LEARNING STYLE PREDICTION IN E-LEARNING PLATFORMS

A Mini Project Report Submitted in partial fulfilment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING

Submitted by

Mr. Akarsh Chagantipati (18071A0507)
Mr. Vishnu Vardhan Kothagadi(18071A0526)
Mr. Dinesh Pandikona(18071A0543)
Mr. Sai Sudheer Reddiboina(18071A0546)

Under the Guidance of

Mrs. A. Madhavi
(Assistant Professor, Dept of CSE)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI INSTITUTE OF ENGINEERING AND TECHNOLOGY

An Autonomous Institute, NAAC Accredited with 'A++' Grade
NBA Accredited for CE, EEE, ME, ECE, CSE, EIE, IT B. Tech Courses
Approved by AICTE, New Delhi, Affiliated to JNTUH
Recognized as "College with Potential for Excellence" by UGC
ISO 9001:2015 Certified, QS I GUAGE Diamond Rated

Vignana Jyothi Nagar, Pragathi Nagar, Nizampet (S.O), Hyderabad – 500 090, TS, India

2021

VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI INSTITUTE OF ENGINEERING AND TECHNOLOGY

An Autonomous Institute, NAAC Accredited with 'A++' Grade
NBA Accredited for CE, EEE, ME, ECE, CSE, EIE, IT B. Tech Courses
Approved by AICTE, New Delhi, Affiliated to JNTUH
Recognized as "College with Potential for Excellence" by UGC
ISO 9001:2015 Certified, QS I GUAGE Diamond Rated

Vignana Jyothi Nagar, Pragathi Nagar, Nizampet (S.O), Hyderabad – 500 090, TS, India

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project report entitled "**Learning Style Prediction in E-Learning Platforms**" is a Bonafede work done under our supervision and is being submitted by **Mr. Akarsh Chagantipati (18071A0507), Mr. Vishnu Vardhan Reddy Kothagadi(18071A0526), Mr. Dinesh Pandikona(18071A0543), Mr. Sai Sudheer Reddiboina(18071A0546)** in partialfulfilment for the award of the degree of **Bachelor of Technology** in Computer Science and Engineering, of the VNRVJIET, Hyderabad during the academic year 2020-2021. Certified further that to the best of our knowledge the work presented in this thesis has not been submitted to any other University or Institute for the award of any Degree or Diploma.

Mrs. A. Madhavi
Assistant Professor & internal guide
Department of CSE
VNR VJIET

Dr.S.Nagini
Professor & HOD
Department of CSE
VNR VJIET

VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI INSTITUTE OF ENGINEERING and TECHNOLOGY

An Autonomous Institute, NAAC Accredited with 'A++' Grade
NBA Accredited for CE, EEE, ME, ECE, CSE, EIE, IT B. Tech Courses
Approved by AICTE, New Delhi, Affiliated to JNTUH
Recognized as "College with Potential for Excellence" by UGC
ISO 9001:2015 Certified, QS I GUAGE Diamond Rated

Vignana Jyothi Nagar, Pragathi Nagar, Nizampet (S.O), Hyderabad – 500 090, TS, India

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

DECLARATION

We declare that the major project work entitled "Learning Style Prediction in E-Learning Platforms" submitted in the department of Computer Science and Engineering, Vallurupalli Nageswara Rao Vignana Jyothi Institute of Engineering and Technology, Hyderabad, in partial fulfilment of the requirement for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering** is a Bonafede record of our own work carried out under the supervision of **Mrs. A. Madhavi, Assistant Professor, Department of CSE, VNRVJIET.** Also, we declare that the matter embodied in this thesis has not been submitted by us in full or in any part thereof for the award of any degree/diploma of any other institution or university previously.

Place: Hyderabad

Akarsh Chagantipati	**Vishnu Kothagadi**	**Dinesh Pandikona**	**Sudheer Reddiboina**
(18071A0507)	(18071A0526)	(18071A0543)	(18071A0546)

ACKNOWLEDGEMENT

Firstly, we would like to express our immense gratitude towards our institution VNR Vignana Jyothi Institute of Engineering and Technology, which created a great platform toattain profound technical skills in the field of Computer Science, thereby fulfilling our most cherished goal.

We are very much thankful to our Principal**, Dr. Challa Dhanunjaya Naidu** and our Head of Department, **Dr.S.Nagini,** for extending their cooperation in doing this project in stipulated time.

We extend our heartfelt thanks to our guide**, Mrs. A. Madhavi** and the project coordinators **Dr.Ch. Suresh** and **Nayamtulla Khan,** for their enthusiastic guidance throughout the course of our project.

