Predicting Undergraduate Student Dropout Using Artificial Intelligence, Big Data and Machine Learning.

by

Vikas Kunchala

(Under the Direction of Khaled Rasheed)

Abstract

The aim of this research work is predicting undergraduate student dropout in a public post-secondary education institution in the Southeast United States. The main sources of data are college database storage and National Student Clearinghouse. Datasets DS-57, DS-11 and DS-101 are created from those sources. All datasets are trained using suitable classification machine learning models. Agile practices are followed to perform experiments. From the results, it is observed that important features predictive of dropouts are related to academic performance and financial aid. Models are evaluated on percent accuracy and F-measure. Random Forest performed with 0.86 F-measure and 87.04 percent classification accuracy. Further training with ensemble machine learning techniques improved F-measure to 0.903 and classification accuracy to 90.8 percent.

Index words: Machine Learning, Naïve-Bayes, Random Forest, Logistic Regression, Bayes Net, Ensemble Machine Learning, Artificial Intelligence, Dropout, Student, Prediction, Classification, AutoML, AutoGluon.

Predicting Undergraduate Student Dropout Using Artificial Intelligence, Big Data and Machine Learning.

by

Vikas Kunchala

B.Tech., Jawaharlal Nehru Technological University, India, 2009

A Thesis Submitted to the Graduate Faculty of the
University of Georgia in Partial Fulfillment of the Requirements for the Degree.

Master of Science

Athens, Georgia

2021

Predicting Undergraduate Student Dropout Using Artificial Intelligence, Big Data and Machine Learning.

by

Vikas Kunchala

Major Professor:	Khaled Rasheed
Committee:	Tianming Liu
	Frederick Maier

Electronic Version Approved:

Ron Walcott
Dean of the Graduate School
The University of Georgia
August 2021

Dedication

To Mom and Dad, who gave endless love, support, and encouragement to go on every adventure, especially this one.

Acknowledgments

Special thanks to Major professor Khaled Rasheed for helping through issues, supporting with resources, and navigating out of errors, David Tanner for allowing me to be a part of the team, James Byars for working diligently in gathering and processing data, as well as contributing to the overall work. I would also like to thank my committee, Frederick Maier and Tianming Liu. They have advised and guided me throughout the masters of science program. I would like to thank all folks of Athens, Carl Vinson Institute of Government, Institute of Artificial Intelligence, and The University of Georgia, for their great help during the uniquely hard times of COVID-19. Finally, I would like to thank all my friends and family for their incredible support.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

While stories of Steve Jobs, Mark Zuckerberg, or Larry Ellison dropping out of college are inspiring, the United States has the highest dropout rate of any developed country for kids who start a higher-education program. And that has a huge impact on their ability to build a career and earn a good living.(Christenson et al., 2000)

Students dropping out of college, high school is a worldwide problem. It impacts people's lives by limiting their potential to get more opportunities. It creates a shortage of skilled employees for businesses seeking to expand. Policymakers, parents, and the academic community are seriously concerned about trying to mitigate this problem for many years. Research has been studied in this regard in various countries dating back to 40 years. Every year 30% of students in the US do not return to finish their second year. A calculated cost of 9 billion is spent on educating these students. Uncalculated costs for society are estimated in billions of dollars in lost revenues, welfare programs, unemployment programs, underemployment, and crime prevention and prosecution (Christenson et al., 2000).

In Figure: 1.1 (Hanson, 2021) The chances of dropping out of College is less in the age group of 19 or younger. But dropout is high after age 20. Students once leave college due to personal reasons, or financial reasons, or health reasons often never return to complete education. Much of research work on student dropout is theoretical .

Most of the dropout studies are restricted to institutions or classes. This makes it difficult to have an empirical understanding of what causes student dropout. To address this challenge German Centre for Higher Education Research and Science Studies (DZHW) conducted studies (Heublein, 2014) from 2007 to 2008. 87 German institutions are surveyed across the country. 4500 deregistered(dropout students) took part in the survey. The study gives a national picture of dropout patterns. Some of the key findings from this study are seen in Figure: 1.2. With the rise in information technology and the internet, there is a boom in e-learning, MOOC, and online courses. This boom has opened data analytics opportunities to study student learning patterns across the globe. Major e-commerce retailers like Amazon, Walmart, and eBay have already started using big data to learn customer behavior, so they can increase cart-to-purchase conversion rates. On the other hand, companies like Facebook, Google have also relied on a similar data-driven approach to learn user behavior to increase engagement metrics that would result in increased ad

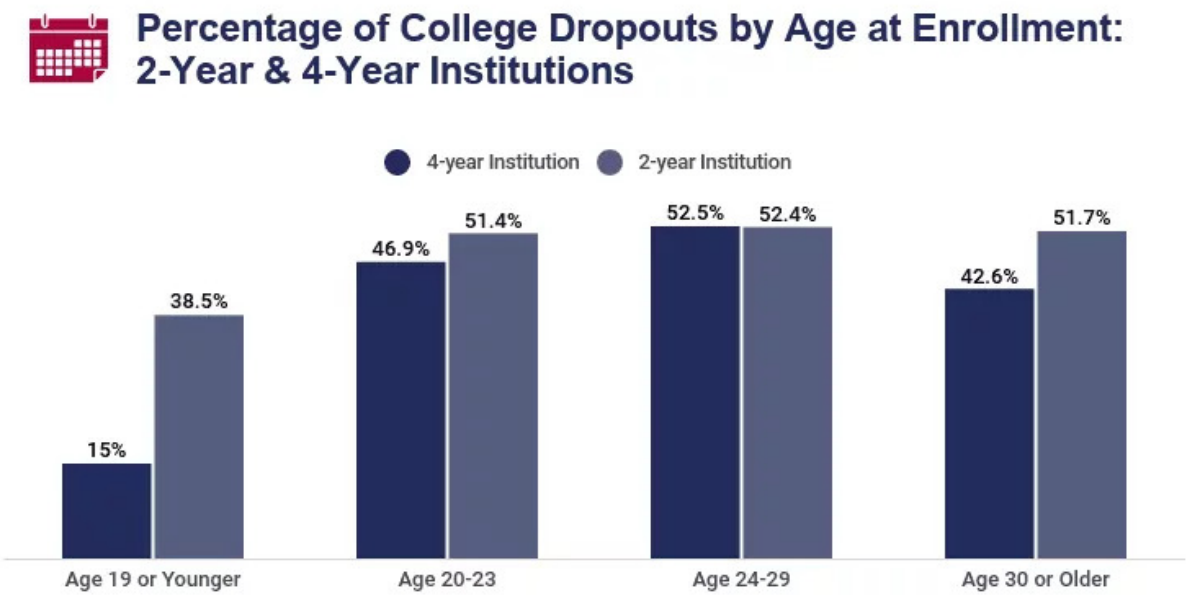


Figure 1.1: Percentage of Student dropout in 4- year and 2-year institutions across age groups.(Source:Hanson, 2021)

revenue. All these trends have converged to drive innovation in AI/ML space resulting in technologies like Keras, TensorFlow, cloud, and PyTorch. The boom in MOOC big-data and open-source ML software from big-tech has led to multiple studies in student completion rates of MOOC online courses.S Whitehill et al., 2017 is one such example. In this paper, the researchers use the HarvardX MOOC platform. Over 80% of students approximately still enroll in college and take classes in a physical classroom setting. At the writing of this thesis, due to COVID-19 classes have shifted online. With students taking classes in ZOOM video conferences. However, the focus of current research is purely on data collected on students who pursue education in a real class setting in a real college and classroom.

1.1 Literature Review

Research work in this regard is rare to find. After exhaustive search research resembling similar classroom setting environment are listed below:

1. Predicting University Students' Academic Success and Major Using Random Forests (Beaulac & Rosenthal, 2019)

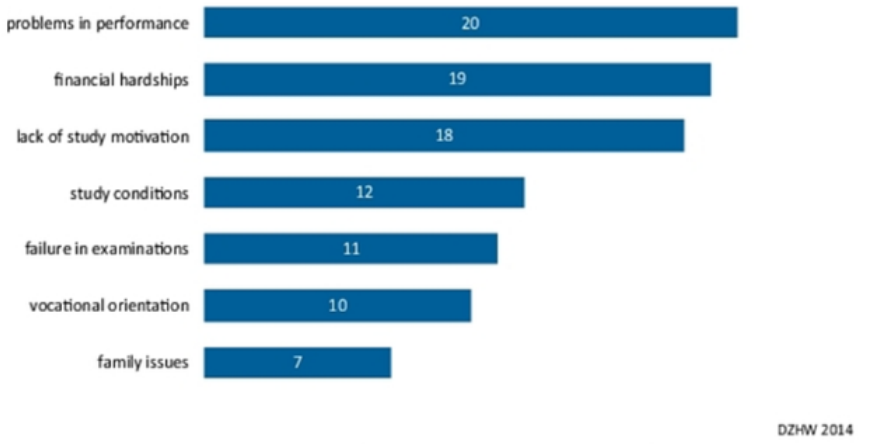


Figure 1.2: Most crucial motives of student dropout in DZHW studies.(percentages) (Source:Heublein, 2021)

2. Predicting Student Dropout in Higher Education (Aulck et al., 2017)

3. Predicting student dropout: A machine learning approach (Kemper et al., 2020)

In the above three papers, the student dropout problem is studied in an actual class setting. In the first paper, data is used from the University of Toronto, in second paper, data is used from the University of Washington, and in the third paper, the data is used from the Karlsruhe Institute of Technology. While the third publication is an interesting one, it does not match with the current study as education in Germany is free. In current work, apart from academic indicators like GPA; financial indicators like grant money, financial aid money are also taken into consideration while training the models. Of all the publications out on the internet, first two papers(Toronto and Washington) come close to current research in environment setting and data collection. Percentage Classification accuracy from these publications is from 53% to 78%. For the current study, the data used is heterogeneous. It is collected from several sources. Data collected is from a major public college generated over 12 years (2006-2018).

1.2 Background motivation of Research

In 2018 a public post-secondary education institution in the southeastern United States of America approached the Carl Vinson Institute of Government with the student dropout problem. Due to the data share agreement, the education Institution's name is kept confidential. In this research, it will be referred to as a public postsecondary education institution in the Southeast United States or just simply a southeast college. The student dropout problem is framed as a supervised classification problem.

1.3 Structuring of Thesis

The thesis paper is organized as follows: chapter 2 describes data, software, and hardware used. In chapter 3 methods, environments, machine learning theory are discussed. In chapter 4, data sets, Features, Features selection methods used and Feature importance of select models are shown. Chapter 5 presents a detailed discussion on experiments. AutoGluon results are also discussed. Finally, chapter 6 offers a conclusion and future work.

Chapter 2

Data Collection, Software, And Hardware

2.1 Data Collection

Our main goal is to organize data into a useful structure, run machine learning algorithms on them to uncover hidden patterns, and finally build a model that would make a prediction. We used college data obtained from Banner tables for the years 2006 to 2018. An important decision is taken from the beginning of this study to de-identify data and apply innovative ways to gather data that give a clear picture of a student's academic outcome.

Initially, the dataset is designed to be consumed by ML models that predict in a supervised multi-label classification problem. In this dataset, there were four labels- graduated, continuing, dropout, and transferred. A major part of student data is extracted from an education enterprise resource planning (ERP) system called Ellucian's Banner. This system has labels related to continuing or graduated. Labels of students who left college are acquired from The National Student Clearinghouse. It is the source for degree verification and enrollment verification and student educational outcomes research. Driving distance and driving data are created using contact addresses and college addresses. This is achieved by using ESRI's ArcGIS Pro("ArcGIS Pro", 2020) which geocoded (e.g., Latitude and longitude) the current address of students. An API to Google Maps is then utilized to calculate the drive distance and drive duration.

To summarize, data is collected first from college, second from the national database to capture student outcomes, after the person left college. Finally, location information is considered to study if there is any correlation to student outcome.

Data from different sources is piped to the local My SQL server. SQL queries are run to extract custom fields and further saved as CSV files. These CSV files are further processed, analyzed using python packages such as PANDAS, NUMPY. WEKA is also used to preprocess and create different datasets. The overall data used during this study was 4266543 data points, consisting of 42243 rows and 101 columns. The dataset is imbalanced.

Table 2.1: Dataset Rows, Columns And Datapoints

Dataset	Rows	Columns	Datapoints (R*C)
DS-101	42243	101	4266543
DS-57	29203	57	1664571
DS-11	29203	11	321233

Table 2.2: Dataset Class labels

Dataset	Class Labels	Labels
DS-101	4	Dropout, Transferred, Continuing, Graduated
DS-57	2	Dropout, Non-Dropout
DS-11	2	Dropout, Non-Dropout

Few techniques are explored to balance the four label datasets. One such technique is SMOTE. Even after treating the data with imbalance no improvement in performance is observed. Hence, we explored additional options. During experimentation and feedback, the idea to create a dataset with two labels is imagined. This led to the exploration of ideas on new datasets. This time the dropout problem is framed as a binary classification problem. From the perspective of college, predicting the dropout student label with high accuracy is more important than making predictions on graduated, transferred, or continuing labels. With this newly revised vision and further preprocessing, datasets with two labels, dropout, and non-dropout are created. It was done by basically, renaming three labels graduated, transfered and continuing in to one label called Non-Dropout. One of the first datasets had 57 columns and 29203 rows, with 1664571 cells. Imbalance is also addressed with this new two-label configuration. The results on this dataset were in line with published study research and pointing towards the random forest as an ideal model.

2.2 Software

Given the heterogeneous landscape of IT systems and limitations of legacy systems; a collection of tools and software are used to find high-performing models while operating within the constraints of available data, computational power, and regulations. The exhaustive list of software used is as follows-

1. WEKA
2. ESRI's ArcGIS Pro
3. MySQL
4. Python packages
5. Conda framework
6. WinSCP

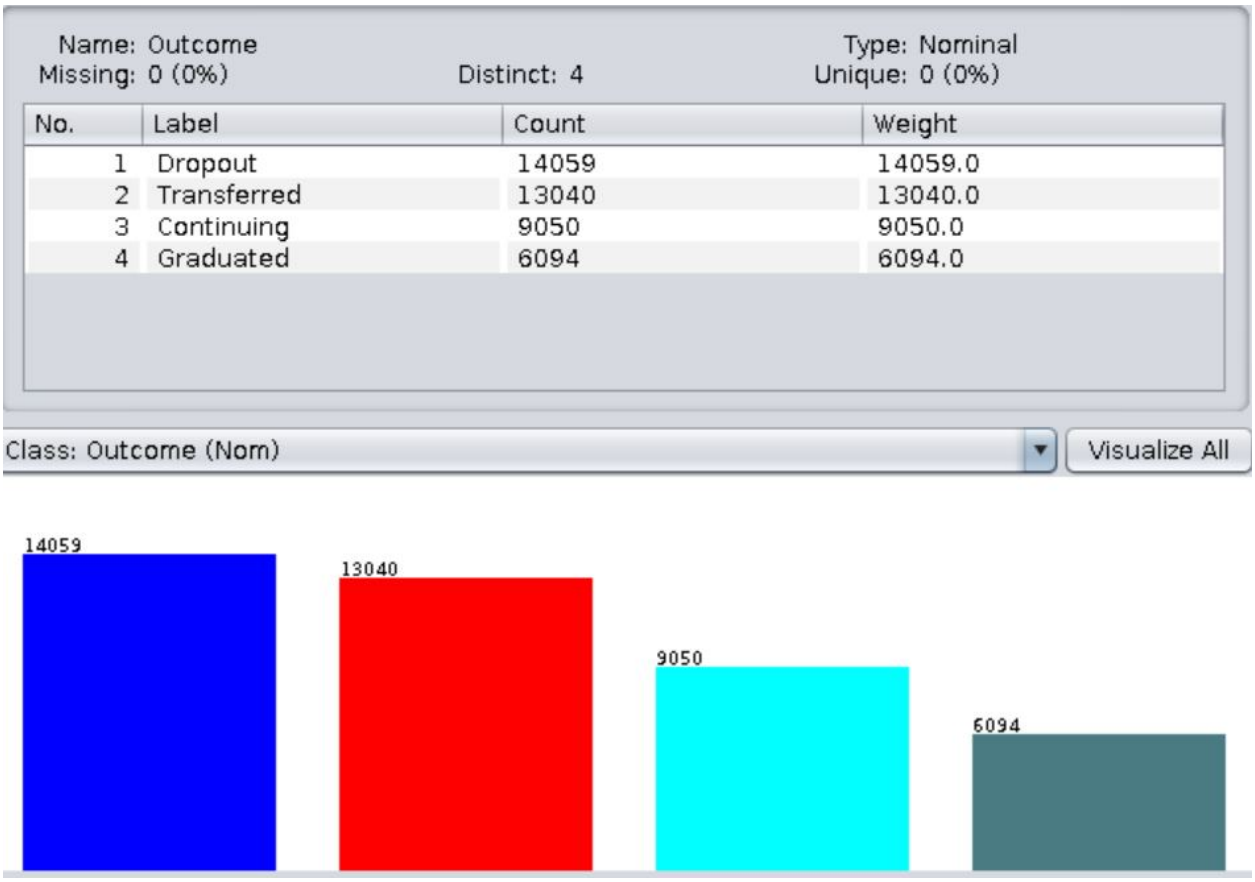


Figure 2.1: Class Label Distribution in DS-101

7. Xming
8. Putty
9. AutoGluon
10. Jupyter Notebooks.

2.3 Hardware

Georgia Advanced Computing Resource Center provided High Performance Computing(HPC) clusters and High-Performance Storage. The HPC cluster is called the Sapelo2 Cluster. It runs CentOS Linux distribution. The cluster currently has

1. More than 16,000 cores.
2. general purpose compute nodes (128GB, 192GB RAM).
3. high-memory nodes (256GB, 512GB & 1TB RAM).
4. CPU/GPU hybrid nodes(NVidia K20X, K40 & P100 GPUs).
5. High-Performance Storage- DDN SFA14KX.
6. Lustre appliance (1.50PB).
7. ZFS storage chains (100TB).

