a case study. *International Journal of Intelligent Systems and Applications*, *7*(1), p.49.

[11] Enughwure, Akpofure & Ogbise, Mercy. (2020). Application of Machine Learning Methods to Predict Student Performance: A Systematic Literature Review. 7. 3405-3415.

[12] C. -C. Kiu, "Data Mining Analysis on Student's Academic Performance through Exploration of Student's Background and Social Activities," 2018 Fourth International Conference on Advances in Computing, Communication & Automation (ICACCA), 2018, pp. 1-5, doi: 10.1109/ICACCAF.2018.8776809.

[13]C. Li Sa, D. H. b. Abang Ibrahim, E. Dahliana Hossain and M. bin Hossin, "Student performance analysis system (SPAS)," The 5th International Conference on Information and Communication Technology for The Muslim World (ICT4M), 2014, pp. 1-6, doi: 10.1109/ICT4M.2014.7020662.

[14]H. Turabieh, "Hybrid Machine Learning Classifiers to Predict Student Performance," 2019 2nd International Conference on new Trends in Computing Sciences (ICTCS), 2019, pp. 1-6, doi: 10.1109/ICTCS.2019.8923093.

[15] E. H. Yossy, Y. Heryadi and Lukas, "Comparison of Data Mining Classification Algorithms for Student Performance," 2019 IEEE International Conference on Engineering, Technology and Education (TALE), 2019, pp. 1-4, doi: 10.1109/TALE48000.2019.9225887.

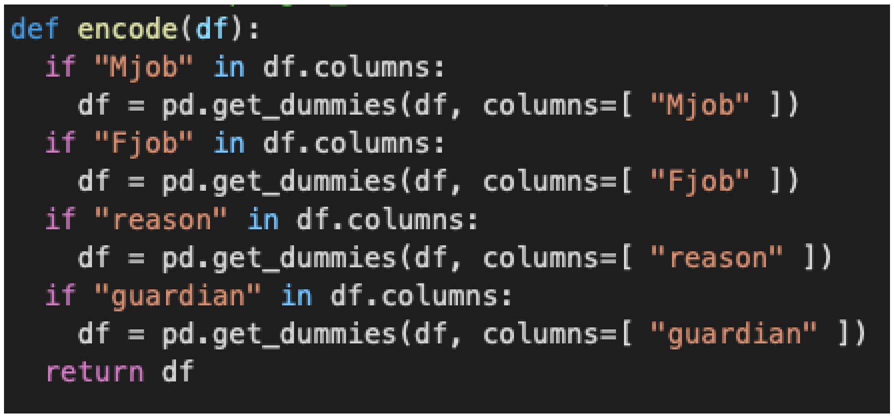
[16] https://archive.ics.uci.edu/ml/datasets/Student+Performance

[17] https://github.com/sumitajmera/Student-Performance-analysis

APPENDIX

In this section, we have added the code implementation to demonstrate the programs that power our systems and the whole architecture.
The screenshots pasted below will demonstrate the key portions of the project which lies as the backbone of the architecture.