Last but not least, our appreciable obligation also goes to all the staff members of Computer Science & Engineering department and to our fellow classmates who directly or indirectly helped us.

Mr. Akarsh Chagantipati (18071A0507)
Mr.Vishnu Vardhan Kothagadi(18071A0526)
Mr. Dinesh Pandikona (18071A0543)
Mr. Sudheer Reddiboina (18071A0546)

ABSTRACT

This project aims to develop a machine learning model using SkLearn and other machine learning tools to understand the learning behavior in students using online learning platforms. Generally, most of the students tend to have a single dominant type of intelligence for learning new things, classifying them into intelligence categories help students to understand about themselves and improve accordingly.

Learning management systems are widely used in educational organizations and universities to deliver self-paced online courses. Furthermore, educational theories have suggested that providing learners with learning material suitable for their learning styles may affect their learning performance.

Human brains generally use different methods for grasping knowledge faster and easier. We call these methods as ***learning styles***. Learners with different individual traits, levels of knowledge, backgrounds, and characteristics have different learning styles. Determining student's learning style improves the efficiency of the learning process.

A model for accurate automated learning style recognition is needed to provide personalized learning content to the student based on his/her learning style. In this work we implement an intelligent model for accurate detection of student's learning styles.

INDEX

CHAPTER 4

List of Figures

1. INTRODUCTION

There is a myriad of online learning platforms in today's date. They use a variety of methods to deliver the teaching content to the students, ranging from video lectures to posting assignments to assess the understandability of a student. As per statistics, about 33 percent of college students are taking at least one course online, according to a survey by the Babson Survey Research Group. Schools and employers are recognizing that online courses can be just as effective — and sometimes more effective — than classroom courses. But are the courses really being helpful to all the students?

Unlike a traditional classroom where a lecturer can assess one's capabilities to comprehend a topic and provide the necessary help accordingly, online platforms are completely dependent on self-analysis. Hence most of the students find it difficult to understand the lectures on the online platforms. You may have heard of the idea that we all respond best to different styles of learning. That is exactly what the seven learning styles theory supports. All the styles capture an individual strength that likely helps a person retain information more effectively. They each focus on one of the five senses or involve a social aspect. This theory is popular because, by finding an individual learner's style and tailoring teaching to it, it was thought their efficiency could be improved. The 7 styles of the theory are:

- visual
- kinesthetic
- aural
- social
- solitary
- verbal
- logical

It So, to address the issue, a tool that can predict the learning style of an individual and provide the appropriate content based on the predictions would enhance the learning process of the student. A 2012 study revealed that 93% of teachers in the UK agree that students learn better when they receive information in their preferred learning style.

The learning styles derived from Howard Gardner's 1960s theory of Multiple Intelligences. This theory states that: "we are all able to know the world through language, logical-mathematical analysis, spatial representation, musical thinking, the use of the body to solve problems or to make things, an understanding of other individuals, and an understanding of ourselves." A model for accurate automated learning style recognition is needed to provide personalized learning content to the student based on his/her learning style. In this work we implement an intelligent model for accurate detection of student's learning styles. The integration of user profiles with learning style data could enhance intelligent learning systems by providing a learner with personalized learning behavior guidance that is appropriate to the needs.