Chapter 3

Methods

In this section experimental setup is discussed. The environments used for research. A brief background on selected machine learning algorithms, and few ensemble machine learning models.

3.1 Environments

This research uses two environments. WEKA(Frank et al., 2005) and Anaconda distribution ("Anaconda Software Distribution", 2020). WEKA is a GUI based data mining tool. Anaconda is a scientific computing distribution; it has a rich set of machine libraries and wide support of programming languages.

3.1.1 Anaconda Environment

Anaconda is maintained by Anaconda Inc. It is a distribution of Python and R languages. It is free for individual use. A popular distribution for scientific computing. Some of its benefits include simplified package management and deployment. Package versions are managed by the package management system Conda. Conda was critical while setting up virtual environments and installing dependent libraries in HPC without admin access. To run experiments, virtual python environments are created in the Sapelo2 cluster. All relevant and latest packages are installed. These are then loaded in Jupyter notebooks to be used in code. The Anaconda distribution is supported on Windows, Linux, and macOS. The default installation comes with the following applications:

- JupyterLab
- Jupyter Notebook
- QtConsole
- Spyder
- Glue

- Orange
- RStudio
- Visual Studio Code

We used Jupyter notebooks extensively to run all experiments, for which code is written in python v3. Jupyter Notebooks is a web-based interactive computational environment. It can be run with different kernels, but by default it supports python. At the time of this research, it supports 49 language kernels. Jupyter notebooks have a ".ipynb" extension. It has become popular in the industry as well. A derivative of Jupyter notebooks is present across major cloud vendors- Amazon's SageMaker Notebooks, Google's Colaboratory, and Microsoft's Azure Notebook. Jupyter notebooks are used to process raw scientific data, during the first observation of gravitational waves by LIGO.

3.1.2 WEKA environment

WEKA stands for Waikato Environment for Knowledge Analysis. The Machine Learning Group at the University of Waikato started developing WEKA data science software in 1993. Ian H. Witten, a famous computer scientist is the creator of this tool. It is an open-source tool. Currently provides a vast collection of machine learning algorithms that could be easily applied to data sets. It evolved over time and currently, WEKA implements various machine learning classification methods, algorithms for regression and clustering along with several visualization tools. Using package manager, we can further install additional packages developed by the WEKA community. It is widely accepted across the academic community. For several years it was a preferred environment for research work and data mining tasks. Some commercial and enterprise-grade GUI data mining tools have been derived from WEKA, these include H2O, Rapid Miner. The main advantages of WEKA are free availability, portability, a wide collection of machine learning techniques, and ease of use.

3.2 Background Theory on Machine Learning

The entire collection of machine learning algorithms available in WEKA and Anaconda Python distribution are used in building models for every created dataset. In this thesis, only the top-performing experiments are discussed. We started with Zero R (Table 4.2) as the baseline model. The models used for comparison and analysis are listed below:

- Logistic
- NaiveBayes
- BayesNet
- J48 Decision Tree

- RandomForest
- RBFClassifier
- ZeroR

3.2.1 Logistic Regression

Logistic regression (Cox, 1958) is also known as logit model. It is a statistical model like linear regression. A linear regression (Barnard, 1989) fits a straight line or hyperplane to data points, but Logistic regression squeezes output values using a logit function between 0 and 1. It predicts the probability of a label. Logit function is defined as:

$$logistic(\eta) = \frac{1}{1 + exp^{(\eta)}}$$

Logit Function looks like below in Figure 3.1 .

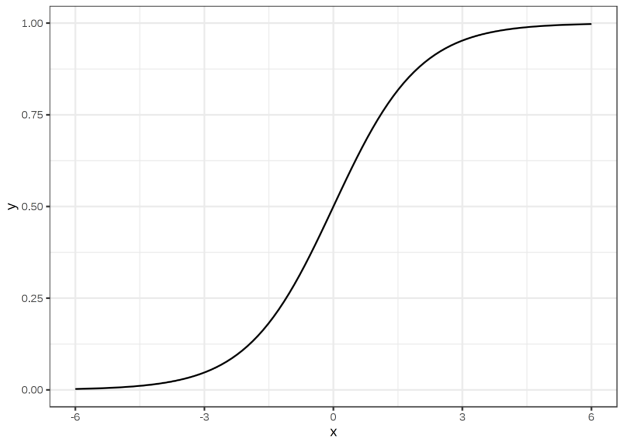


Figure 3.1: A Logit Function

3.2.2 Naïve Bayes

Naives Bayes classifier (Vikramkumar et al., 2014) is based on Bayes Theorem (Stone, 2013). It is easy to build compared to other models. Despite its simplicity, it is famous for performing better than complex models. NB classifier assumes that the effect of features(x-value) on a given class (y-value) is independent of other features(x-values). This assumption is called Class Conditional Independence. In simple terms, Naive Bayes updates prior belief of an event given new information. Spam detection in emails is one of the famous applications for the Naïve Bayes classifier.

$$P(class/features) = \frac{P(class) * P(features/class)}{P(features)}$$

- P(class/features) : Posterior Probability
- P(class) : Class Prior Probability
- P(features/class) : Likelihood
- P(features) : Predictor Prior Probability

3.2.3 BayesNet

Bayes Net (Friedman et al., 1997) is also known as Bayesian Network, Belief Network, or Decision Network. They are directed acyclic graphs (DAGs). Graphs are made of nodes and edges. Nodes represent random variables in the Bayesian sense. Edges between pairs of nodes represent the causal relationship and conditional probability distribution of each node. It is essentially a compact representation of joint probability distribution. In practice these networks get complex. They have a strong ability to capture causality. They are widely used in the health field. Just like Naïve Bayes, Bayes net is based on the Bayes theorem. A simple Bayes Net is shown with an example scenario in Figure 3.2.

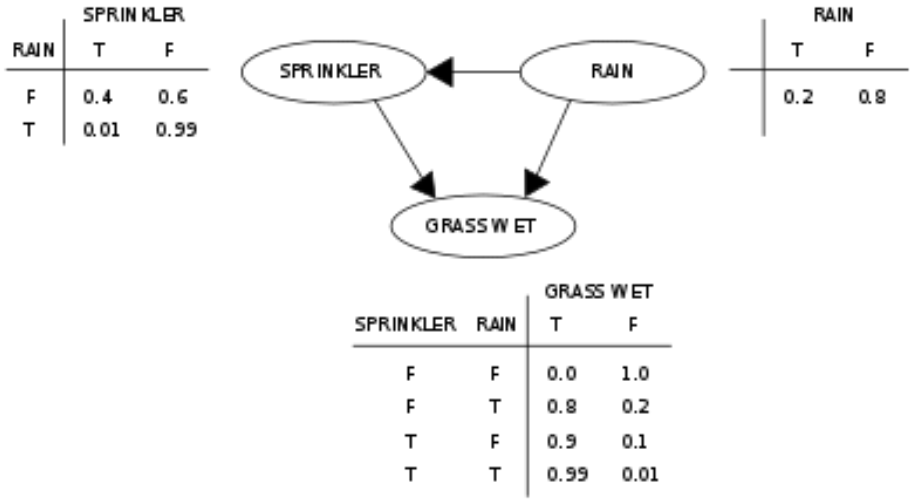


Figure 3.2: A simple Bayesian network with conditional probability tables.(Source-AnAj, 2006)

3.2.4 Decision Trees

Ross Quinlan developed the C4.5 algorithm (Quinlan, 1993). It is an extension of the ID3 algorithm(Quinlan, 1986). A decision tree has root nodes, internal nodes, leaf nodes. Below in Figure 3.3 is an example deci-

sion tree that helps us decide whether to walk or take a bus; based on weather. Weather is the root node, Time and Hungry are Internal nodes. Orange-colored boxes are leaf nodes. Concepts of entropy and information gain are used to construct a decision tree. The tree is built top-down. In WEKA, the decision tree is known by the name J48.

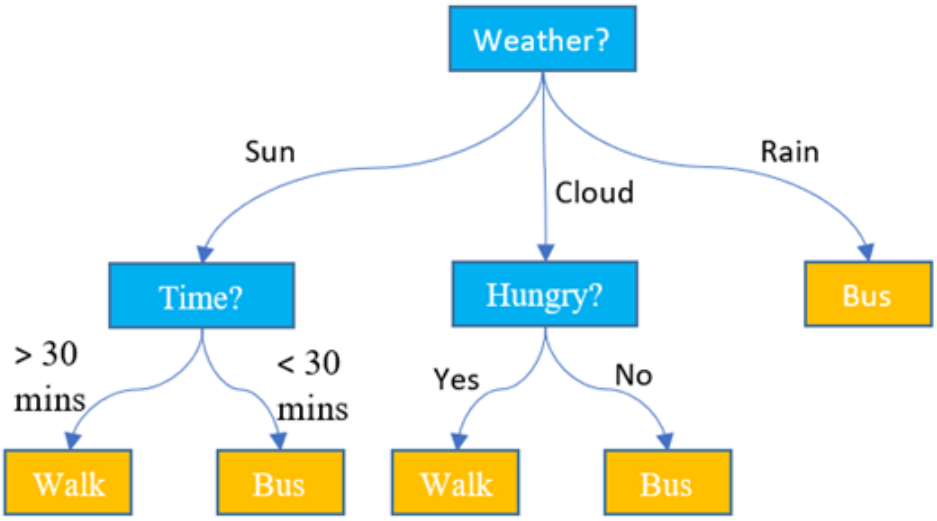


Figure 3.3: A simple decision tree.(Source-Hoare, 2021)

3.2.5 Random forest

The first algorithm of Random Forest was developed by Tin Kam Ho(Ho, 1995). Random forests (Breiman, 2001) are also called Random decision trees. It is an ensemble method. Multitudes of decision trees are constructed during training. Each tree predicts a class label. The Majority vote of labels is taken as the final label. The process is illustrated in Figure 3.4. Generally speaking, Random forests attain higher accuracies than decision trees. But perform lower than gradient boosted trees (which are discussed in further sections). Easy interpretability of decision trees is lost in random forests because of an increase in complex tree formations.

3.2.6 ZeroR

ZeroR is the simplest classification method. It stands for zero rules. The classifier simply predicts the majority category and does not have any predictive power. But is used to establish baseline performance and used as a benchmark for comparing other classification models. It is a frequency table-based model. The performance of the baseline model is the least or worst metric. Any trained model with metrics lower

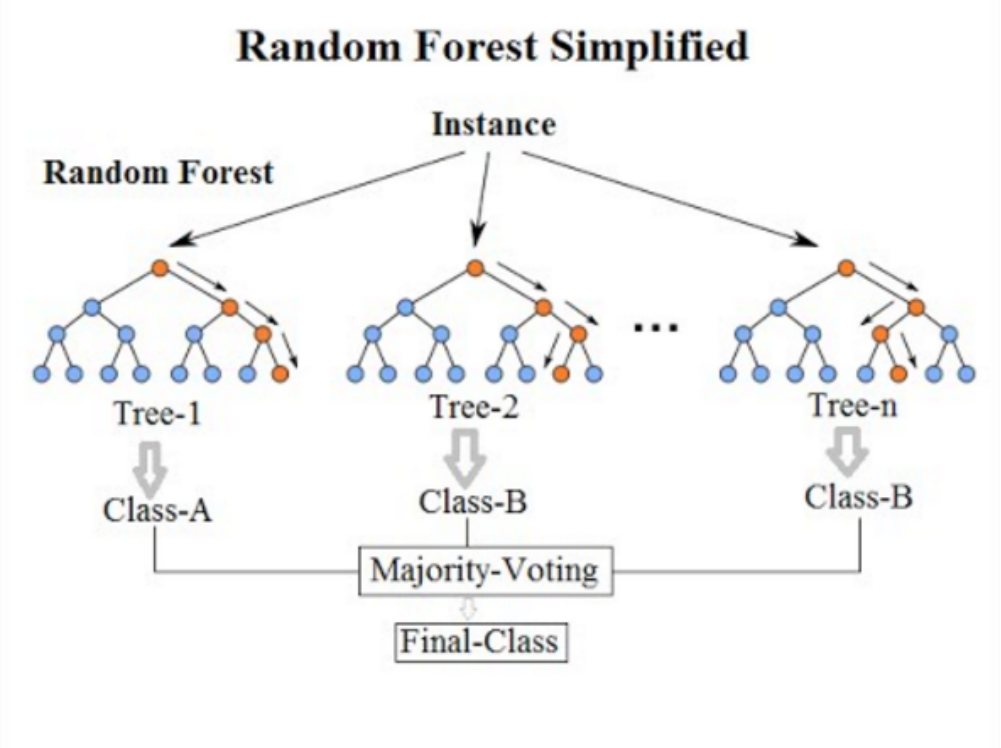


Figure 3.4: A simplified diagram of random decision forest(Source-Jagannath, 2017)

than ZeroR value or baseline performance can be confidently removed from future experiments. This baseline helps us narrow down the list of machine learning models. Saves time by only focusing on models that are showing results above baseline. ZeroR does not use features for making a prediction, it uses a frequency of classes from the frequency table.

3.3 Ensemble Machine Learning Models

"Ensemble" is derived from the word musical ensemble. Just like a group of musicians plays together with a musical piece, an ensemble machine learning algorithm combines a collection of ML models to predict a label. The performance of an ensemble is better than individual models. They are also referred to as meta-algorithms. Bagging, boosting, and stacking are different ensemble techniques.

3.3.1 LightGBM

LightGBM (Ke et al., 2017) stands for Light Gradient Boosting Machine and is developed by Microsoft Corporation. It is a gradient boosting framework that uses a tree-based learning algorithm. Advantages include:

- Faster training speed and higher efficiency.
- Lower memory usage.
- Better accuracy.
- Support of parallel, distributed, and GPU learning.
- Capable of handling large-scale data.

It uses histogram-based algorithms for decision tree learning, which reduces memory usage and increases training speeds. Generally, trees are grown level(depth) wise, but LightGBM grows trees leaf-wise(best-first). Leaf-wise tree growth as mentioned on its webpage is shown in Figure 3.5 .

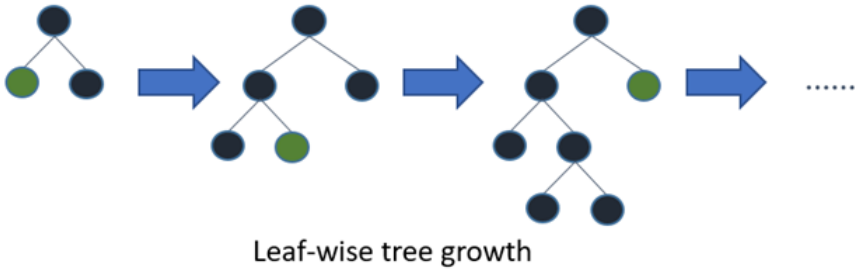


Figure 3.5: Leaf-wise Tree Growth(Source-LightGBM, 2021)

3.3.2 XGBoost

XGboost stands for eXtreme Gradient Boosting (Chen & Guestrin, 2016). It has dominated Kaggle competitions. This belongs to the family of Gradient boosting frameworks. It is designed by Tainqi Chen as part of the DMLC (distributed Machine Learning Community) group. It uses a pre-sort-based algorithm for decision tree learning. Some of the important features of XG boost as listed on the website-

- Flexible- Supports both regression and classification problems.
- Portable- supports Windows, Linux and OSX
- Multiple Languages- C++, python, R, Java, Julia, Scala
- Battle Tested- Won most Kaggle competitions and highly cited in publications.
- Distributed on Cloud- Supports distributed training on the cloud.
- Performance- High performance due to distributed load. And can be trained on billions of records.