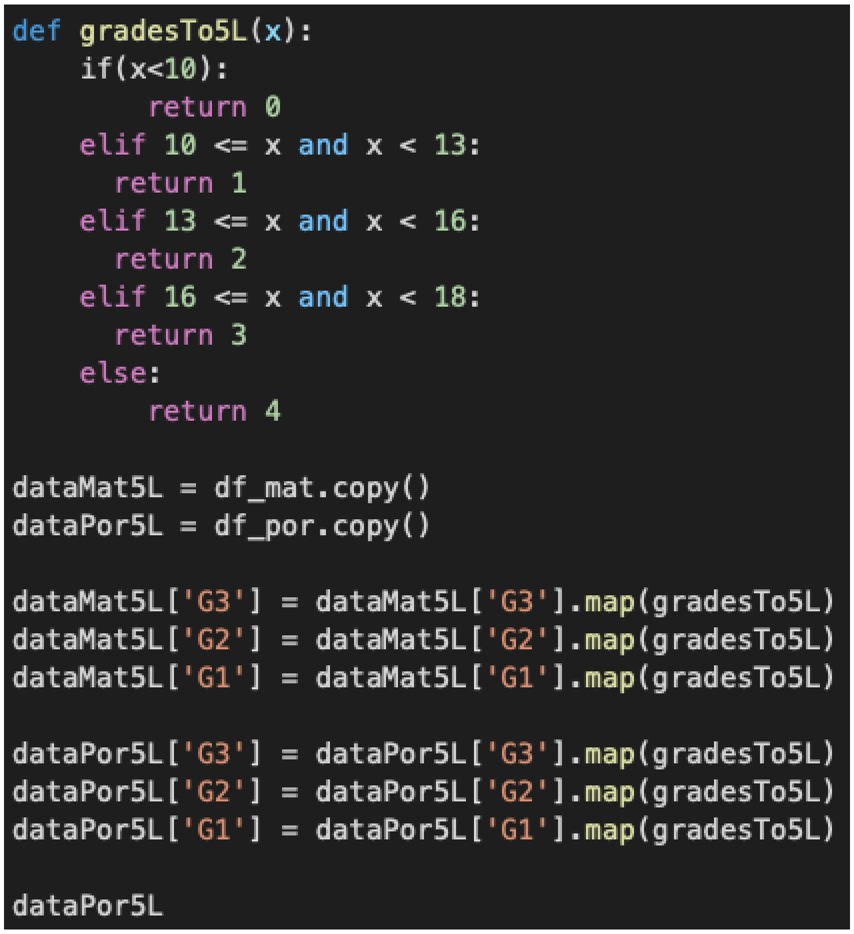
This diagram below shows the encoding scheme -- One Hot Encoding which is being done on the fields of job, guardian status, and a few other parameters.



```
def encode(df):
  if "Mjob" in df.columns:
    df = pd.get_dummies(df, columns=[ "Mjob" ])
  if "Fjob" in df.columns:
    df = pd.get_dummies(df, columns=[ "Fjob" ])
  if "reason" in df.columns:
    df = pd.get_dummies(df, columns=[ "reason" ])
  if "guardian" in df.columns:
    df = pd.get_dummies(df, columns=[ "guardian" ])
  return df
```

Figure 8 : Encoding scheme

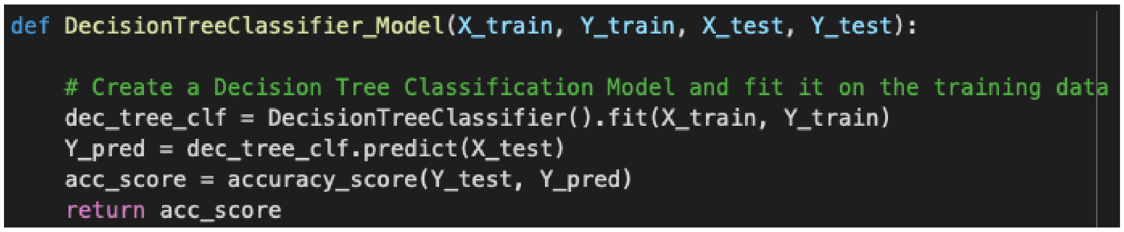
The following code is for the transforming of the grades into a 5 level scheme