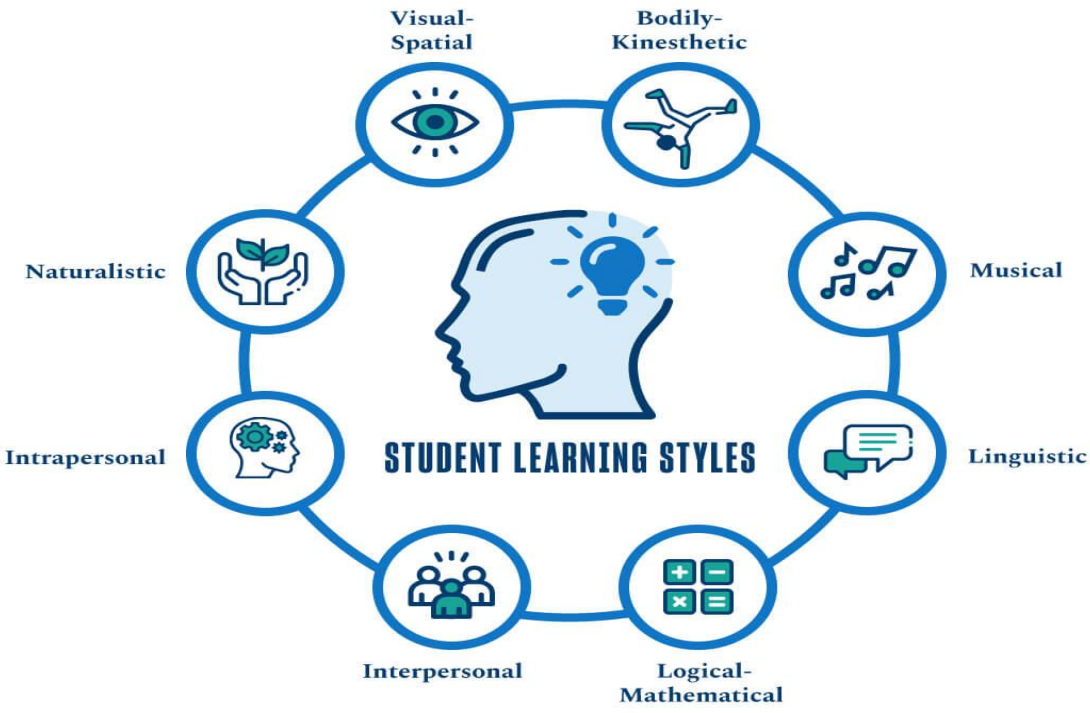

Fig 1.1: General types of intelligences based on theories.

Howard Gardner (2013) asserts that regardless of which subject you teach— "the arts, the sciences, history, or math"—you should present learning materials in multiple ways. Gardner goes on to point out that anything you are deeply familiar with "you can describe and convey in several ways. We teachers discover that sometimes our own mastery of a topic is tenuous, when a student asks us to convey the knowledge in another way and we are stumped." Thus, conveying information in multiple ways not only helps students learn the material, it also helps educators increase and reinforce our mastery of the content.

The Gardner's multiple intelligences theory can be used for curriculum development, planning instruction, selection of course activities, and related assessment strategies. Gardner points out that everyone has strengths and weaknesses in various intelligences, which is why educators should decide how best to present course material given the subject-matter and individual class of students. Indeed, instruction designed to help students learn material in multiple ways can trigger their confidence to develop areas in which they are not as strong. In the end, students' learning is enhanced when instruction includes a range of meaningful and appropriate methods, activities, and assessments.

While Gardner's MI have been conflated with "learning styles," Gardner himself denies that they are one in the same. The problem Gardner has expressed with the idea of "learning styles" is that the concept is ill defined and there "is not persuasive evidence that the learning style analysis produces more effective outcomes than a 'one size fits all approach'" (as cited in Strauss, 2013). As former Assistant Director of Vanderbilt University's Center for Teaching Nancy Chick (n.d.) pointed out, "Despite the popularity of learning styles and inventories such as the VARK, it's important to know that there is no evidence to support the idea that matching activities to one's learning style improves learning." One tip Gardner offers educators is to "pluralize your teaching," in other words to teach in multiple ways to help students learn, to "convey what it means to understand something well," and to demonstrate your own understanding. He also recommends we "drop the term 'styles.' It will confuse others and it won't help either you or your students" (as cited in Strauss, 2013).

The key objectives for the project were identified as:

- Identify different ways to obtain data for the experiment.
- Develop a system that can understand student activity and identify their study needs.
- Compare different algorithms in predicting the learning style of a student.
- Develop a self-improving system based on student activity and questionnaire results.

Problem definition:

Educators have investigated the problem of students learning through online platforms and found that the learning materials recommended by these systems do not satisfy the requests of learners sufficiently and they fail to meet the potential needs of learners in some cases. Problems known as “learning deviation” and “cognitive overload” are of great importance, and it has been proposed that most learners are unsure of their actual needs during the learning process, which may lead to inaccurate requests. To address these problems, it might be beneficial if intelligent learning systems were designed to analyze the actual needs of learners, thereby helping to improve their learning performance.

Challenges:

- Collecting un-biased data of real students interacting with learning management system
- Understanding edge cases where students possess multiple cohesive intelligences based on different factors including environment, upbringing, etc.

2. LITERATURE SURVEY/ EXISTING SYSTEM

2.1 FEASIBILITY STUDY

In a recent study conducted by researchers on university students, participants performed visual activities of varying characteristics, and the eye-tracking analysis revealed quantitative differences on visual behavior among individuals with different cognitive styles. Based on these differences, a series of classification experiments were conducted, and the results revealed that gaze-based implicit elicitation of cognitive styles in real-time is feasible, which could be used by interactive systems to adapt to the users' cognitive needs and preferences, to better assist them, and improve their performance and experience. Hence this proves the feasibility of our model.

2.1.1 ORGANIZATIONAL FEASIBILITY

The feasibility of amount of preparation of school students for the complete and fair implementation of the learning style prediction will be done. The results of the feasibility of implementing electronic learning to the educational system managers will have the opportunity to evaluate the needs of schools, educational environment for the implementation of electronic learning system gives them. Students are the most important people in the use of electronic learning environments because by having e-learning standard content, with a new style of learning will be familiar.