3.3.3 CatBoost

The CatBoost (Prokhorenkova et al., 2018) is developed by Yandex. It is the successor of the MatrixNet algorithm, used widely by Yandex to rank "search results". It is the latest in the family of Gradient boosting Framework algorithms. It often outperforms other boosting algorithms. It is free and open-source. Ordered boosting, native handling of categorical attributes, and usage of symmetric trees are its main features. It was initially launched in 2017. Given dataset is divided into random permutations and boosting is applied to them. Reduces the need for hyper-parameter tuning. It is used in many tasks like weather prediction, music, or shopping recommendations. It complements deep learning by integrating deep learning models on homogenous datasets and combining them to make predictions. It is even implemented in Apple CoreML.

Chapter 4

Datasets and Features

Before we dive into experiments and discussion, initial efforts of building models were not fruitful. Early classification models often resulted in performance accuracies as low as 60%. The large data set size, missing values, high dimensionality, and labeling of data are some of the challenges specific to education data mining. These challenges coupled with limited resources gave rise to a host of issues. Getting around these issues or unblocking obstacles is key to making progress. Some of the major issues include memory issues, high latency issues while reading/querying data through SQL, and lack of skill in certain technologies that had a high learning curve. Weka is a great tool to run on small data sets on a laptop. To consume a large dataset addressing memory issues is key. Hence, we ran Weka on the HPC cluster with a maxheap value of 250GB to 800GB. This approach gave significant velocity in trying out all the algorithms. It enabled us to rapidly build and test prototypes of datasets configurations and identify relevant algorithms. And led us to differentiate between algorithms with good performance and algorithms with low performance.

Jupyter notebooks are run on the same cluster with GPU RAM of 990 GB. We used this approach to build neural networks and test out AutoML packages. Handling of missing values and labeling of data was solved by a hybrid approach. This approach involved using a combination of manual work and relying on python packages- PANDAS, NUMPY. In short, numerous sets of CSVs or ARFF files are created then processed by ML algorithms. Using trial and error, different data sets are trained and tested. In this thesis work, we mainly focus our discussion only on three data sets. In further sections selected experiments with respective datasets are discussed in detail. All trial-and-error methods followed the process as shown in Figure: 4.1 .

4.1 Datasets

Collected raw data is passed through three preprocessing stages: a) data transformation, b) data cleaning and c) attribute selection. This ensured that the resulting dataset was suitable for algorithmic consumption.

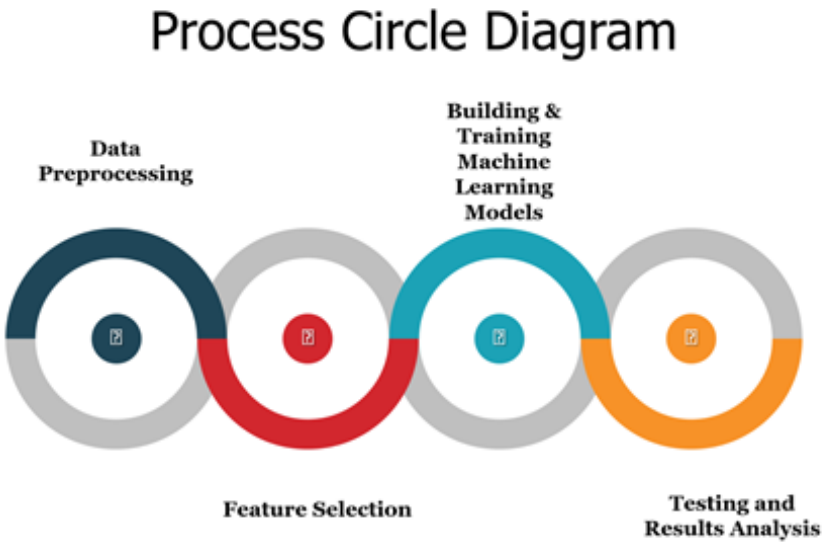


Figure 4.1: Process Flow Diagram

In data transformation, creating a general structure of the dataset with attributes and labels is done. For example- labels are converted from multiclass four labels (dropout, continuing, transferred, and graduated) set to a binary class form of Dropout and non-dropout.

This dataset is later passed to the Data cleaning phase, which deals with handling missing values and duplicate values. The duplicates are filtered using preprocessing functions in WEKA. The missing values are replaced with "-1". Manually the data was analyzed to look at columns with high missing values and very few distinct values. These columns are checked for Cramér's V coefficient. All dataset configurations are verified for the presence of columns with Cramer's V coefficient closer to 1. We did not identify any columns that had coefficients closer to 1. This can imply that the common problem of y-value leakage into predictions is highly unlikely. Columns with coefficient values closer to 0 are selected. These columns are marked and removed when several datasets are created with various configurations.

At any given point, an experiment consisted of a combination of a dataset, technology, algorithm, and hyperparameters. As we conducted experiments, we learned along the way that the search space of experiments was getting exponentially high. So once the results proved to be ineffective, the phase of research was dialed back to the initial stage, and a search for a new configuration of attributes and labels was initiated. We used an approach of inspecting, adapting, and continuously iterating. This approach is borrowed from agile methodologies prevalent in technology firms like Google, IBM, Amazon, or Facebook. Stacey matrix model(Figure: 4.2(Source-d Stacey, 1996)). Cynefin framework is also (Figure: 4.3(Source-Dave Snowden, 2003)) widely used in industry for identifying the need for agile methodologies.

Table 4.1: Dataset rows, columns, and data points

Dataset	Rows	Columns	Datapoints
DS-101	42243	101	4266543
DS-57	29203	57	1664571
DS-11	29203	11	642466

When we break down the research problem, we can see that student dropout project requirements are highly unclear, and technology to implement conceived solutions is constantly evolving. For example- a laptop would be sufficient to train a model for a small dataset related to the semester. But as we scaled up to include over 10 years of data, complexity grew and turned quickly to a big data problem. A team of Ph.D. students with different skill sets and resource constraints has added another layer of complexity. When we weigh in all these variables, the research problem can be positioned into a Complex zone (Figure: 4.2). When in a complex zone, the agile method is an efficient way to drive a research project. Hence, a deliberate choice was made early on to structure research experiments using agile.

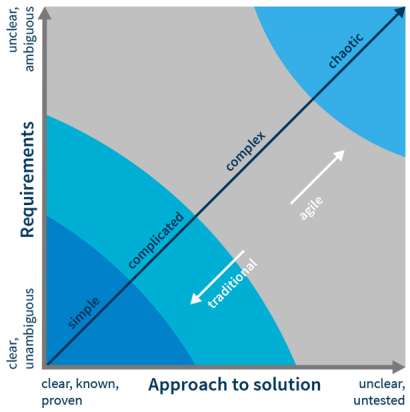


Figure 4.2: Stacey matrix model(Source-Stacey, 1996)

Each category of experiments would begin with wrangling and manipulating raw data to form a dataset. Then technology would be used for early rapid training, with the first choice as WEKA. If the results are not promising, the dataset would be discarded. The research project is pushed back to the drawing board and started from the beginning. If the results are promising, then the experiments are pushed forward to use advanced libraries offered by SCI-KIT learn. More analysis is performed by pushing the experiments to search for the entire search space of hyperparameters. This agile iteration has resulted in many discarded datasets and low-performing experiments but created a streamline of the process that saved time and led us to three promising datasets, refer to Table-4.1. It can be noted that DS-101 has the highest number of rows, columns, and data points.

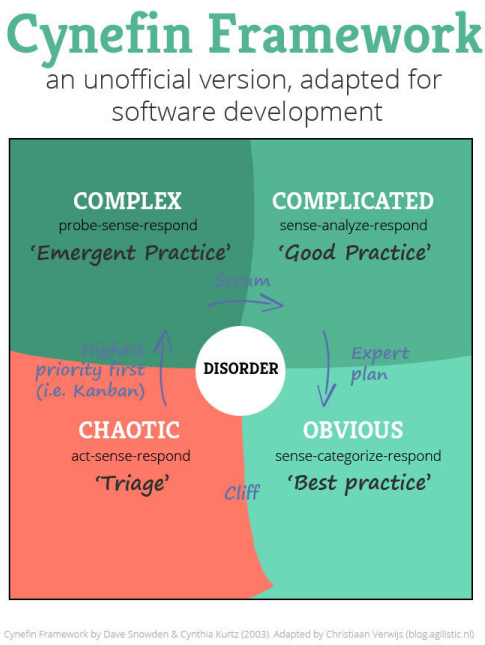


Figure 4.3: Cynefin framework(Source-Snowden and Boone, 2003Snowden and Kurtz, 2003)

Table 4.2: Baseline accuracy-ZeroR on all three datasets

Dataset	DS101	DS57	DS11
Classifier Model name	ZERO-R	ZERO-R	ZERO-R
Baseline Accuracy	25.8	52.3	52.3

Baseline results from the ZeroR algorithm on all datasets are shown in the table below. Baseline results can be used as a measure of progress to measure improvements. Any model that performs worse than baseline results can be identified and removed in future experiments.

4.2 Features

Few features are listed and described below:

1. Grants All- This attribute includes the sum of all financial aid received by students. Pell Grant, HOPE Scholarship, Zell Miller Scholarship are some of the types of financial aid a student can receive.

2. COMM- Applicants whose GPA is below the minimum for college level placement are encouraged to submit SAT/ACT or complete the ACCUPLACER test series. ACCUPLACER test series are also called COMPASS. COMM indicates grade in COMPASS ALGEBRA/MATH.

3. COME- COME indicates grade in COMPASS Writing English.
4. COMR- COMR indicates grade in COMPASS Reading English.
5. CROW_DISTANCE-The direct distance between points- students place and college.
6. DRIVE_DISTANCE Actual Driving distance between students place to college
7. DRIVE_DURATION Indicate driving time from leaving students place to college

The OVERALL attribute is a combination of Transferred(TRANSFER) and institutional(INST) values. A student is transferred from another institution to the current institution. These persons- transfer gpa, hours and points are captured into the college system. Institutional values gpa , hours and quality points are recorded when the student is the current institution. Overall, Transferred and Institutional values are listed below:

1. OVERALL_LGPA_GPA- Total combined GPA.
2. OVERALL_LGPA_HOURS_ATTEMPTED- Total combined attempted hours
3. OVERALL_LGPA_HOURS_EARNED- Total combined earned hours
4. OVERALL_LGPA_HOURS_PASSED- Total combined passed hours
5. OVERALL_LGPA_GPA_HOURS- Total combined hours
6. OVERALL_LGPA_QUALITY_POINTS-Total combined quality points.
7. INST_LGPA_GPA- -Total institutional GPA.
8. INST_LGPA_HOURS_ATTEMPTED-Total institutional hours attempted.
9. INST_LGPA_HOURS_EARNED–Total institutional hours earned.
10. INST_LGPA_HOURS_PASSED-Total institutional hours passed
11. INST_LGPA_GPA_HOURS-Total institutional hours.
12. INST_LGPA_QUALITY_POINTS- Total institutional quality points.
13. TRANSFER_LGPA_GPA- Total transferred GPA.
14. TRANSFER_LGPA_HOURS_ATTEMPTED-Total transferred attempted hours.
15. TRANSFER_LGPA_HOURS_EARNED-Total transferred earned hours.
16. TRANSFER_LGPA_HOURS_PASSED- Total transferred passed hours.

17. TRANSFER_LGPA_GPA_HOURS- Total transferred hours.
18. TRANSFER_LGPA_QUALITY_POINTS- Total transferred quality points.

final label is listed below:

1. Outcome- Class label incase of two labels it is Dropout and Non-Dropout and incase of four labels, the values are Dropout, Graduated, Transferred and continuing.

All Features cannot be discussed due to proprietary reasons. DS-11 is the final dataset with eleven features which has good performance and consumes a small feature set for training. All the features in DS-11 are elaborated along with few additional related features.

4.3 Feature Selection

Attribute selection or feature selection is a method by which one can select the best subset of attributes/features in a dataset. Some of the benefits of this method are: reducing overfitting, improves accuracy and reduces training time. Weka is used to perform feature selection. Some of the Attribute evaluation methods used are:

1. CfsSubsetEval - This method selects a subset of features which are highly correlated with the class attribute and have low correlation with each other.
2. ClassifierSubsetEval- This method estimates the merit of a subset of features using a classifier.
3. WrapperSubsetEval- This method evaluates subsets of features by using a learning scheme. Cross validation is used to estimate the accuracy of the learning scheme for a set of attributes.
4. CorrelationAttributeEval - Correlation also referred to as Pearson's correlation coefficient in statistics. Weka implements this technique but requires use of a Ranker search method.
5. InfoGainAttributeEval-Information Gain Based Feature Selection.Features that contribute more information will have a higher information gain value and will be selected, whereas those that do not add much information will have a lower score and will be removed. Like the correlation technique above, the Ranker Search Method must be used.

Sometimes the selection method would return irrelevant attributes and leave out most important attributes. This was taken care of by reviewing the selected attributes with domain experts. A hybrid approach, in which automation of feature selection is later complemented by domain expert review. The main DS_57, DS_11 are created by hybrid approach.

4.4 Feature Importance

From the feature importance map, we can note that the most important features are related to personal academic performance and financial aid. The specific details on importance can be seen in data set DS-11(Figure: 4.4) This importance is calculated using sci-kit library packages on Random Forest.

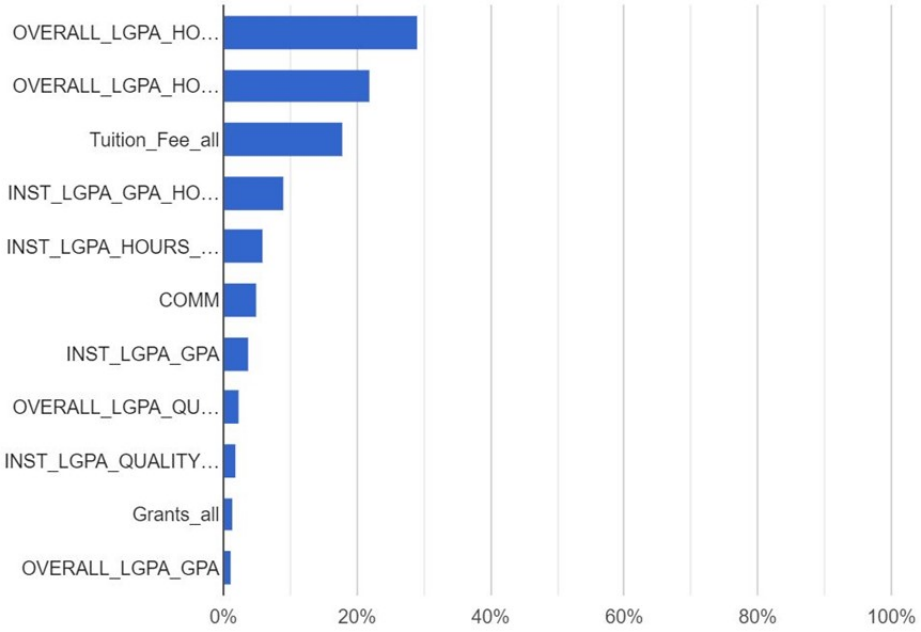


Figure 4.4: Feature importance in DS-11

Chapter 5

Experiments and Discussion

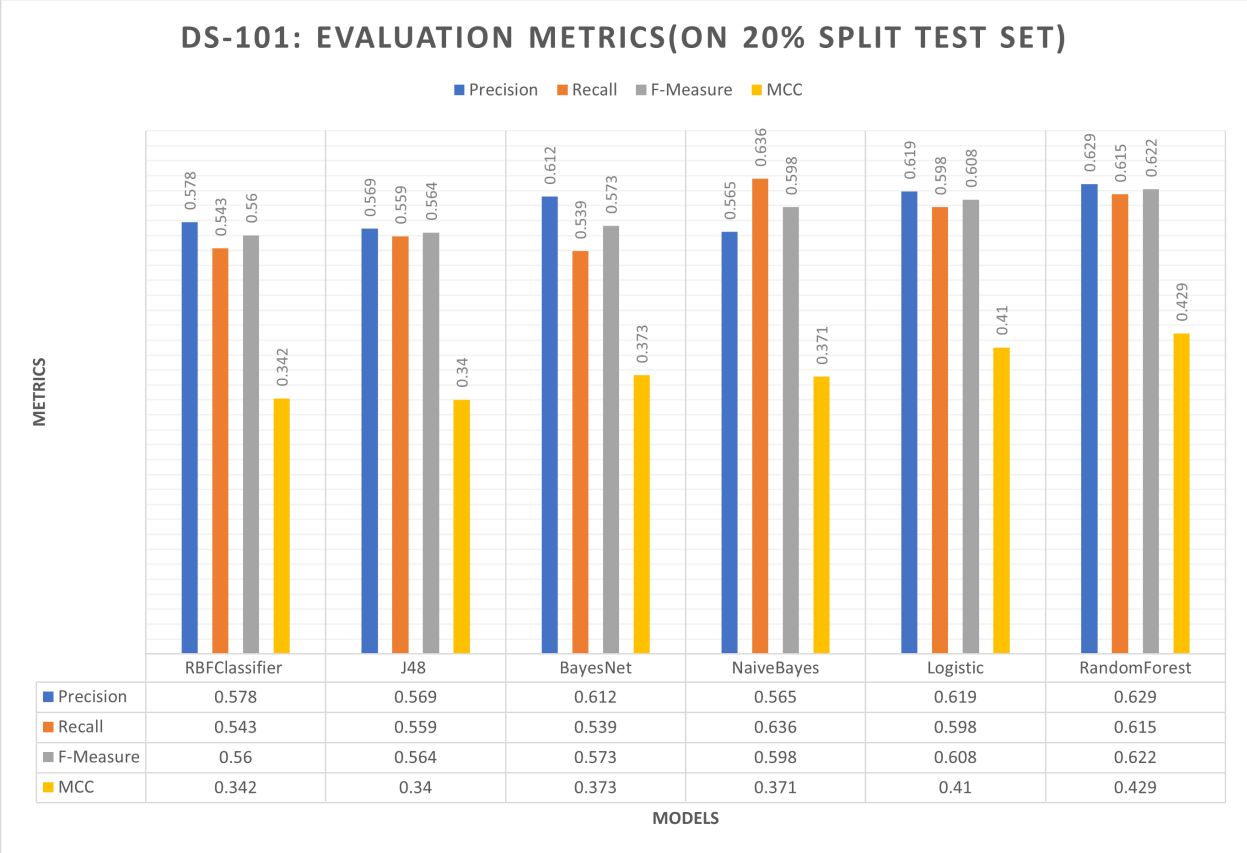
In this chapter details of experiments are discussed. Each experiment is a permutation of the machine learning algorithm, its hyperparameters, dataset, technology, and attribute selection methods. Over 300 experiments are performed. Discussing every experiment would be inefficient and verbose. Hence only a few sets of important experiments that led to major performance breakthroughs are described. These experiments can be primarily categorized based on datasets. There are three categories of experiments based on the dataset, namely DS-101, DS-57, DS-11. The first section examines classification models used, performance metrics, frameworks, and common experimental setup. The next three sections talk about results from DS-57, DS-11, and Auto Gluon. The last section further explains and investigates the results and predictive power of various attributes.