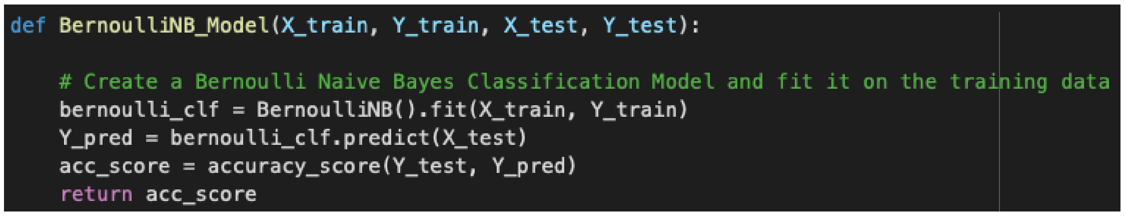
```python
def gradesTo5L(x):
    if(x<10):
        return 0
    elif 10 <= x and x < 13:
      return 1
    elif 13 <= x and x < 16:
      return 2
    elif 16 <= x and x < 18:
      return 3
    else:
        return 4

dataMat5L = df_mat.copy()
dataPor5L = df_por.copy()

dataMat5L['G3'] = dataMat5L['G3'].map(gradesTo5L)
dataMat5L['G2'] = dataMat5L['G2'].map(gradesTo5L)
dataMat5L['G1'] = dataMat5L['G1'].map(gradesTo5L)

dataPor5L['G3'] = dataPor5L['G3'].map(gradesTo5L)
dataPor5L['G2'] = dataPor5L['G2'].map(gradesTo5L)
dataPor5L['G1'] = dataPor5L['G1'].map(gradesTo5L)

dataPor5L
```

Figure 9 : 5 Level encoding scheme

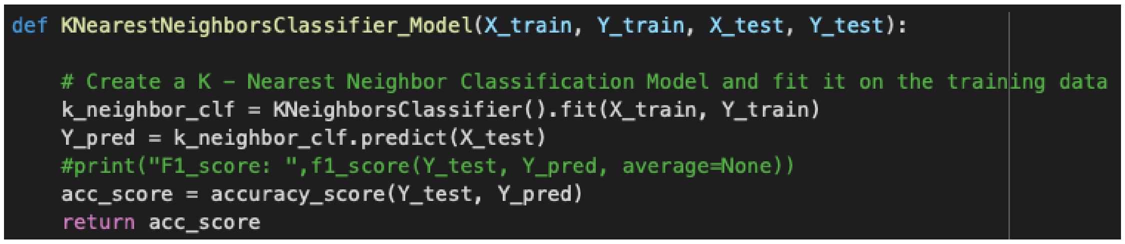
The following code denotes the ML algorithms that power the basic ML module of our architecture :



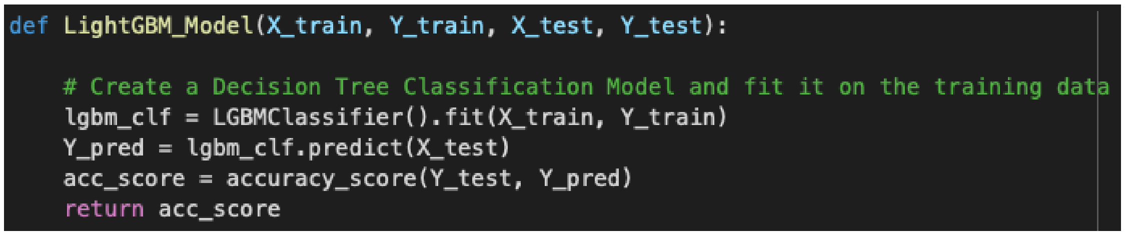
```python
def DecisionTreeClassifier_Model(X_train, Y_train, X_test, Y_test):

    # Create a Decision Tree Classification Model and fit it on the training data
    dec_tree_clf = DecisionTreeClassifier().fit(X_train, Y_train)
    Y_pred = dec_tree_clf.predict(X_test)
    acc_score = accuracy_score(Y_test, Y_pred)
    return acc_score
```



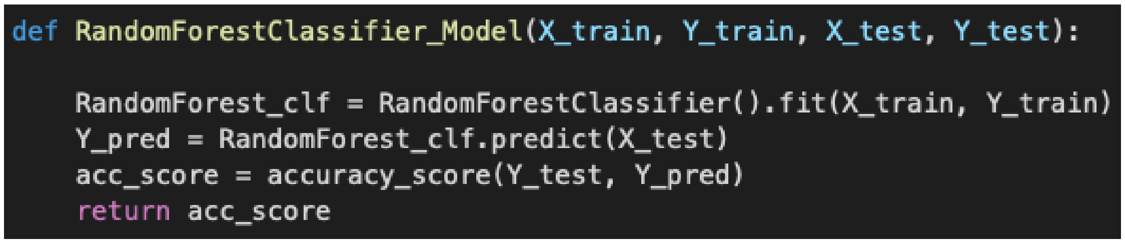
```python
def BernoulliNB_Model(X_train, Y_train, X_test, Y_test):

    # Create a Bernoulli Naive Bayes Classification Model and fit it on the training data
    bernoulli_clf = BernoulliNB().fit(X_train, Y_train)
    Y_pred = bernoulli_clf.predict(X_test)
    acc_score = accuracy_score(Y_test, Y_pred)
    return acc_score
```