2.1.2 ECONOMIC FEASIBILITY

The Given the fact that our tool will enhance the learning experience of students on the platform and that will in turn make the online platform more effective, the tool will be funded and affordable for the platform. For the seamless functioning of the tool, regular inspections must be performed with the help of system and data engineers. However, since the application is mostly automated, there wouldn’t be any extra requirement of human sourcing, making it economically feasible like any other web-application.

2.1.3 TECHNICAL FEASIBILITY

The software Anaconda, Spyder, Python are needed to be installed in the system to build this application along with Blender software to design the models. A minimum of 8GB ram is required to run all this software smoothly. Anaconda support package needs to be installed into the computer from marketplace hub which helps the editor to import required packages and take necessary steps in the background to build the structure of the application. Spyder is the software that is helpful to edit and use the necessary packages like TensorFlow, Pandas, etc., which are important to build the application in the background. Thus, this application is technically feasible.

2.1.4 BEHAVIOURAL FEASIBILITY

The tool is behaviorally feasible because it does not require any technical guidance or answers to any kind of questionnaire or surveys. Since it is going to enhance the learning experience to a student on any online platform, the users will react to this tool very positively. Moreover, it is a user-friendly application, as in it is totally customized for an individual student depending on their inherent abilities to learn, making it a behavioral feasible one.

2.2 LITERATURE REVIEW

Before discussing the methods used to identify the learning style, a quick account of what is learning style, thinking style, learning patterns, characteristics of learners with a particular learning style, whether they are malleable or not, and learning styles in online learning is needed. In addition to the learning style models discussed in the previous section, there are a few researchers in the area whose work is worth discussing. Among them is Jan D. Vermunt, who proposed an integrated approach to learning style. The final version of the questionnaire he designed consisted of 120 questions that cover the following learning components: cognitive processing strategies, metacognitive regulation strategies, conceptions of learning, and learning orientations (Vermunt & Vermetten, 2004). There are various scales and values in the questionnaire, in the first component, we have deep processing and stepwise processing, and in turn, each of them has different extremes. The regulation strategies have two scales self-regulation and external regulation with their values. The questionnaire helps us gain a complete insight into a

learner's mind; it has high validity and reliability and has been widely used. However, a questionnaire of this kind, and depth, seems difficult to understand in an online learning context.

Few of its concepts, such as external regulation and learning orientations even when monitored in online learning, are challenging to include in online learning, and it would use up resources. Extensive research must be done to see how this can be incorporated into online learning. Learning styles and their malleability has been researched and discussed by Zhang (2011), where the researcher has discussed how learning style is affected by culture. Cognitive styles are also malleable, as proven by Angeli and group, using an experiment with children aged 6 to 7 years (Angeli, Valanides, Polemitou, & Fraggoulidou, 2016). Researchers have been using different methods and have based their research on different learning style models. Some of the approaches to identify learning style and the frequency of usage of learning style models by various researchers are summarized in this section. Researchers have results that prove the link between learning style and academic performance (Lynch, Woelfl, Steele, & Hanssen, 1998; Komarraju, Karau, Schmeck, & Avdic, 2011; Cassidy & Eachus, 2000). Researchers have proposed many artificial intelligence techniques to detect the learning style of the learners automatically. There are two different approaches defined by Bernard, Chang, Popescu, and Graf (2017) literature-based and data-driven approach for automatic detection of learning style. Data-driven strategies aim at building classifiers based on the data. Literature-based approaches use the user model to get hints from it and generate simple if-then rules to detect the learning style. The evidence of researchers using literature-based methods can be found in (Carver, Howard, & Lane, 1999; Dung, & Florea, 2012; Graf, Liu, & Kinshuk., 2010; Sabine Graf, 2006; Latham, Crockett, McLean, & Edmonds, 2012; Popescu, 2009; and Sangineto, Capuano, Gaeta, & Micarelli, 2008). The researchers using data-driven approaches have used different classification algorithms such as Bayesian networks, Decision Trees, Neural Networks, and other models. All these methods extracted some attributes from the user behaviour and built classifiers using the extracted data.