5.1 DS-101

DS-101 is initially split into 80% train set, 20% test set. Weka is used to train the models. The list of model and evaluation metrics are listed in table -ds-101-split. No new preprocessing steps have been applied to the data. This is the only dataset with four labels. This is a multilabel classification problem. Hence the appropriate metric to utilize is F-score. The trends are plotted in Figure : 5.1.

Table 5.1: DS-101 Evaluation metrics on test set in split mode.

Model= Split	Precision	Recall	F-Measure	MCC
RBFClassifier	0.578	0.543	0.56	0.342
J48	0.569	0.559	0.564	0.34
BayesNet	0.612	0.539	0.573	0.373
NaiveBayes	0.565	0.636	0.598	0.371
Logistic	0.619	0.598	0.608	0.41
RandomForest	0.629	0.615	0.622	0.429



	RBFClassifier	J48	BayesNet	NaiveBayes	Logistic	RandomForest
Precision	0.578	0.569	0.612	0.565	0.619	0.629
Recall	0.543	0.559	0.539	0.636	0.598	0.615
F-Measure	0.56	0.564	0.573	0.598	0.608	0.622
MCC	0.342	0.34	0.373	0.371	0.41	0.429

Figure 5.1: DS-101 Evaluation metrics trends in Split mode

One thing to note about the RBF classifier here is- it uses the BFGS(Broyden–Fletcher–Goldfarb–Shanno) optimization algorithm to minimize the cost function. The learning rate is automatically picked for this algorithm, due to which arriving at convergence is faster, compared to the gradient descent optimization method. There is an option to use conjugate gradient descent instead of BFGS for faster training, but because the training was being performed on a high performing cluster, we chose BFGS. These algorithms are beyond the scope for discussion, they are often debated in the context of advanced numerical computing. Hyperparameter settings for the RBF classifier are shown in Figure: 5.2.

It took 160.9 seconds to train the model on 33794 instances, and 0.16 seconds to test on 8449 instances. We tried changing the poolsize hyper-parameter to 30 threads, the training speed increased to 180.16 seconds.

J4.8 generates a pruned C4 decision tree. The setting unpruned is set to false, and 0.25 is selected as a confidence factor which will incur more pruning. CollapseTree is set to True. It will remove parts of the tree if the training error is not reduced. An overview of hyper-parameters for J4.8 is shown in Figure: 5.3.

Similarly, other model hyperparameters are tweaked and tuned to get better F-scores. It can be noted that there is a vast search space for hyperparameters. This will pose an obstacle and increase time and resources exponentially if we were to try all hyperparameters manually. To address this challenge Auto ML is utilized, results of which are discussed in later sections.

From preliminary results, we can infer that RBF Classifier and J48 show similar f-score. In other models, the F-score follows an upward trend starting with Bayes net, Naïve Bayes, Logistic. The highest metric of performance is achieved by Random Forest, with 0.622. It can be also noted that the selected configuration of the experimental setup has the lowest F-score of 0.622. Initially to save time we used the split mode or hold out method. DS-101 is further investigated with a cross-validation approach as well. The results are shown in ds-101-cv.

In both cases, trained models are as good as a coin toss. The current configuration does not suit the practical application. From this batch of experiments, we got the below insights. We are presented with three different choices 1- to gather more data 2- Maybe class labels need to be balanced to reduce the error rate. 3- Discard the dataset and proceed with a different configuration. We had done a couple of data gathering studies and understood that the cost of collecting more data is high, hence the first choice is dropped. We moved to the next option, addressing the class imbalance. Class balancing techniques like SMOTE (Synthetic Minority Oversampling Technique), and others are applied to the dataset and then more experiments are done. After a few experiments, the pattern in results emerged, most of the experiments ended with low-performance metrics. This led us to a third option to rethink our approach and create another dataset.

After the decision is made on the third approach, brainstorming sessions are conducted with subject matter experts or advisors. From the feedback of advisors, reducing class labels from four to two are conceived. The main class label "Dropout", which is the focus of the study, is retained. Other class labels, graduated, transferred, continuing, are diluted, and merged to form a super class-label named- "Non-Dropout" This approach gave several benefits- without increasing rows, more data is distributed evenly among two classes, thereby mitigating class imbalance. Ensured maximum usage of existing data, which gives the model more chance to train. It has also reduced development time associated with constructing new datasets as we can rely on existing python code and just replace labels with a new name. We call this configuration of data as DS-57.

5.2 DS-57

DS-57 is created by using attribute selection methods over 100 attributes, ranking them, and manually evaluating them for relevance. After vetting all attributes, 57 attributes are selected to a dataset. This is imported in WEKA and trained on several models. The first batch of training is accomplished in split mode with 80% train, 20% test. The results are summarized in below table

For Visualization purposes, DS-57 Evaluation metrics with split mode are shown in Figure: 5.5.

Similar hyperparameters from the DS-101 batch are used for all the models. For RBF and J4.8 hyperparameters are already discussed in the DS-101 section. Hyperparameters for Random Forest are shown

Table 5.2: DS-57 Evaluation metrics on test set in split mode.

Model= Split	Precision	Recall	F-Measure	MCC
RandomForest	0.896	0.873	0.884	0.784
J48	0.855	0.849	0.852	0.72
RBFClassifier	0.867	0.837	0.852	0.723
Logistic	0.866	0.855	0.86	0.737
NaiveBayes	0.709	0.834	0.766	0.529
BayesNet	0.782	0.784	0.783	0.587

Table 5.3: Random Forest maxDepth hyperparameter

tree depth	f-measure	accuracy
1	0.779	80.458
2	0.791	81.54
3	0.822	83.92
7	0.86	87.09
14	0.88	89.16
57	0.88	89.21
0	0.88	89.21

in Figure: 5.6. maxDepth hyperparameter signifies maximum tree depth. If the value is set to 0, the training can grow a tree to unlimited depth. We tested with varying depths of 1,2,7,14 and 57. The values of F-measure vs tree depth are shown in the table below.

As the depth increased, results improved. A depth of 57 was set due to the number of features in the data set is also 57. Also, it can be noted that results at 57 value and unlimited depth (maxDepth==0) are similar. Depth hyperparameter is tested with similar values in the remaining datasets. No major deviations are observed in the results. If there are limitations in computing capacity, it is recommended to select lower values of maxDepth. In the HPC cluster, we can train fast on maxDepth value set at 0, which will enable Random Forest to grow to unlimited depth.

Moving on to the discussion of results on DS-57; significant improvement is achieved over the previous ds-101 configuration. Random forest performed better compared to other models in both dS-101 and ds-57 based on F-scores for dropout labels. Let us have a closer look at the F-score on both classes-Dropout and Non-dropout depicted in Figure: 5.7.

Hold out the approach or split approach is a sufficient and fast way to train the model. Even though cross-validation is not necessary once the hold-out approach is used. We went ahead with the cross-validation approach to know training times for different models, especially the lazy learning algorithms. SVM took the longest time on cross-validation. Experiments are repeated with cross-validation to observe training times. We can also note similarities in the results of split mode. Cross-Validation results are shared below table.

Table 5.4: DS-57 Evaluation metrics on test set in Cross-validation mode

Model= CrossValidation -10	Precision	Recall	F-Measure	MCC
RandomForest	0.9	0.863	0.881	0.775
J48	0.881	0.845	0.863	0.741
Logistic	0.868	0.846	0.857	0.727
RBFClassifier	0.866	0.827	0.846	0.709
BayesNet	0.776	0.781	0.779	0.572
NaiveBayes	0.703	0.833	0.762	0.511

For Visualization purposes, DS-57 Evaluation metrics with Cross-validation mode are shown in Figure: 5.8.For easy visualization and analysis, the plot is also shown in the figure. The F-score for both classes is shown in the next Figure: 5.9 .

It can be summarized that so far, the highest accuracy was above 80%. Let us continue with additional experiments to maximize F-score and accuracy beyond 95% accuracy or .95 f- score, respectively. We tried further configurations by reducing the feature set, mixing, and matching models, and their hyperparameters. In this process, we created the DS-11 dataset. Let us discuss the results in the next section.

5.3 DS-11

With the dataset DS-11 results, we can observe a decreasing trend in F-score. Random forests remains the strong candidate in the set of models tested. Between DS-57 and DS-11, we have tried other configurations of data sets but at DS-11, we have seen a decreasing trend in performance. For Visualization purposes, DS-11 Evaluation metrics with split mode are shown in Figure: 5.10.

For Visualization purposes, DS-11 Evaluation metrics with Cross-Validation mode are shown in Figure: 5.11.

We have progressively elaborated data configurations through trial and error. The best candidates with the right fit between model performance and attributes are identified to be DS-57 and DS-11. We have approximately found three possible candidates of datasets that show a varying degree of performance in student dropout. Accuracy with DS-101 is low, with DS-57 it was high, and ds-11 it started to decrease again.

5.4 AutoGluon

We continue the journey of experimentation with a prime focus of improving F- score and accuracy. WEKA has served the purpose of narrowing down the feature set and generating possible datasets from raw data. We now shift to the Anaconda framework, to take advantage of a wide collection of libraries and advanced functionalities. From previous experiments, we can say that optimizing hyperparameters is

a tiresome process. Oftentimes optimizing for hyperparameters can be difficult to debug and implement. With these constraints in mind, we started exploring AutoML solutions. After considerable experimentation with a few solutions, we chose AutoGluon (Erickson et al., 2020) as the main candidate for our task. From the citation, AutoGluon boasts better results compared to other solutions.

AutoGluon is run on Jupyter Notebooks in the HPC cluster at 990 GB Ram and 48 CPU cores. Experiments are executed on two viable datasets(DS-57, DS-11) in AutoGluon. Results are presented below. Figure: 5.12 and Figure: 5.13. Let us further analyze the results. The highest accuracy on DS-57 is achieved by the WeightedEnsemble_L2 model, which is .90. And very close are a list of ensemble-tree-based models and neural-network-based models. It is also observed that a new group of ensemble models performs better than Random Forest. Random Forest ranks 6th position from top in DS-57 category and 7th position in DS-11 category. If we compare results for DS-11, we note that there is no significant difference from classical machine learning model results.

5.5 Results Discussion

Results presented in this study differ from research work discussed in earlier sections. It is hard to compare student data analyses from institutions of Seattle or Toronto or Karlsruhe. The collection of data, localization of features, and various other hosts of reasons add to the complexity. Hence results are not compared with similar research work. It should be referred to and viewed as an independent study. Summary of best performing models on three data sets from over 50 experiments in three different frameworks is presented in the table.

Based on the below results, how do we choose the best model? In pure computer science terminology, Random Forest wins the performance race. Other similar tree-based ensemble models also yield better results. But the best model implementation is the one that serves the real goal of addressing the student dropout problem, in a real-world setting. To answer this question, we need to take additional factors into consideration like the effort of collecting data, training model, and deploying in the right environment (either mobile or web service or even a secure workstation). In terms of collecting data, DS-11 wins over DS-57 due to fewer attributes. This will play a significant role when student data is collected to check for data drift in the inference model. The prediction service can be made accessible to different users, like students, university administrators, or even guardians. Based on the appropriate use case, it will either be deployed in the mobile app or as a web service. If it is a mobile app, it should be a lightweight model file. To serve this purpose, further load testing and performance testing should be done. Weka would not be a great fit for deployment. But it offers significant advantages for early-stage exploration. For practical production deployment applications, using sci-kit learn, AutoGluon, or heterogeneous cloud environment as deployment options are effective. Based on individual institution needs, a suitable option can be selected. In this research, we have demonstrated models that fit real-world implementations.

Table 5.5: Results on DS-101

S.no	Model Name	Framework	F-score
1	Random Forest	Weka-split	0.622
2	Random Forest	Weka-CV	0.622

Table 5.6: Results on DS-57

S.no	Model Name	Framework	F-score	Accuracy
1	Random Forest	Weka-split	0.884	89.2123
2	Random Forest	Weka-CV	0.881	88.7724

Table 5.7: Results on DS-11

S.no	Model Name	Framework	F-score	Accuracy
1	Random Forest	Weka-split	0.864	87.1361
2	Random Forest	Weka-CV	0.863	87.0473

Table 5.8: AutoGluon Models on DS-57 in 80/20 split mode

S.no	Model Name	Accuracy
1	CatBoost	0.902243
2	RandomForestEntr	0.902756
3	LightGBMLarge	0.903784
4	XGBoost	0.904811
5	LightGBM	0.904982
6	NeuralNetFastAI	0.907892