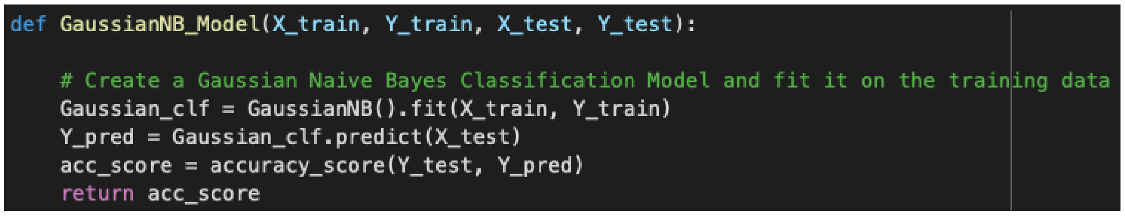
```python
def KNearestNeighborsClassifier_Model(X_train, Y_train, X_test, Y_test):

    # Create a K - Nearest Neighbor Classification Model and fit it on the training data
    k_neighbor_clf = KNeighborsClassifier().fit(X_train, Y_train)
    Y_pred = k_neighbor_clf.predict(X_test)
    #print("F1_score: ",f1_score(Y_test, Y_pred, average=None))
    acc_score = accuracy_score(Y_test, Y_pred)
    return acc_score
```



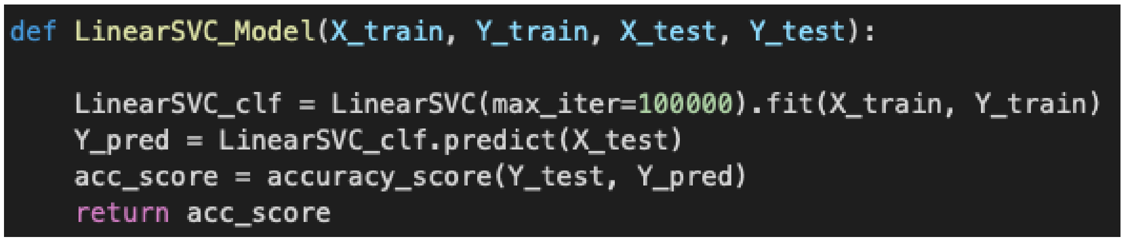
```python
def LightGBM_Model(X_train, Y_train, X_test, Y_test):

    # Create a Decision Tree Classification Model and fit it on the training data
    lgbm_clf = LGBMClassifier().fit(X_train, Y_train)
    Y_pred = lgbm_clf.predict(X_test)
    acc_score = accuracy_score(Y_test, Y_pred)
    return acc_score
```



```python
def RandomForestClassifier_Model(X_train, Y_train, X_test, Y_test):

    RandomForest_clf = RandomForestClassifier().fit(X_train, Y_train)
    Y_pred = RandomForest_clf.predict(X_test)
    acc_score = accuracy_score(Y_test, Y_pred)
    return acc_score
```