Alkhuraiji, Cheetham, and Bamasak (2011), Carmona, Castillo, and Mill´an (2008), and García, Schiaffino, and Amandi (2008), García, Amandi, Schiaffino, and Campo (2007) are the researchers that have built Bayesian networks to detect learning styles. Decision Tree is a method that works in two phases, first building the tree and then pruning the tree. A decision tree was used by the following researchers ¨Ozpolat and Akar (2009), Crockett, Latham, and Whitton (2017), and Cha et al. (2006). The neural network is a machine learning technique that is based on the concept of neurons and the learning capability of the human brain. Neural networks have been used by the following researchers to recognize learning style: Kolekar, Sanjeevi, and Bormane (2010), Villaverde, Godoy, and Amandi (2006), Georgiou and Makry (2004), and others. Some of the recent work which has been done in this area is summarized in Table below.

S. No	Title of the Paper	Methods/Approach	Pros/Cons	Year
1	Learning Style detection in E-learning Systems using Machine Learning Techniques	• Private Dataset • Felder-Silverman classification of learning styles • Models: SVMs, KNN, Naïve Bayes, Random Forest	• Dataset is not disclosed publicly • Questionnaire for improving the model is not discussed • SVMs performed best with accuracy of 75.55 %	2021
2	A hybrid Machine Learning Approach to predict learning styles in adaptive E-learning System	• Private student activity logs used as dataset • Felder-Silverman classification of learning styles • Hybrid model : K-Means + Naïve Bayes	•Private Dataset •Based on assumption that learning style doesn't change often •The hybrid model performed well with 83% accuracy	2019

3	E-learning Student Reactions	• Web mined data of students of Brazilian university • Authors explored the link between the reactions and the skill levels shown by the students • Model : XgBoost Regressor	• Dataset only accessible for their university • The research had 80% accuracy in predicting student's success in examination	2019
4	Automatic Detection of Students Learning Style in Learning Management System	• Data is collected from Moodle LMS • Custom ruleset for decision tree classifier • Model: Decision Tree Classifier	• Moodle considers many student activities(chats, forums, quizzes, etc) • Custom rules to strengthen decision tree classifier • Dataset mapped with 3 learning dimensions to obtain average accuracy of 87%	2019
5	Students' learning style detection using Tree augmented Naïve Bayes	• Data collected from Moodle LMS by assessing 46 undergrad students • To get better accuracy than basic Bayesian Network • Model : Tree Augmented Naïve Bayes	•Moodle considers many student activities(chats, forums, quizzes, etc) •Tree augmentation to increase the learning capabilities of Naïve Bayes •Detecting student learning style model was around 78%	2018

6	Detecting Learning Styles with Artificial Neural Networks	• Generating Prior Knowledge • Prediction of Learning style • Making online learning personalised • Model: Latent Semantic Indexing (for prior Knowledge)+ Artificial Neural Networks	• Private student web interaction log data • Internal approach derives data from the personality of student • The results of the study were 81% accurate	2019

Fig 2.2: Literature Survey

Dr. Fareeha Rasheed, Abdul Wahid[1] came up with Learning Style detection in E-learning Systems using Machine Learning Techniques in 2021. The authors, in their previous work in (Rasheed & Wahid, 2019), proposed a neural network for the identification of learning styles based on multiple intelligences. The proposed method was not validated, and when it was validated, it generated inaccurate models. Hence, to overcome the problems that occurred from the previous research, this work has been proposed, trained, and validated. A lot of work has been done in the area of learning style detection; the below problems are predominant in the area. • Real datasets have not been used for the validation of the proposed models. • In some publications, the models have not been compared to others for accuracy, precision, and recall • Most of the papers that used the FSLM learning style model mostly skipped either one or two dimensions of the model • Most of the publications we reviewed worked with the Kolb's learning style inventory and FSLM model; a very few of them used Gardner's theory of multiple intelligences. The study initially focussed on studying learning style models, their dimensions, values, and combinations of values to identify the learner's learning styles. The study by Jegatha and group (Jegatha Deborah, Baskaran, & Kannan, 2014), helped us to understand the link between the theory of learning styles and e-learning. Other interesting studies in ¨Ozpolat & Akar (2009), Khan, Shamim, & Nambobi (2018), and Santo (2006) discussed common behaviour, learning preferences, recall and retention rates, efficiency and performance of learners with various learning styles. A study (Huang, Lin, & Huang, 2012) analysed the performance of learners when they learnt using resources according to their learning style. The study is promising as it proves that

presenting learners with customized resources to learn betters their performance. There is a study on linking multiple intelligences and online learning activities (Zhang & Bonk, 2009) helped us identify what learners with various learning styles preferred to do in an online learning system. Students' typical choice of resources, the time taken to learn each concept using their choice, their performance in assessments and other learner attributes were observed. We then extracted indicators from learner's online behaviour that will help us to predict their learning style. After the extraction of learner behaviour from learner logs, we preprocessed it. The processed data was then passed as input to the machine learning classifiers. The Recognition system is based on machine learning classifiers which predict the learning style of the learner based on the inputs provided. The extraction of indicators is necessary because we want to eliminate the use of questionnaires to identify the learning style. To prove the elimination of the use of questionnaires, we evaluate the consistency of questionnaire-based identification and the predicted learning style using the proposed attributes and trained model. The predicted learning style can be used to personalize the learning resources and recommend them specific resources that cater to their learning styles so that learning is optimal. The proposed methodology is an extension of our work done in Rasheed and Wahid (2019). In the paper, we proposed attributes to recognize the type of intelligence based on the theory of Gardner's theory of multiple intelligences. We proposed a neural network architecture to recognize the intelligence, and the intelligences was linked to learning styles.