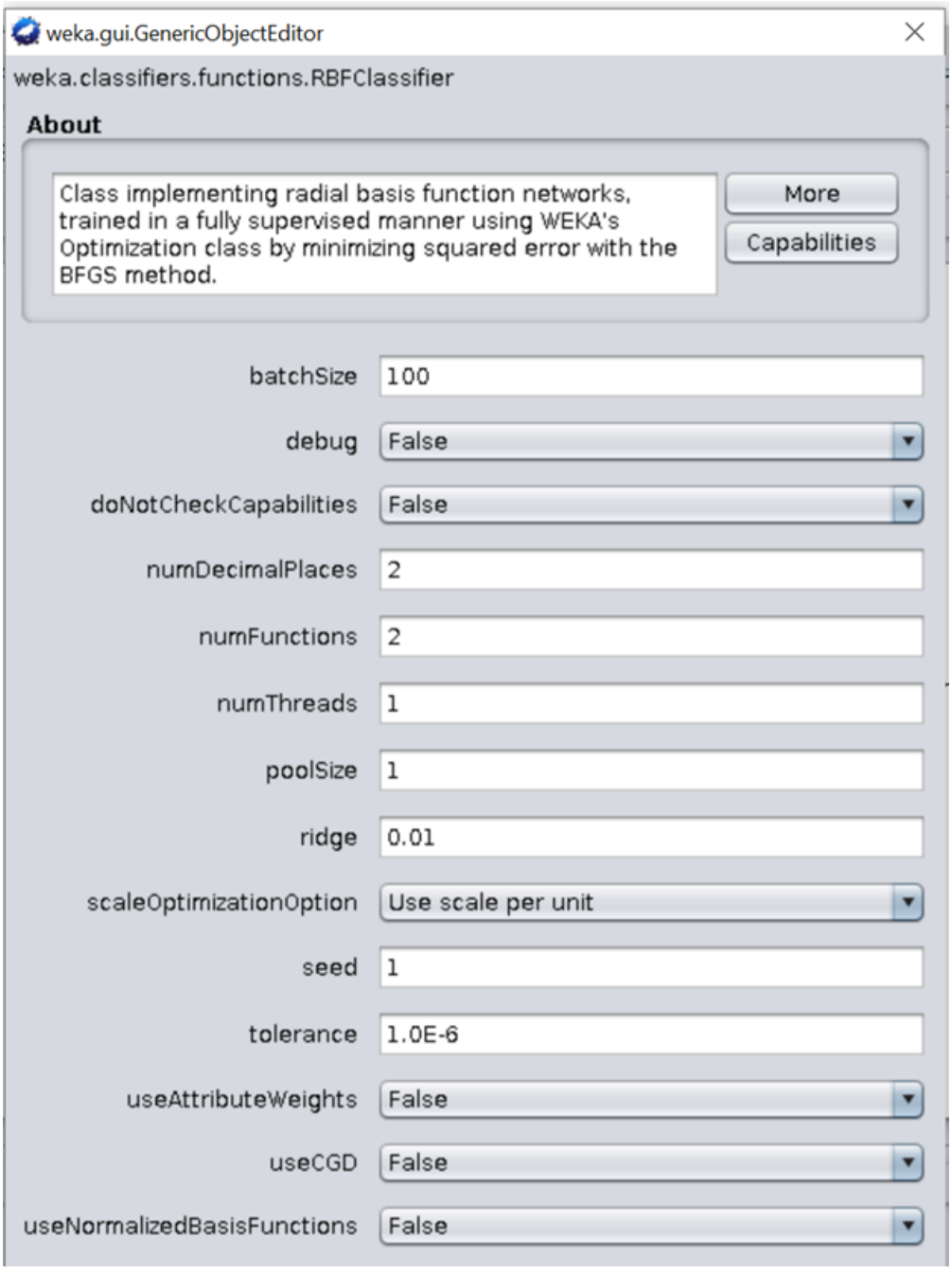


Figure 5.2: RBF hyperparameters

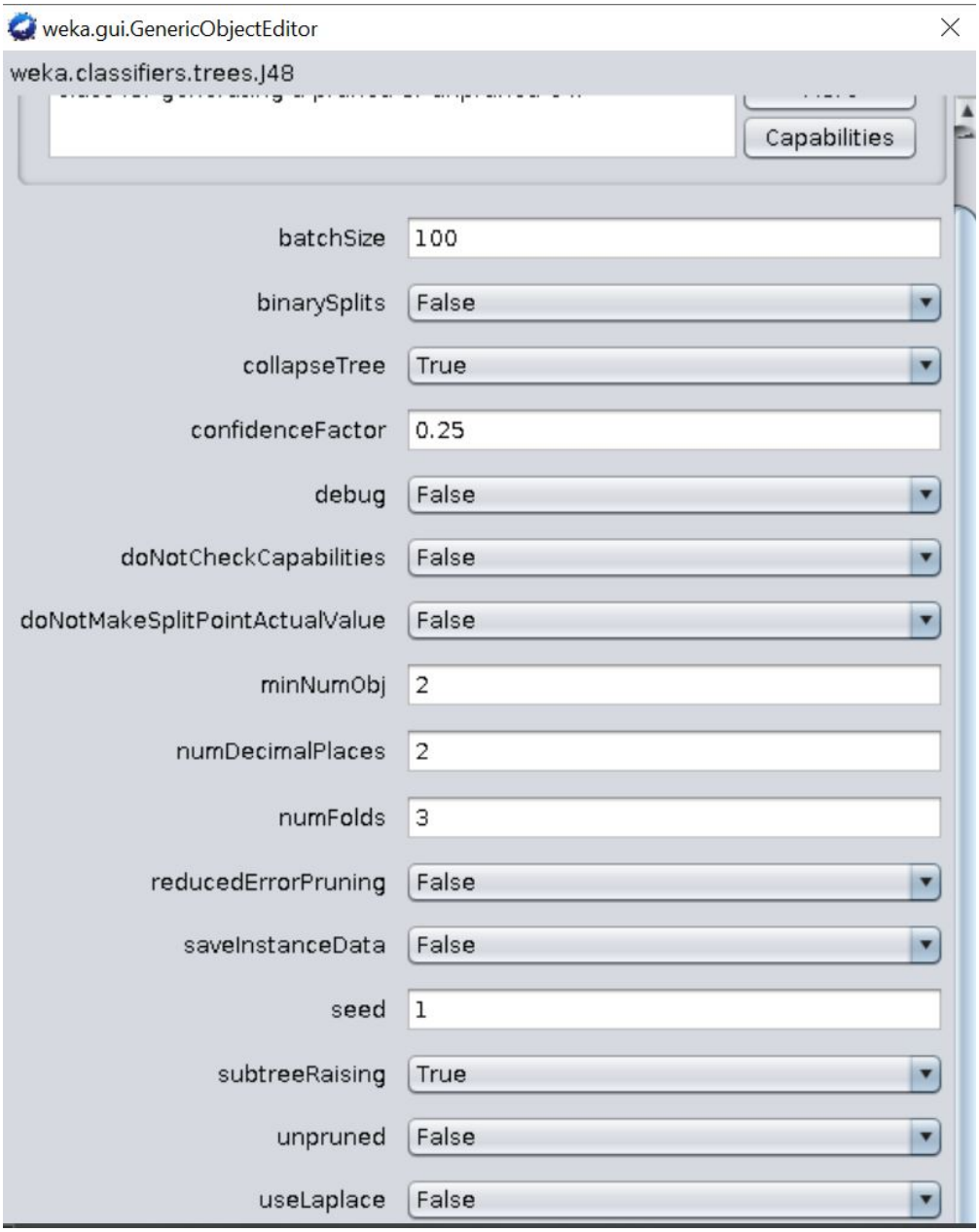


Figure 5.3: J4.8 Hyper-parameters

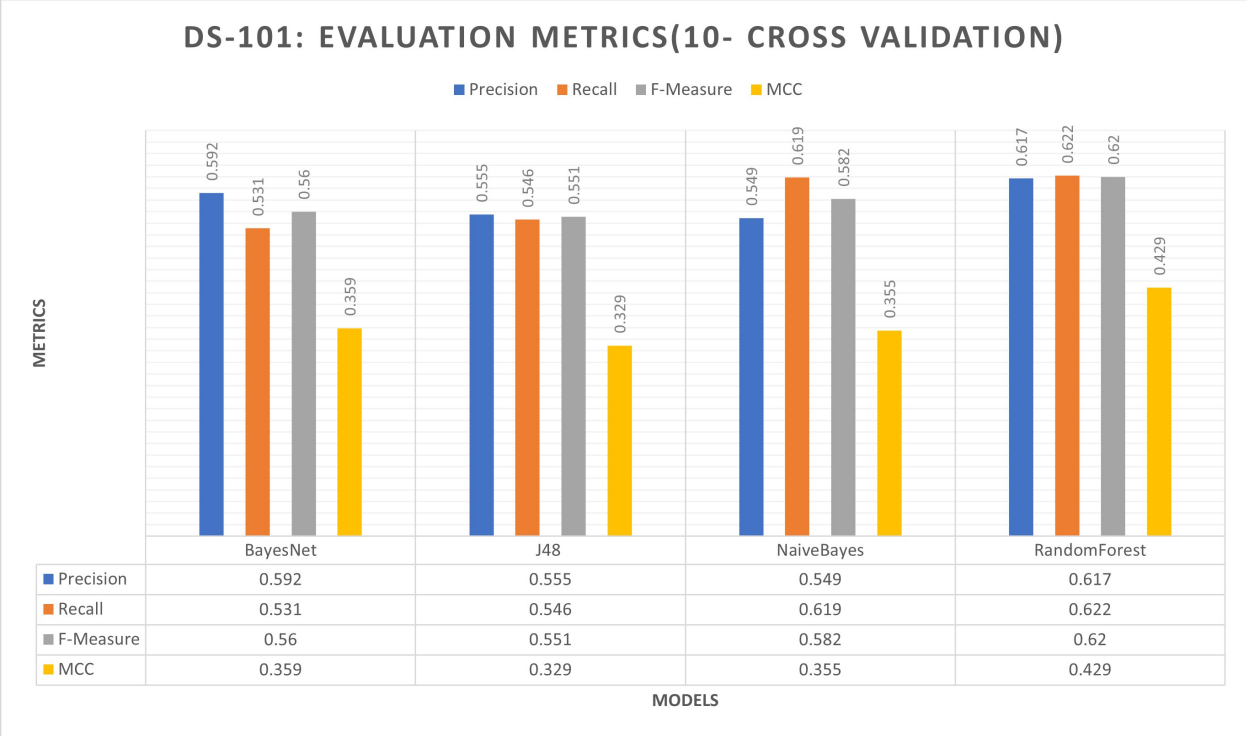


Figure 5.4: DS-101 Evaluation metrics trends with Cross-Validation mode

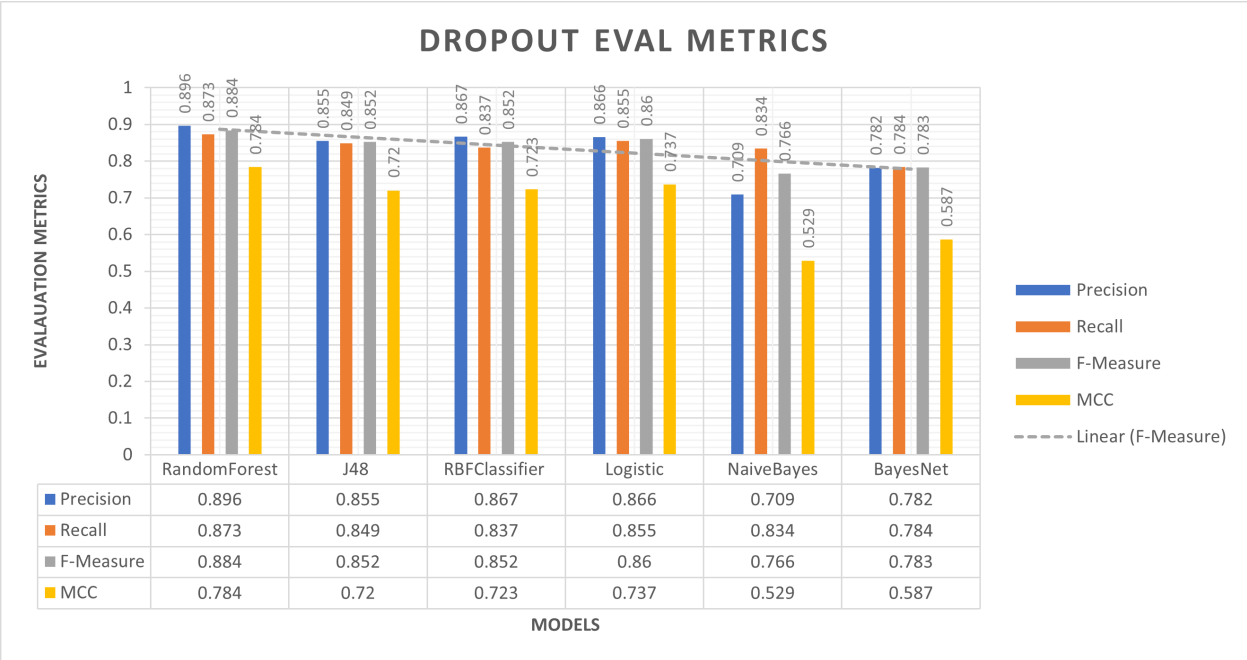


Figure 5.5: DS-57 Evaluation metrics trends with Split mode

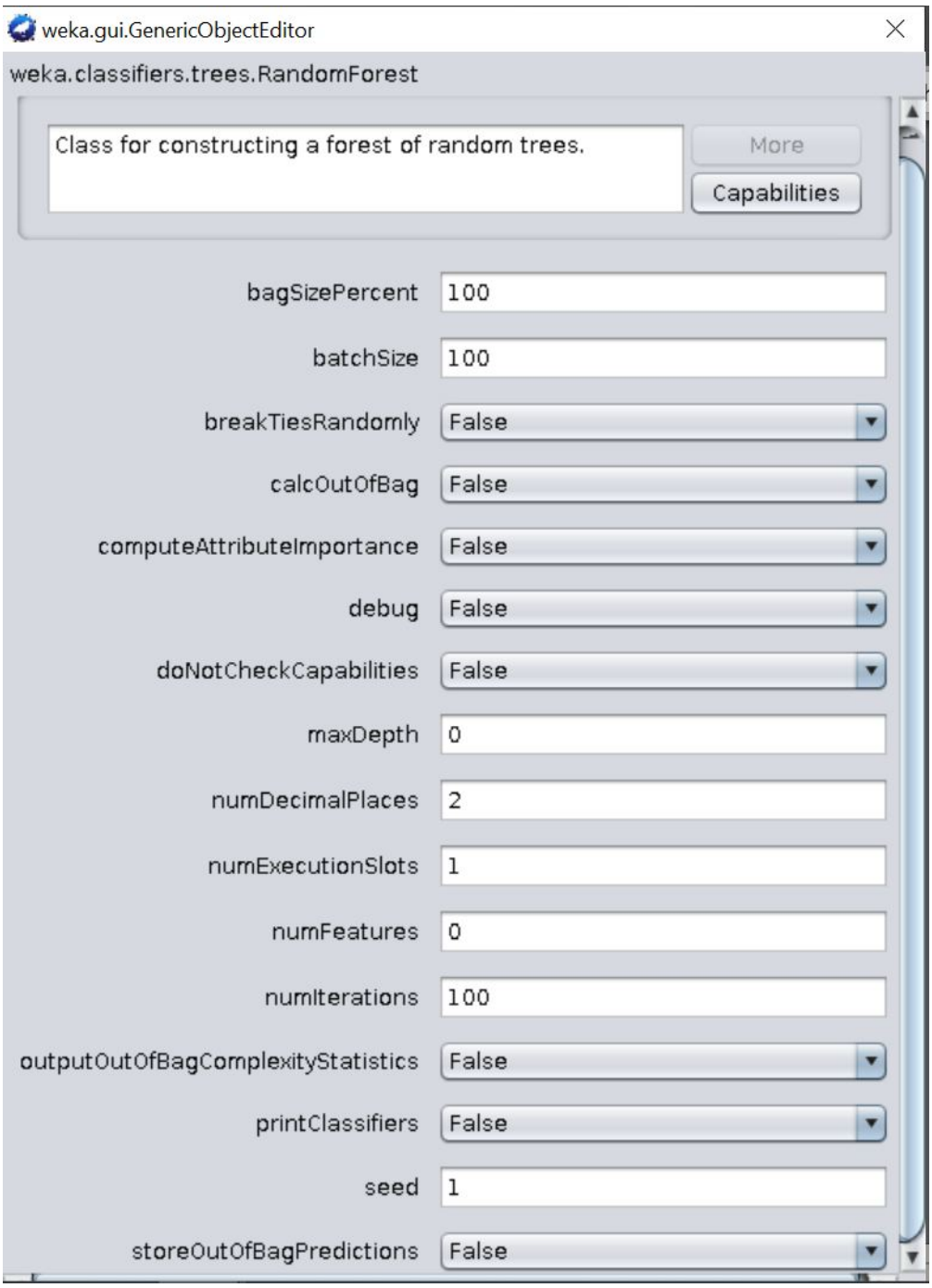


Figure 5.6: Random Forest hyperparameters

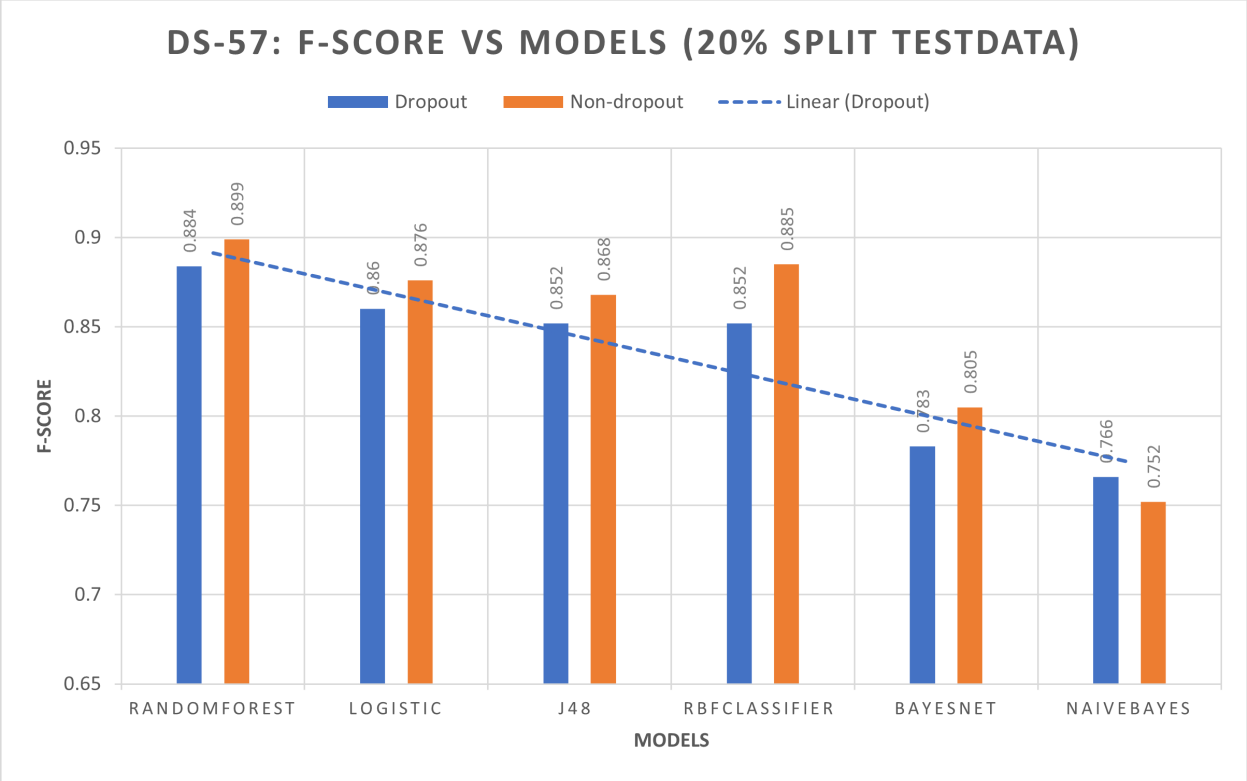


Figure 5.7: F-scores on DS-57 on both classes- Dropout and Non-Dropout in split-mode