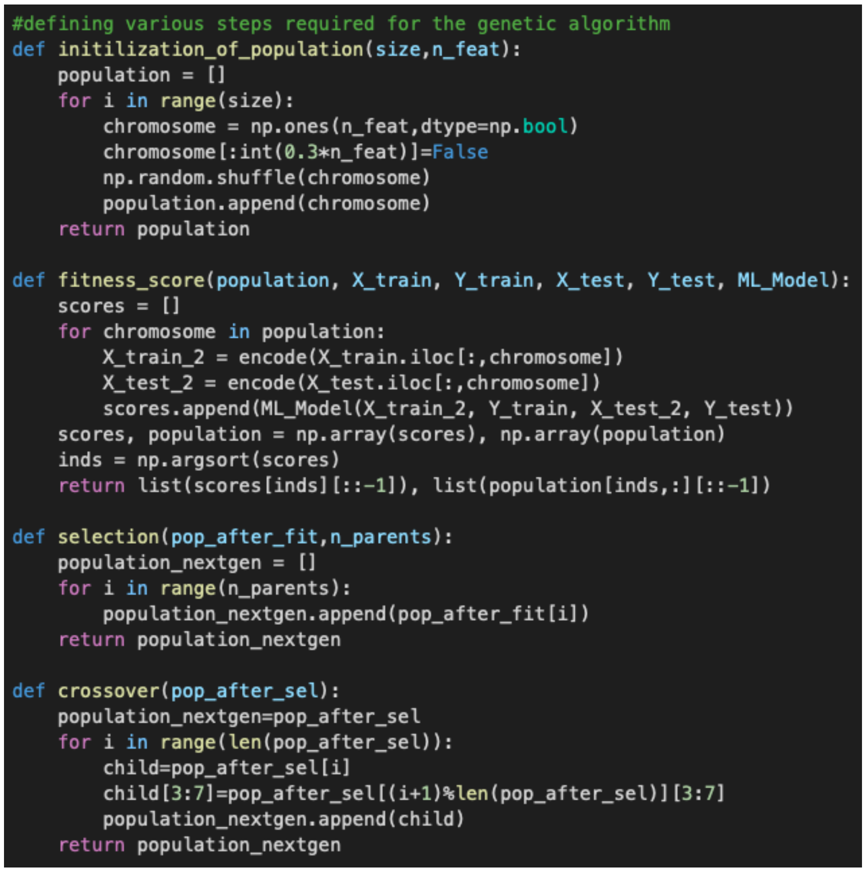


```python
def GaussianNB_Model(X_train, Y_train, X_test, Y_test):

    # Create a Gaussian Naive Bayes Classification Model and fit it on the training data
    Gaussian_clf = GaussianNB().fit(X_train, Y_train)
    Y_pred = Gaussian_clf.predict(X_test)
    acc_score = accuracy_score(Y_test, Y_pred)
    return acc_score
```

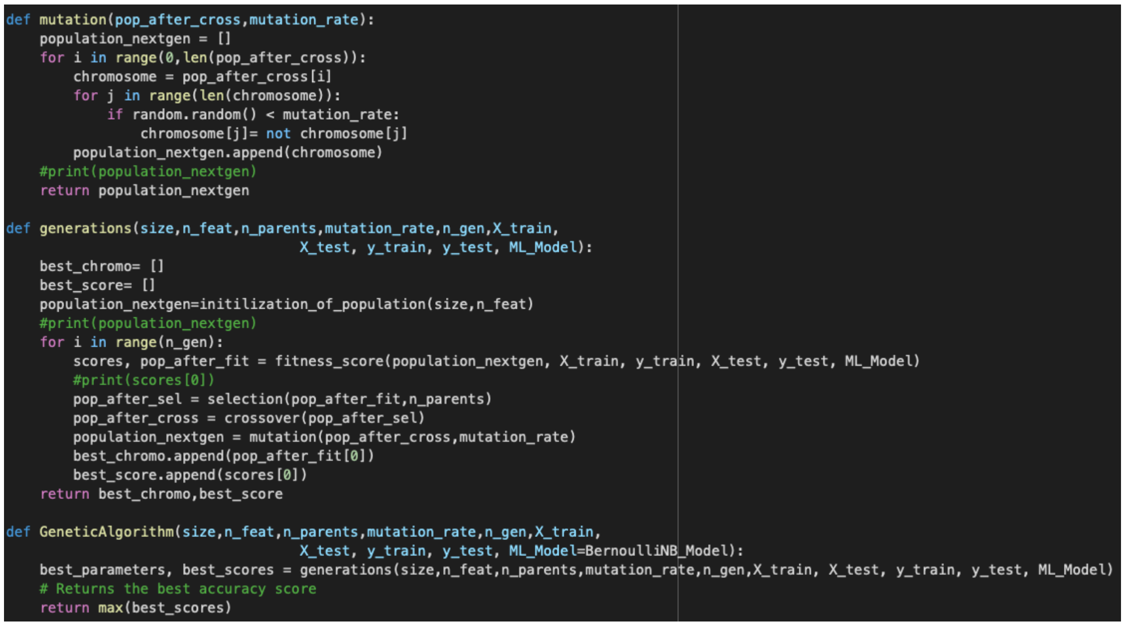


```python
def LinearSVC_Model(X_train, Y_train, X_test, Y_test):

    LinearSVC_clf = LinearSVC(max_iter=100000).fit(X_train, Y_train)
    Y_pred = LinearSVC_clf.predict(X_test)
    acc_score = accuracy_score(Y_test, Y_pred)
    return acc_score
```