For the attributes identified for the theory of multiple intelligences, we collected a dataset with 498 samples from students taking online courses. The pre-processing was done. Then various classification algorithms such as decision trees, Support Vector Machines, K-Nearest Neighbour, Naïve Bayes, Linear Discriminant Analysis, Random Forest, and Logistic Regression were applied. We run the classifiers on the python kernel 3.1. To avoid overfitting (A problem where the trained technique predicts everything you enter into a particular class), we tuned the parameters of the classifiers using the GridSearch technique of hyperparameter tuning. The GridSearch technique helped us gain an understanding as to how the training and testing accuracy differs with automatic tuning. We then performed manual tuning of the parameters to achieve optimal results of accuracy, precision, and recall. The values of the trained parameters can be seen in Table

5. For the attributes identified for the Felder Silverman Learning style model, we collected a dataset with 498 samples from students taking online courses. The pre-processing was done, fit, and transform; the data were normalized between 0 and 1. We used the decision trees, Support Vector Machines, K-Nearest Neighbour, Naïve Bayes, Linear Discriminant Analysis, Random Forest, and Logistic Regression and to avoid overfitting; we performed tuning of the parameters. The algorithms were run on Python Kernel 3.1. Table 6 summarizes the accuracy of the various algorithms for different dimensions of the FSLM model.

Ouafae El Aissaoui1 , Yasser El Madani El Alami , Lahcen Oughdir , and Youssouf El Allioui[2] came up with A Hybrid Machine Learning Approach to Predict Learning Styles in Adaptive E-Learning System in 2019. Many approaches have been proposed to automatically detect students' learning styles based on machine learning techniques. Those literature works have been relied on various classifiers. Feldman et al. [13] found that the Bayesian network classifier is one of the most widely adopted classifiers to infer the leaning style. Garcia et al. [14] used Bayesian Networks to detect the learning style of a student in a Web-based education system. To evaluate the precision of their proposed approach, they compared the learning style detected by their approach against the learning style obtained with the index of learning style questionnaire. Bunt and Conati [15] addressed this problem by building a BN able to detect the difficulty that learners face during the exploration process; and then providing specific assessment to guide and improve the learner's exploration of the available material. Decision tree is a classification algorithm that is also frequently used in the field of automatic detection of learning styles.

Kalhoro [16] proposed an approach to automatically detect the learners' learning styles from web logs of the students using the Data Mining technique, and the Decision Trees classifier. Kolb's learning style theory was incorporated to understand e-learners' learning styles on web. Pantho and Tiantong [17] proposed an approach to classify VARK (Visual, Aural, Read/Write, Kinesthetic) learning styles of learners by using Decision Tree C4.5 algorithm. Data concerning learning styles of learners were collected via a questionnaire responded by 1205 students. The collected data were then classified using Decision Tree C4.5 algorithm. 776 O. El Aissaoui et al. Neural networks are also commonly used in the automatic detection of learning styles, Hmedna et al. [18] proposed

an approach that uses neural networks to identify and track learners learning styles in order to ensure efficient recommendation of resources. Their work was based on Felder-Silverman' dimensions. Hmedna et al. [18] introduced an automatic student modeling approach for identifying learning style in learning management systems according to FSLSM. They proposed the use of fuzzy cognitive maps FCMs as a tool for identifying learner's learning style.