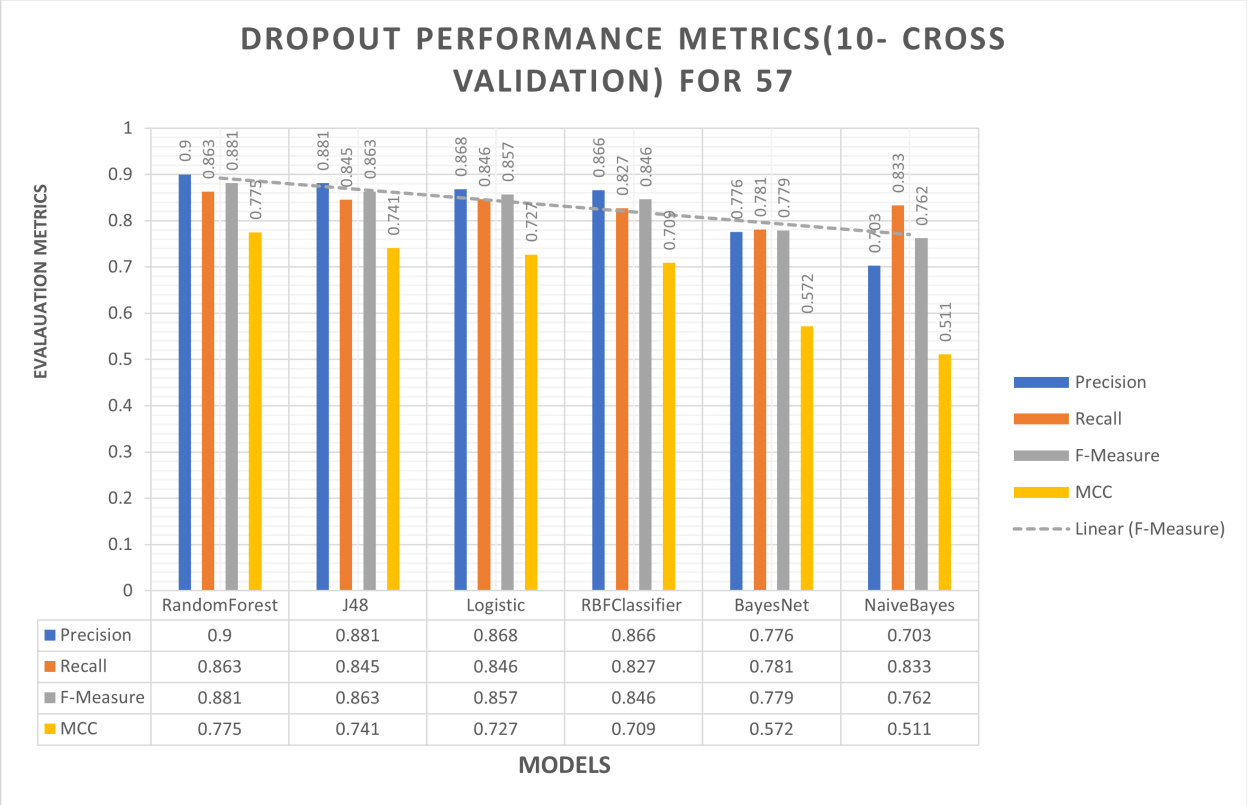


Figure 5.8: DS-57 Evaluation metrics trends with Cross-validation mode

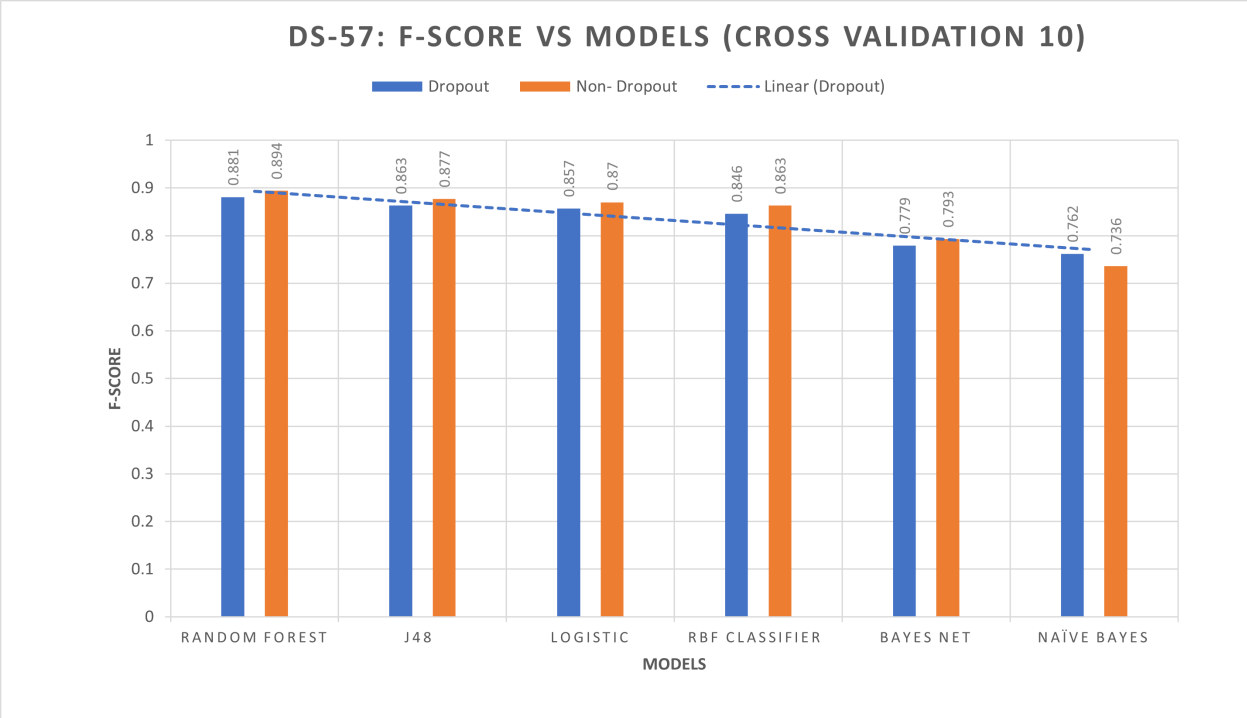


Figure 5.9: F-scores on DS-57 on both classes- Dropout and Non-Dropout in Cross-Validation mode

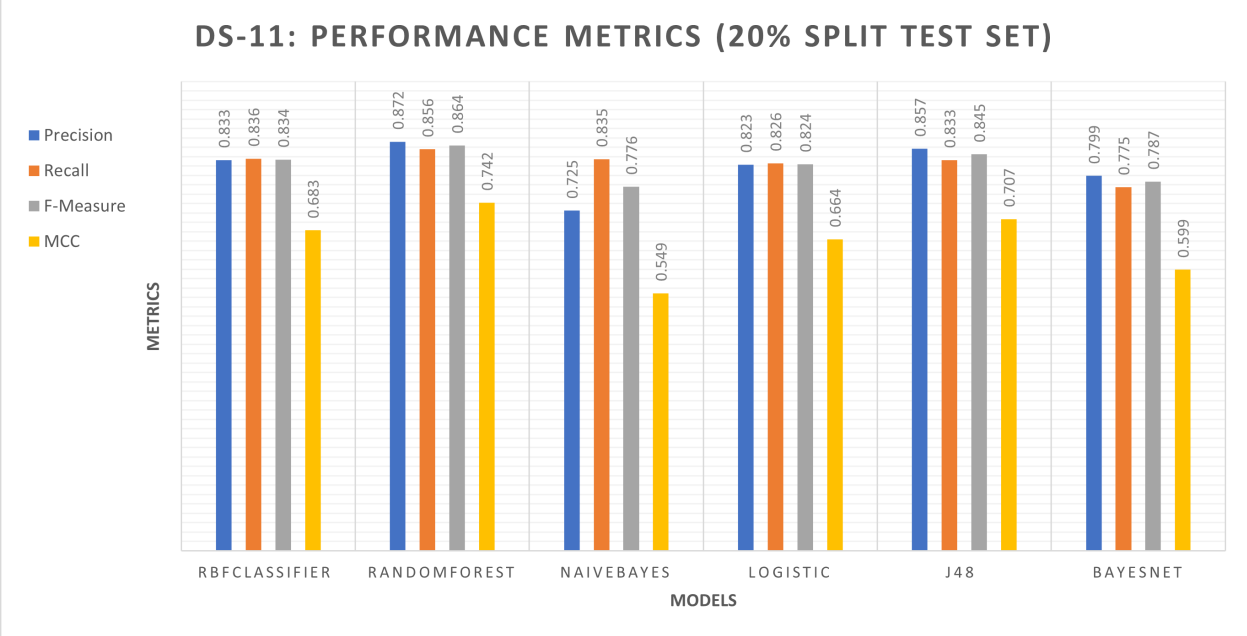


Figure 5.10: DS-11 Evaluation metrics trends with Split mode

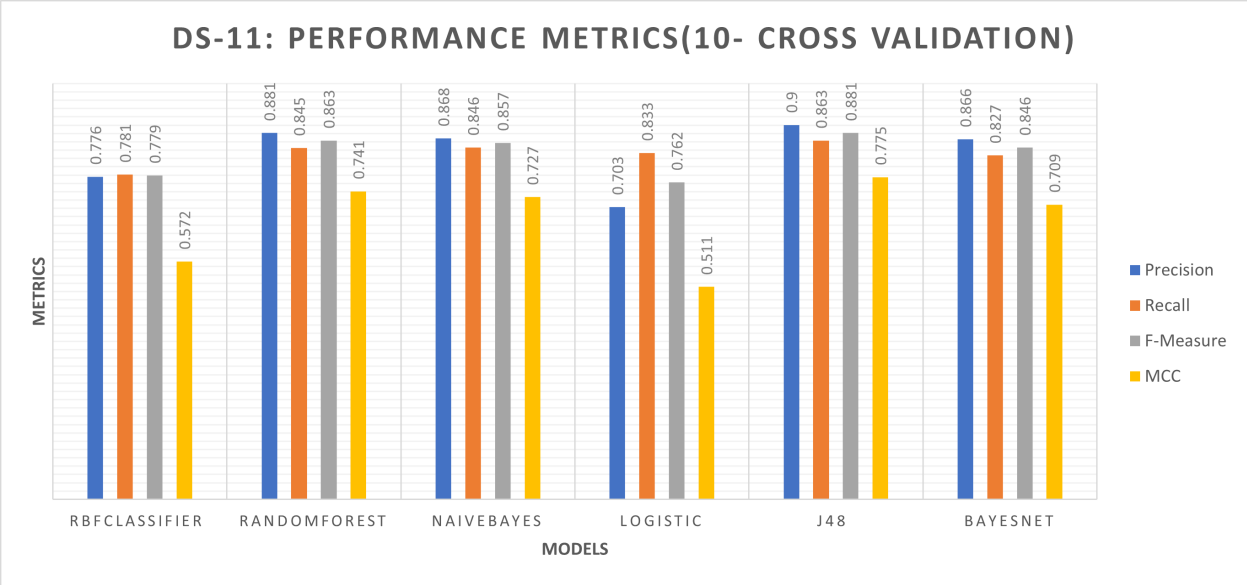


Figure 5.11: DS-11 Evaluation metrics trends with Cross Validation mode

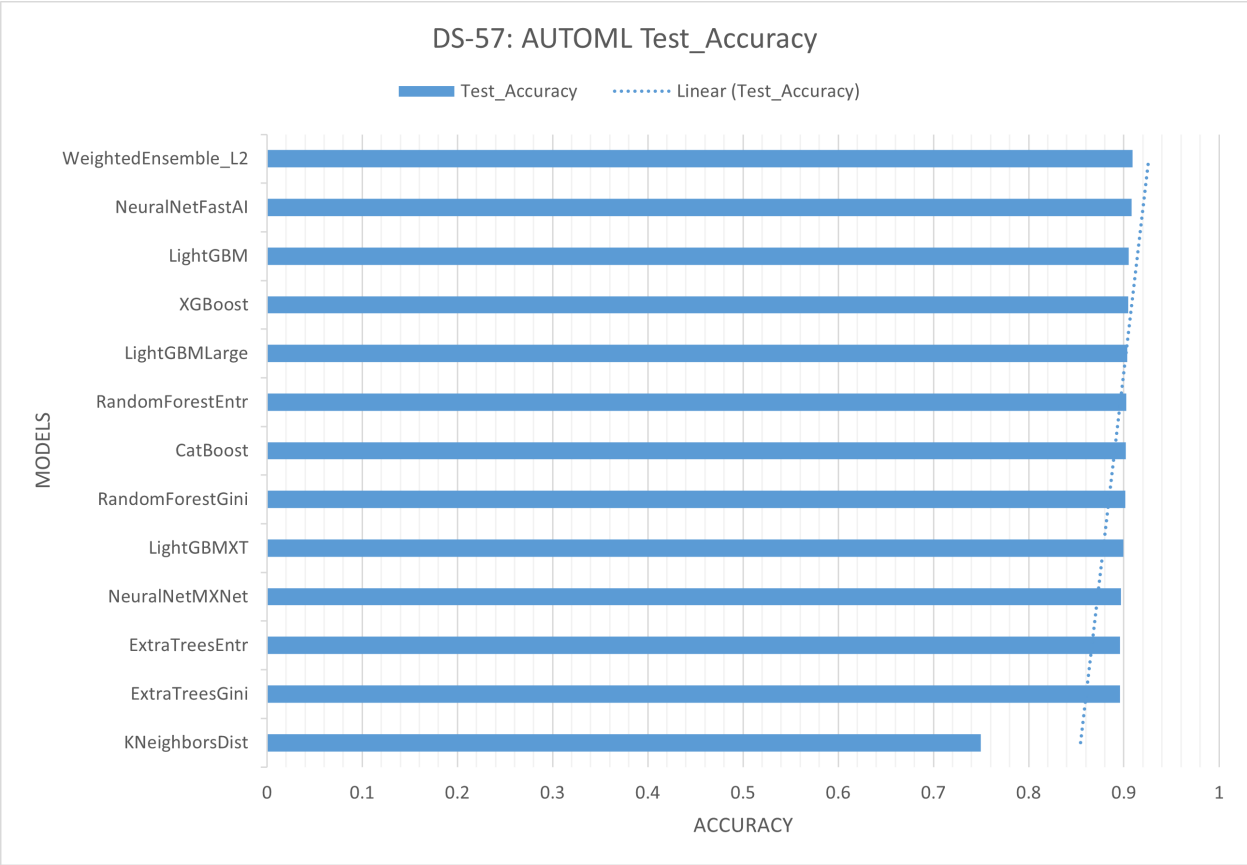


Figure 5.12: AutoGluon best performing model accuracies and trends on DS-57

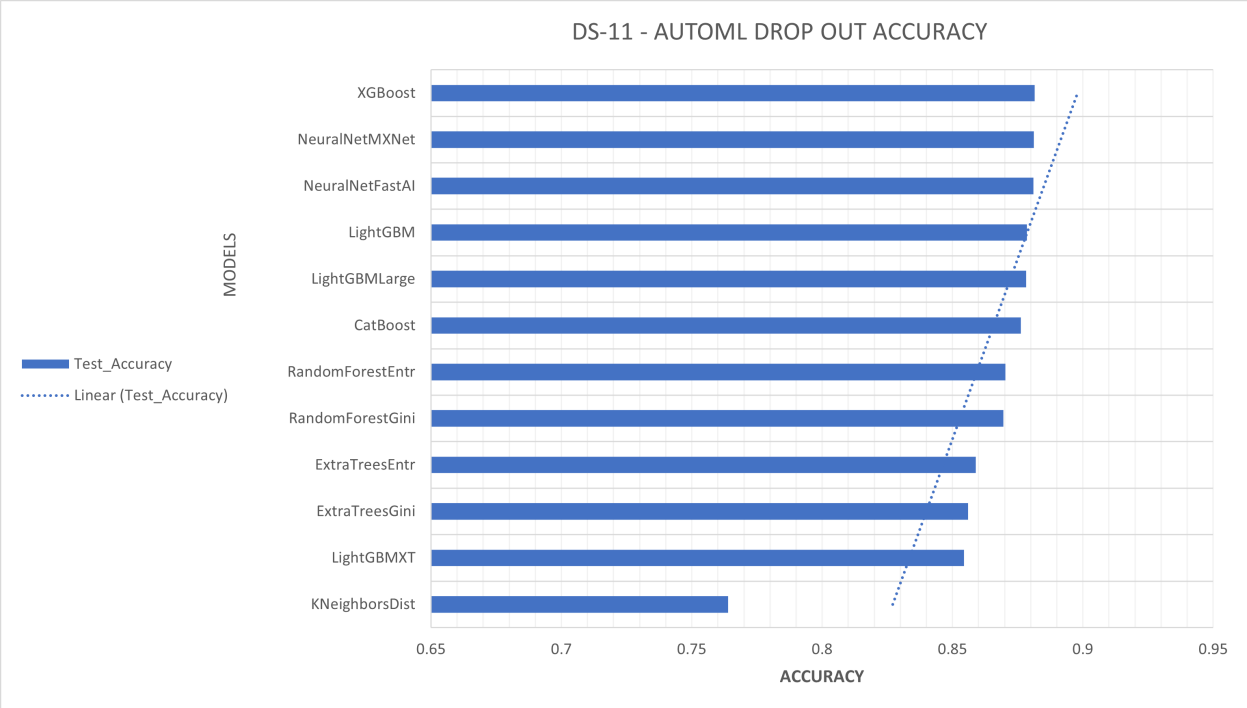


Figure 5.13: AutoGluon best performing model accuracies and trends on DS-11

Chapter 6

Conclusion

With a dramatic increase in student dropout in America, the mounting debt of student loans, and increasing unemployment due to a paradigm shift in automation, there is an urgent need for fixing the crisis of student dropout before it escalates into a humanitarian crisis. Current studies related to student dropout are done using empirical research, theoretical approach, or socio-economic approach based on racial/ethnic backgrounds. In this research, we used machine learning, data, and artificial intelligence to build models that perform student dropout prediction. We have built these models in different packages- WEKA, Conda Framework, and AutoGluon in the high-performance computing platform. We have presented several models with good performance and these models developed in a wide variety of computing platforms, have the potential to easily fit the budget of any IT department in any college.