Figure 10 : ML Algorithms

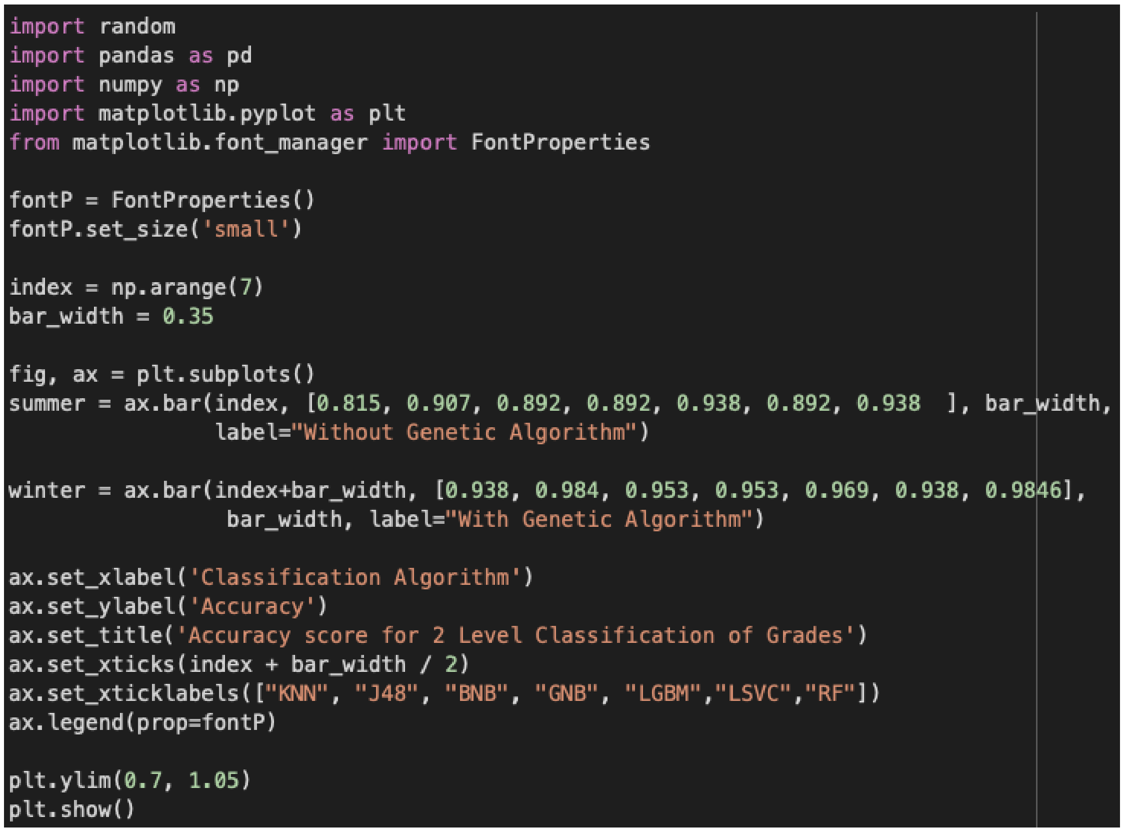


```python
#defining various steps required for the genetic algorithm
def initilization_of_population(size,n_feat):
    population = []
    for i in range(size):
        chromosome = np.ones(n_feat,dtype=np.bool)
        chromosome[:int(0.3*n_feat)]=False
        np.random.shuffle(chromosome)
        population.append(chromosome)
    return population

def fitness_score(population, X_train, Y_train, X_test, Y_test, ML_Model):
    scores = []
    for chromosome in population:
        X_train_2 = encode(X_train.iloc[:,chromosome])
        X_test_2 = encode(X_test.iloc[:,chromosome])
        scores.append(ML_Model(X_train_2, Y_train, X_test_2, Y_test))
    scores, population = np.array(scores), np.array(population)
    inds = np.argsort(scores)
    return list(scores[inds][::-1]), list(population[inds,:][::-1])

def selection(pop_after_fit,n_parents):
    population_nextgen = []
    for i in range(n_parents):
        population_nextgen.append(pop_after_fit[i])
    return population_nextgen

def crossover(pop_after_sel):
    population_nextgen=pop_after_sel
    for i in range(len(pop_after_sel)):
        child=pop_after_sel[i]
        child[3:7]=pop_after_sel[(i+1)%len(pop_after_sel)][3:7]
        population_nextgen.append(child)
    return population_nextgen
```



```python
def mutation(pop_after_cross,mutation_rate):
    population_nextgen = []
    for i in range(0,len(pop_after_cross)):
        chromosome = pop_after_cross[i]
        for j in range(len(chromosome)):
            if random.random() < mutation_rate:
                chromosome[j]= not chromosome[j]
        population_nextgen.append(chromosome)
    #print(population_nextgen)
    return population_nextgen

def generations(size,n_feat,n_parents,mutation_rate,n_gen,X_train,