FCMs are a soft computing tool which is a combination of fuzzy logic and neural network. Similarly to the previous algorithms; KNN is frequently used to detect the learning styles automatically, Chang et al. [19] proposed a learning style classification mechanism to classify and then identify students' learning styles. The proposed mechanism improves k-nearest neighbor (k-NN) classification and combines it with genetic algorithms (GA). As can be noticed, all the previous works relied on a learning style model in their approaches. Most of the proposed approaches used the FSLSM's dimensions and considered that there are 8 learning styles where each one corresponds to a dimension's category. In reality, there are sixteen learning style combinations obtained by combining one category from each dimension. Similarly, to the previous work; we have also relied on the FSLSM, but by considering sixteen LSC. 4 Methodology An adaptive E-learning system takes into account the learner's learning style and provides contents to the learners based on their preferred learning styles which are identified using the learners' sequences. To identify a learner's learning style we have to rely on a standard learning style model such as FSLSM, where the captured sequences can be labeled with a specific learning style combination using an unsupervised algorithm. In order to implement the unsupervised algorithm, the learner's sequences which have been extracted from the log file, must be transformed to the input of that algorithm. The learning sequence of a learner is defined by the various learning objects accessed by that specific learner during a session. Each sequence contains the sequence id, session id, learner id, and the set of learning objects accessed by the learner in a session. After detecting the learners' sequences; our first aim is to classify them according to FSLSM by assigning a specific learning style combination to each sequence, and our second aim is to use those labeled sequences as training set in order to predict the learning style of a new sequence.

Many definitions appear in the literature concerning the term learning style. Laschinger and Boss [1] defined learning style as the way in which individuals organize information and experiences, while Garity [6] defined it as the preferred way to learn and process information. Keefe [7] described learning styles as the cognitive, effective, and psychological behaviors that serve as relatively stable indicators of how learners perceive, interact with, and respond to the learning environment. Learning style refers to the preferred way in which a learner perceives, reacts, interacts with, and responds to the learning environment. Each learner has his own preferred ways of learning; there are some students who prefer to study in a group, A Hybrid Machine Learning Approach to Predict Learning Styles … 773 other prefer to study alone; some prefer to learn by reading written explanations, other by seeing visual representations, pictures, diagrams, and charts. In order to ensure an efficient learning process for learners; the e-learning system should consider the differences in learners' learning styles. 2.2 The Felder and Silverman Learning Style Model A learning style model classifies the learners into a specific number of predefined dimensions, where each dimension pertains to the way they receive and process information. Many learning style models have been proposed in the literature, in this work we have based on the Felder and Silverman model. According to the previous researches [4, 5], the FSLSM is the most used in adaptive e-learning systems and the most appropriate to implement them. The FSLSM presents four dimensions with two categories for each one, where each learner has a dominant preference for one category in each dimension: processing (active/reflective), perception (sensing/intuitive), input (visual/verbal), understanding (sequential/global). Active (A) learners prefer to process information by interacting directly with the learning material, while reflective (R) learners prefer to think about the learning material. Active learners also tend to study in group, while the reflective learners prefer to work individually. Sensing (Sen) learners tend to use materials that contains concrete facts and real world applications, they are realistic and like to use demonstrated procedure and physical experiments. While the intuitive learners prefer to use materials that contains abstract and theoretical information, they tend to understand the overall pattern from a global picture and then discovering possibilities. The visual (Vi) learners prefer to see what they learn by using visual representations such as pictures, diagrams, and charts. While the verbal (Ve) learners like information that are explained with words; both written and spoken. Sequential (Seq) learners prefer to focus on the details by going through the course step

by step in a linear way. In the opposite, the global (G) learners prefer to understand the big picture by organizing information holistically

In this work, they have proposed an approach which aims to predict the learners' learning style automatically, this approach consists of two steps; in the first step the learners' sequences were extracted from the log file then transformed to an input of the K means algorithm. The k means algorithm was used to group students into sixteen clusters based on FSLSM, where each cluster was labeled with a learning style combination. The second step consists in performing an unsupervised algorithm (Naive Bayes) to predict the LS for a new sequence. We evaluated the performance of our approach using the confusion matrix. The produced results show that our approach performs well. In the future work we will have compared the performance of the naive Bayes classifier with other machine learning techniques such as the neural networks and decision tree.

A group of **Brazilian authors [3]** came up with E-learning Student Reactions in 2019. This Dataset was compiled after 4 months of an Algorithm Introductory Class at a Brazilian University.

A traditional grading system was adopted for evaluation of the students performance, and, at the same time, an online environment let students share posts, answers and classify productions with emojis-based reactions.

The Class was project-based and the evaluation of the skills followed the so-called "21th Century Skills", in a scale from 0 to 10 each Skill:

- Critical Thinking and Problem Solving Skills - named as SK1;
- Criativity and Inovation Skills - named as SK2;
- Constant and Self Learning Skills - named as SK3;
- Collaboration and Self-Direction Skills - named as SK4;
- Social and Cultural Responsability - named as SK5.