Instead of treating the student dropout crisis as a natural process and logical consequence of social stratification resulting from socio-economic factors, we built a student dropout predictive tool that has a high potential of working outside the confines of a lab environment. This tool can further be polished and vigorously tested before integrating into the student services of the university, and report findings to the university governance team. This can shift focus from searching for answers to the question – "what socio-economic or racial background affected student dropout?" to problem-solving approach- "We have now identified students who can dropout in our college – what can we do about it, with all our resources and strengths? ". We intend this tool to find answers to the latter question. We have faith in the academic community. We believe they will innovate novel methods to target these students with tailored solutions like personal counseling services, additional tutoring classes in weak subjects, or personalized financial support to convert the dropout risk students to graduate.

While this tool can only work as a generalized university-wide prediction model, it can predict dropout outcomes for a university-wide student population with 85-90% accuracy. It is well-known that college administration is interested in underlying factors that contributed to dropout. Due to the heterogeneous nature of data, we could not address this research problem. For future work, more homogeneous data can be built, such as student data with only the chemistry department and predict dropout in the chemistry department. Research can further explore which chemistry courses are causing students to drop out and in which semester students are at high risk of dropout. Details like these can be explored by diving

deep into a specific group of homogeneous data of the student population (for example- The chemistry department). Instead of a university-wide parent model, several specific child models can be built for various departments. These models can later be used as an ensemble voting model and when combined will form a new suite of analytic tools designed to assist the university governance team to proactively monitor student graduation health.

We are confident that, in time with improved identification of dropout students, university administration will plan and implement prevention strategies. Over time, it should help learn strategies that work and strategies that will not work. This will create a new knowledge base that can be shared among other universities. In the future, as more strategies are discovered and applied to fix student dropout, gradually, we should see positive results. We should see dropout population decreasing. This phenomenon will lead to data drift. The current models will no longer serve future contexts. With a decreasing student dropout population, problem definition will slowly migrate from binary classification to anomaly detection. New raw data collected from data- drift will have fewer samples of student dropout. In this scenario, student dropout is not a crisis but an anomaly. Hence, classification algorithms will not be suitable anymore. We should shift the goalpost from classification to anomaly detection, thus a new chapter will begin to focus on state-of-the-art anomaly detection machine learning models.

Appendix A

Summary of results

S.no	Model Name	Dataset	Framework	Mode	F-score	Accuracy	Labels
1	Random Forest	DS-101	Weka	Split-80%train, 10% test	0.622	68.56	4
2	Random Forest	DS-101	Weka	Cross-Validation is 10	0.622	68.6173	4
4							
5	Random Forest	DS-57	Weka	Split-80%train, 10% test	0.884	89.2123	2
6	Random Forest	DS-57	Weka	Cross-Validation is 10	0.881	88.7724	2
7	WeightedEnsemble_L2	DS-57	AutoGluon	Split-80% train, 20%test	0.903	90.8	2
9							
10	Random Forest	DS-11	Weka	Split-80%train, 10% test	0.864	87.1361	2
11	Random Forest	DS-11	Weka	Cross-Validation is 10	0.863	87.0473	2
12	WeightedEnsemble_L2	DS-11	AutoGloun	Split-80% train, 20%test	0.8842	88.4	2
14	CatBoost	DS-57	AutoGloun	Split-80% train, 20%test	NA	90.22%	2
15	RandomForestEntr	DS-57	AutoGloun	Split-80% train, 20%test	NA	90.28%	2
16	LightGBMLarge	DS-57	AutoGloun	Split-80% train, 20%test	NA	90.38%	2
17	XGBoost	DS-57	AutoGloun	Split-80% train, 20%test	NA	90.48%	2
18	LightGBM	DS-57	AutoGloun	Split-80% train, 20%test	NA	90.50%	2
19	NeuralNetFastAI	DS-57	AutoGloun	Split-80% train, 20%test	NA	90.79%	2

Figure A.1: Summary of Results

Appendix B

Class Distributions of Datasets

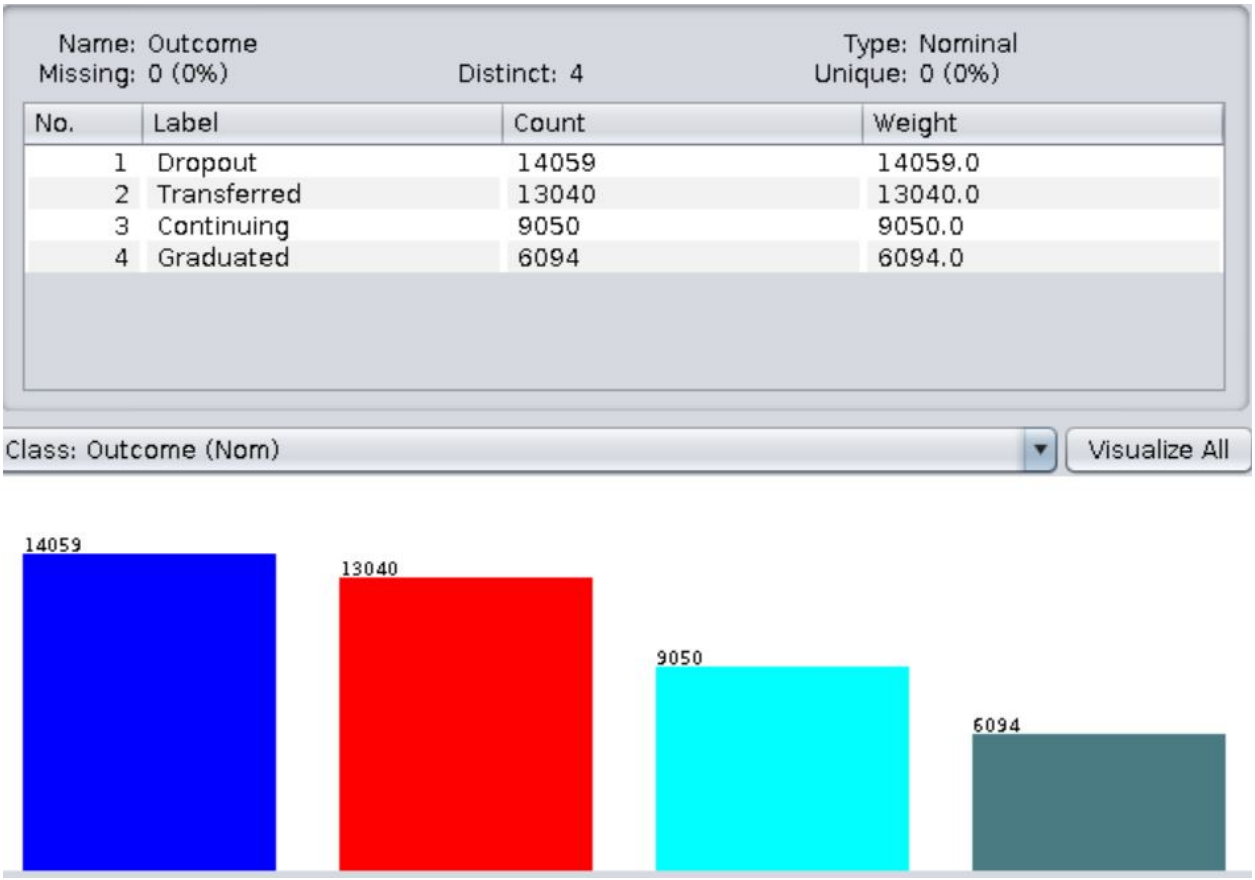


Figure B.1: Class Label Distribution in DS-101

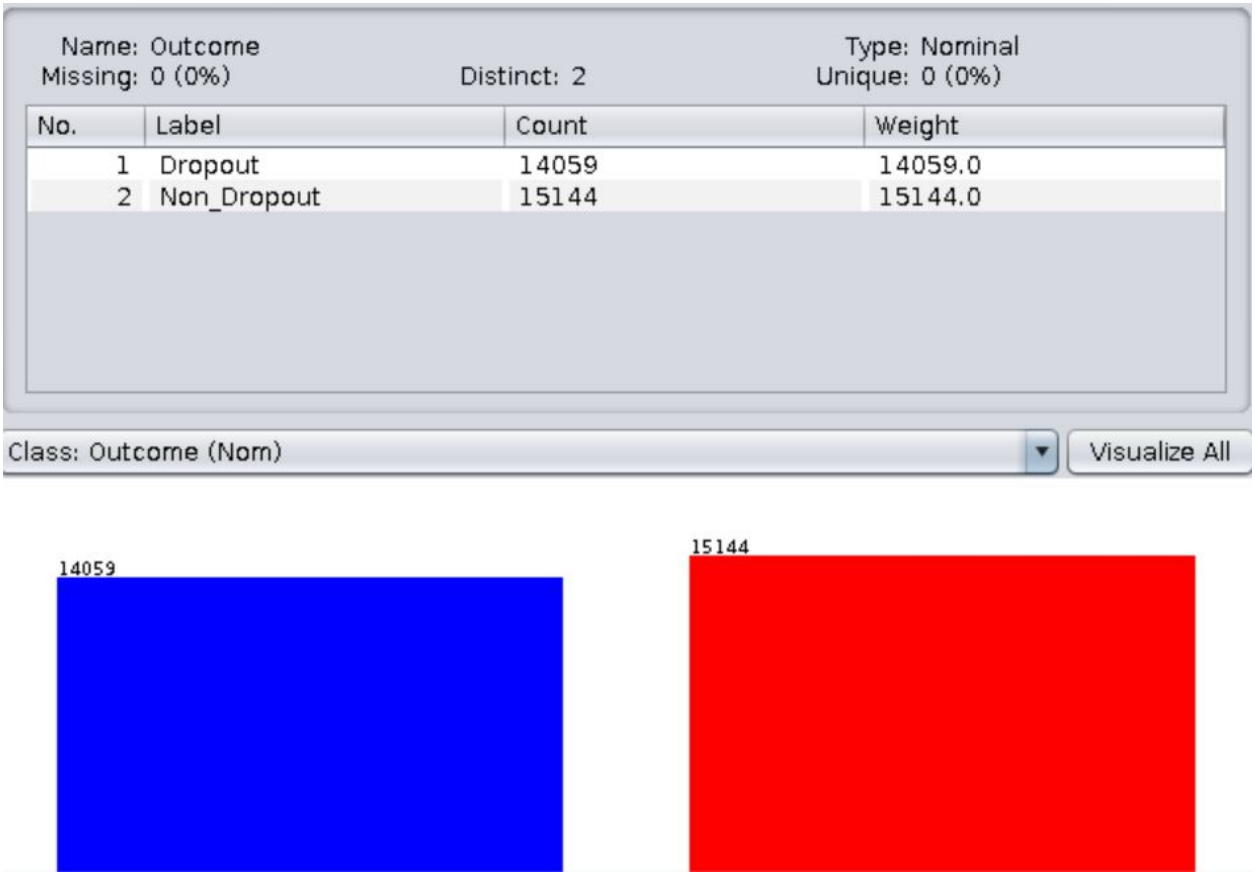


Figure B.2: Class Label Distribution in DS-11 and DS-57

Appendix C

Detailed view of DS-ii

OVERALL_LGPA_HOURS_EARNED	Total combined earned hours
OVERALL_LGPA_HOURS_ATTEMPTED	Total combined attempted hours
Tuition_Fee_all	Total Tution fee
INST_LGPA_GPA_HOURS	Total institutional hours
INST_LGPA_HOURS_PASSED	Total institutional hours passed
COMM	Indicate the "Compass Algebra" grade
INST_LGPA_GPA	Total institutional GPA
OVERALL_LGPA_QUALITY_POINTS	Total combined quality points
INST_LGPA_QUALITY_POINTS	Total institutional quality points
Grants_all	Total grant amounts
OVERALL_LGPA_GPA	Total combined GPA