                                   X_test, y_train, y_test, ML_Model):
    best_chromo= []
    best_score= []
    population_nextgen=initilization_of_population(size,n_feat)
    #print(population_nextgen)
    for i in range(n_gen):
        scores, pop_after_fit = fitness_score(population_nextgen, X_train, y_train, X_test, y_test, ML_Model)
        #print(scores[0])
        pop_after_sel = selection(pop_after_fit,n_parents)
        pop_after_cross = crossover(pop_after_sel)
        population_nextgen = mutation(pop_after_cross,mutation_rate)
        best_chromo.append(pop_after_fit[0])
        best_score.append(scores[0])
    return best_chromo,best_score

def GeneticAlgorithm(size,n_feat,n_parents,mutation_rate,n_gen,X_train,
                                   X_test, y_train, y_test, ML_Model=BernoulliNB_Model):
    best_parameters, best_scores = generations(size,n_feat,n_parents,mutation_rate,n_gen,X_train, X_test, y_train, y_test, ML_Model)
    # Returns the best accuracy score
    return max(best_scores)
```

Figure 11 : Genetic Algorithms



```
import random
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.font_manager import FontProperties

fontP = FontProperties()
fontP.set_size('small')

index = np.arange(7)
bar_width = 0.35

fig, ax = plt.subplots()
summer = ax.bar(index, [0.815, 0.907, 0.892, 0.892, 0.938, 0.892, 0.938  ], bar_width,
                label="Without Genetic Algorithm")

winter = ax.bar(index+bar_width, [0.938, 0.984, 0.953, 0.953, 0.969, 0.938, 0.9846],
                 bar_width, label="With Genetic Algorithm")

ax.set_xlabel('Classification Algorithm')
ax.set_ylabel('Accuracy')
ax.set_title('Accuracy score for 2 Level Classification of Grades')
ax.set_xticks(index + bar_width / 2)
ax.set_xticklabels(["KNN", "J48", "BNB", "GNB", "LGBM","LSVC","RF"])
ax.legend(prop=fontP)

plt.ylim(0.7, 1.05)
plt.show()
```

Figure 12 : Result Comparisons