Figure C.1: List of Attributes in DS-ii and descriptions

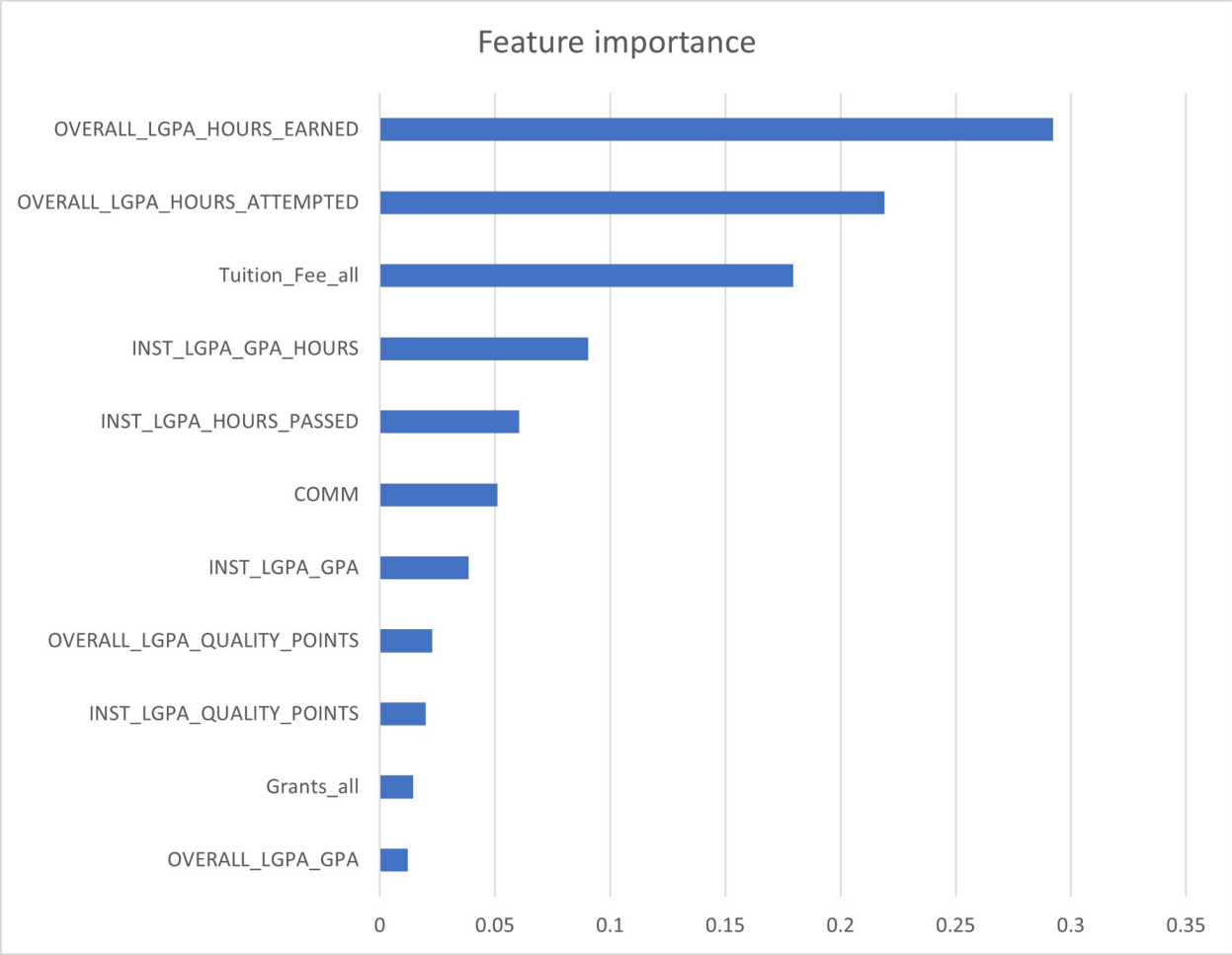


Figure C.2: Feature Importance map of DS-11

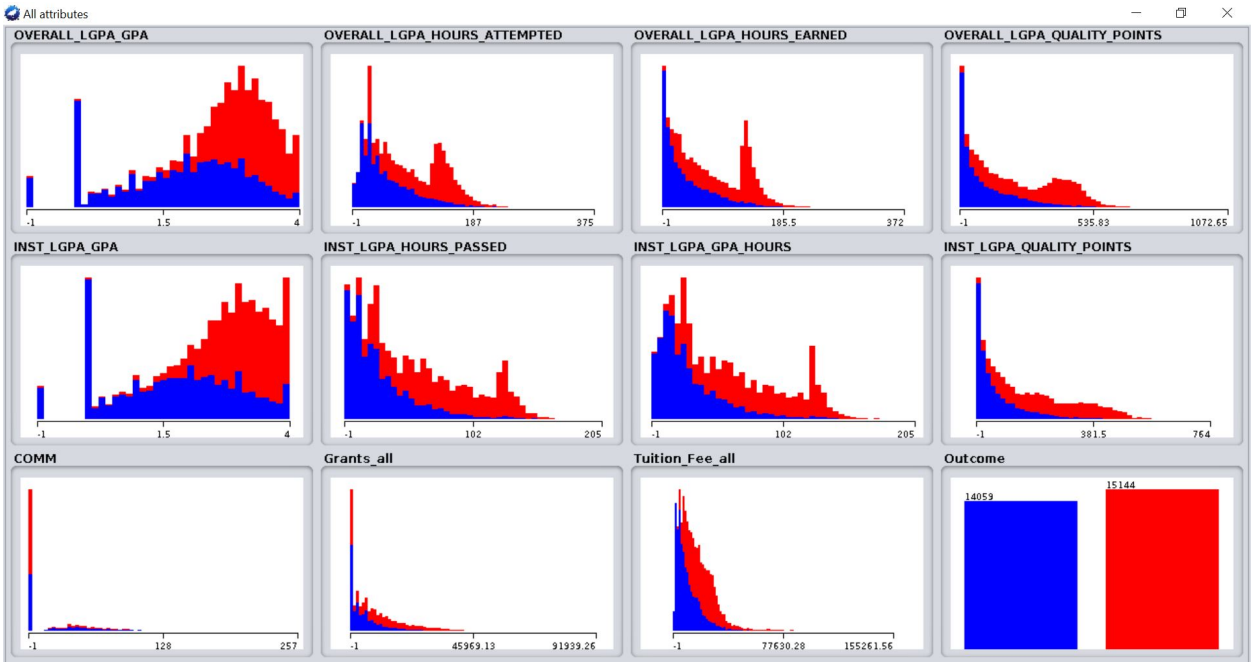


Figure C.3: Attribute Distribution in DS-11

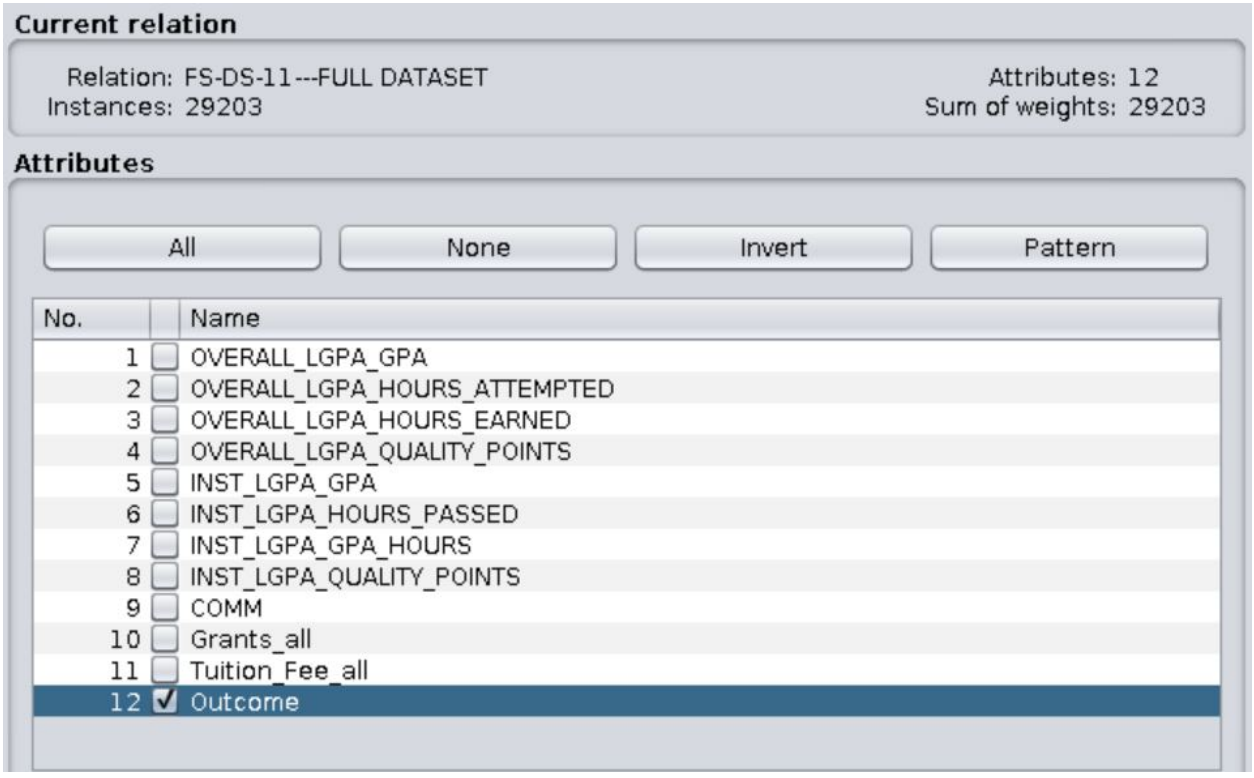


Figure C.4: List of Attributes in DS-11 and additional stats as seen in WEKA

Appendix D

Paired T-test

Random forest is the baseline model compared to remaining models in this Paired T -test. The "*" symbol next to the remaining models in the image below indicates that they are significantly weaker than Random Forest.

```
Tester:     weka.experiment.PairedCorrectedTTester -G 4 -D 1 -R 2 -S 0.05 -result-matrix "weka.experiment.ResultMatrixPlainText -mean-prec 2 -stddev-prec 2
Analysing:  F_measure
Datasets:   1
Resultsets: 6
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       7/11/21 1:30 AM


Dataset                   (1) trees.R | (2) tree (3) func (4) baye (5) baye (6) func
----------------------------------------------------------------------------------
ST_data_train           (100)   0.88 |   0.86 *   0.86 *   0.78 *   0.76 *   0.84 *
----------------------------------------------------------------------------------
                             (v/ /*) |  (0/0/1)  (0/0/1)  (0/0/1)  (0/0/1)  (0/0/1)


Key:
(1) trees.RandomForest
(2) trees.J48
(3) functions.Logistic
(4) bayes.BayesNet
(5) bayes.NaiveBayes
(6) functions.RBFClassifier
```

Figure D.1: Paired T test Results on F-Measure with 10 fold Cross validation and 10 repetitions

```
Tester:     weka.experiment.PairedCorrectedTTester -G 4 -D 1 -R 2 -S 0.05 -result-matrix "weka.experiment.ResultMatrixPlainText -mean-prec 2 -stddev-prec 2 -
Analysing:  Percent_correct
Datasets:   1
Resultsets: 6
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       7/11/21 1:29 AM


Dataset                   (1) trees.Ra | (2) trees (3) funct (4) bayes (5) bayes (6) funct
-----------------------------------------------------------------------------------------
ST_data_train            (100)   88.81 |   86.81 *   86.41 *   78.62 *   74.95 *   85.38 *
-----------------------------------------------------------------------------------------
                               (v/ /*) |   (0/0/1)   (0/0/1)   (0/0/1)   (0/0/1)   (0/0/1)


Key:
(1) trees.RandomForest
(2) trees.J48
(3) functions.Logistic
(4) bayes.BayesNet
(5) bayes.NaiveBayes
(6) functions.RBFClassifier
```

Figure D.2: Paired T test Results on Percent Accuracy with 10 fold Cross validation and 10 repetitions

Appendix E

ROC Curves

The AUC-ROC curve for Random Forest is more compared to remaining models. The Higher the AUC, the better the model is at distinguishing between Dropout and non-dropout students.

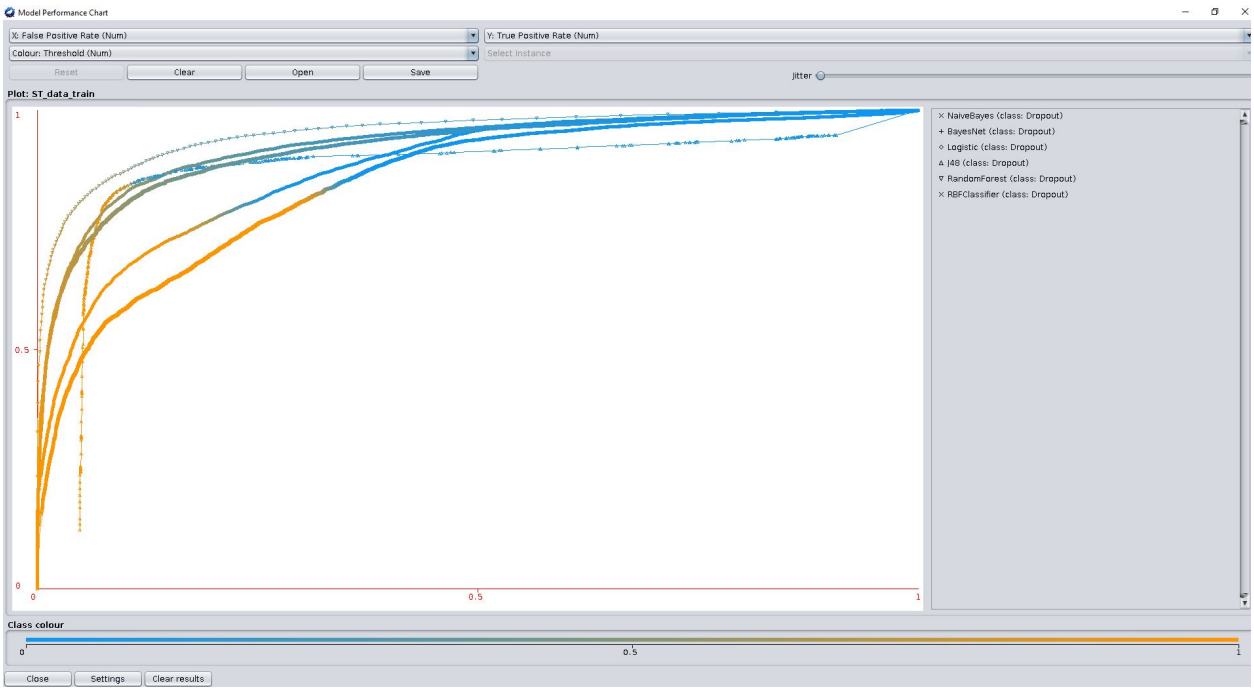


Figure E.1: ROC curve for selected models on DS-57

Bibliography

Anaconda software distribution. (2020). https://docs.anaconda.com/

AnAj. (2006). Wikimedia commons. https://commons.wikimedia.org/wiki/File:SimpleBayesNet.svg

Arcgis pro. (2020). https://www.esri.com/en-us/arcgis/products/arcgis-pro/overview

Aulck, L., Velagapudi, N., Blumenstock, J., & West, J. (2017). Predicting student dropout in higher education.

Barnard, G. A. (1989). The history of statistics: The measurement of uncertainty before 1900 (stephen m. stigler). *SIAM Review*, *31*(3), 499–502. https://doi.org/10.1137/1031103

Beaulac, C., & Rosenthal, J. (2019). Predicting university students' academic success and choice of major using random forests. *Research in Higher Education*, *60*. https://doi.org/10.1007/s11162-019-09546-y

Breiman, L. (2001). Random forests. *Mach. Learn.*, *45*(1), 5–32. https://doi.org/10.1023/A:1010933404324

Chen, T., & Guestrin, C. (2016). Xgboost. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. https://doi.org/10.1145/2939672.2939785

Christenson, S., Sinclair, M., Lehr, C., & Hurley, C. (2000). Promoting successful school completion: Strategies and programs that work. In K. Minke & G. Bear (Eds.), *Preventing school problems-promoting school success* (pp. 377–420). National Association of School Psychologists.

Cox, D. R. (1958). The regression analysis of binary sequences. *Journal of the Royal Statistical Society: Series B (Methodological)*, *20*(2), 215–232.

Dave Snowden, C. K. (2003). https://www.linkedin.com/pulse/complexity-why-your-software-project-needs-scrum-christiaan-verwijs/

d Stacey, R. (1996). https://blog.contact-software.com/en/2020/12/kompliziert-vs-komplex-der-faktor-mensch-im-projektmanagement/

Erickson, N., Mueller, J., Shirkov, A., Zhang, H., Larroy, P., Li, M., & Smola, A. (2020). Autogluon-tabular: Robust and accurate automl for structured data.

Frank, E., Hall, M. A., Holmes, G., Kirkby, R., Pfahringer, B., & Witten, I. H. (2005). Weka: A machine learning workbench for data mining. In O. Maimon & L. Rokach (Eds.), *Data mining and knowledge discovery handbook: A complete guide for practitioners and researchers* (pp. 1305–1314). Springer. http://researchcommons.waikato.ac.nz/handle/10289/1497

Friedman, N., Geiger, D., & Goldszmidt, M. (1997). Bayesian network classifiers. *Machine Learning*, *29*, 131–163. https://doi.org/10.1023/A:1007465528199

Hanson, M. (2021). Educationdata.org. https://educationdata.org/wp-content/uploads/327/college-dropouts-by-age-at-enrollment.webp

Heublein, U. (2014). Student drop-out from german higher education institutions. *European Journal of Education*, *49*(4), 497–513. https://www.jstor.org/stable/26609238

Heublein, U. (2021). Educationdata.org. https://www.jstor.org/stable/26609238

Ho, T. K. (1995). Random decision forests. *1*, 278–282 vol.1. https://doi.org/10.1109/ICDAR.1995.598994

Hoare, J. (2021). https://www.displayr.com/what-is-a-decision-tree/

Jagannath, V. (2017). Wikimedia commons. https://en.wikipedia.org/wiki/Random_forest#/media/File:Random_forest_diagram_complete.png

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T.-Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 3149–3157.

Kemper, L., Vorhoff, G., & Wigger, B. (2020). Predicting student dropout: A machine learning approach. *European Journal of Higher Education*, *10*, 1–20. https://doi.org/10.1080/21568235.2020.1718520

LightGBM. (2021). Wikimedia commons. https://lightgbm.readthedocs.io/en/latest/Features.html#leaf-wise-best-first-tree-growth

Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., & Gulin, A. (2018). Catboost: Unbiased boosting with categorical features. *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, 6639–6649.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, *1*(1), 81–106. https://doi.org/10.1007/BF00116251

Quinlan, J. R. (1993). *C4.5: Programs for machine learning*. Morgan Kaufmann Publishers Inc.

Snowden, D., & Boone, M. (2003). A leader's framework for decision making. *A leader's framework for decision making* (pp. 69–76). Harvard Business Review.

Snowden, D., & Kurtz, C. (2003). The new dynamics of strategy: Sense-making in a complex and complicated world. *The new dynamics of strategy: Sense-making in a complex and complicated world*. IBM Systems Journal Vol 42 No 3.

Stacey, R. (1996). Complexity and creativity in organizations. *Complexity and creativity in organizations*. Berrett Koehler Publishers.

Stone, J. V. (2013). *Bayes' rule: A tutorial introduction to bayesian analysis*. Sebtel Press.

Vikramkumar, B, V., & Trilochan. (2014). Bayes and naive bayes classifier.

Whitehill, J., Mohan, K., Seaton, D., Rosen, Y., & Tingley, D. (2017). Delving deeper into mooc student dropout prediction.