T. Sheeba, Reshmy Krishnan [4] came up with Automatic Detection of Students Learning Style in Learning Management System in 2019. Learning style is one of the major factors of student performance in any learning environment. Determining the learning style of students enhances the performance of learning process. This paper proposes an approach to classify students learning style automatically based on their learning behavior. One of the best widely used classifier algorithm is decision tree which

is proposed in this paper. The main concern in decision tree classifier is the construction of significant rules which are required for accurately identifying learning styles. Lack of significant rules would result in misclassification of learning style. Hence, the main focus of this paper is to construct most significant rules which would strengthen the existing decision tree classifier to precisely and accurately detect the learning style of students. The student behavior is obtained from the web log files and then mapped with three learning dimensions of standard Felder Silverman learning style model. Subsequently, by employing significant rules in decision tree classifier, the student behavior has been automatically classified with high accuracy. This approach was experimented on 100 students for the online course created in Moodle Learning Management System. The evaluation result is obtained using inference engine with forward reasoning searches of the rules until the correct learning style is determined. The result is then analyzed with a confusion matrix of actual class and predicted class which shows that processing dimension shows variance whereas perception and input dimension were detected correctly with an average accuracy of 87%.

The various studies in learning system have proved that the learners learning have improved when the educators' teaching style match with the learners learning styles. T. Sheeba (&) Department of Computer Science and Engineering, Karpagam University, Coimbatore, TamilNadu, 641021, India e-mail: tsheebat2002@yahoo.co.in R. Krishnan Department of Computing, Muscat College, P.O. Box: 2910 P.C: 112, Ruwi, Sultanate of Oman e-mail: reshmy_krishnan@yahoo.co.in © Springer Nature Switzerland AG 2019 A. Al-Masri and K. Curran (eds.), Smart Technologies and Innovation for a Sustainable Future, Advances in Science, Technology & Innovation, https://doi.org/10.1007/978-3-030-01659-3_7 45 In order to detect the learning style of learners and to reduce the gap between educators' and learners' style, there is a need to discover each learner's learning style and adapt the courses or assist the learners to best fit to it. There are two ways to obtain learning style. The most popular method is questionnaire which is a static approach and time consuming and may not be accurate. After the emergence of using Learning Management System (LMS) in education, it become more appropriate to acquire the learning style dynamically and indirectly using automatic detection method. There were two approaches used for automatic detection of learning styles: data-driven and literature-based approach. The data-driven approach uses sample data to build a

classifier that imitates a learning style instrument.

This approach mainly uses AI (Artificial Intelligence) classification algorithm which takes the learner model as input and returns learners learning style preferences as output [1]. This approach has the advantage of to be more accurate and less error-prone, as it uses real data to detect students' learning styles. On the other hand, literature-based methods uses simple rules to calculate learning style from number of matching hints. The weakness in this approach is that there is a possibility that some behavioral patterns of learning styles may not be taken into account for the calculation of learning style [2]. The proposed system employs data-driven approach using AI classification algorithm to automatically detect learning styles. Several models of learning styles have been described for defining learning styles such as Kolb's, Gardner, Felder and Silverman, Biggs and custom learning style model [1]. Each model has its strength and weakness. Among the proposed models, Felder and Silverman learning style model (FSLSM) is chosen as it is the widely used model proven to be suitable to use in educational systems and tested in engineering education exhibiting a good degree of reliability, validity and internal consistency. This model describes the learning style of learners into four different dimensions (active/reflective, sensitive/intuitive, visual/verbal, sequential/global) based on the behavior patterns of learners using the system. Depending on the FSLSM, various research works have been proposed to detect learning styles on a data driven approach using several AI classification algorithms. The various AI techniques used are Bayesian network [3], Decision Tree [4], Genetic Algorithms [5, 6], Neural Network [7], Genetic algorithms with K-NN [1] etc. Comparison of these algorithms in terms of four dimensions [Processing (Pr), Perception (Pe), Input (In) and Understanding (Un)] of FSLSM are shown in [8, 9]. As each algorithm has its own strength and weakness, the most widely used AI algorithms for classification are Bayesian network, Neural Network and Decision Tree.

Bayesian network uses conditional probabilities to represent uncertainty of data, but it has no universally accepted standard for constructing this network from data. Neural network shows good classification accuracy as it has the learning capability from specific examples, whereas it has the weakness of having high computational complexity; no theoretical rule defined to determine the optimal number of hidden neurons; complex to

define number of hidden layers; lack of descriptive power and difficult to identify rules for both inputs and outputs. The decision tree is a widely used classification algorithm which has proved to have high accuracy. This algorithm requires significant classification rules for identifying learning styles. The previous studies reported in the decision tree implementation does not show significant rules for identifying learning styles based on FSLSM. Hence, the construction of most significant rules would be the main focus of the proposed system in order to strengthen the existing classifier to make accurate classification and prediction of learning style of learners and to emphasis on the automatic classification of learning styles when weblog files are given as input to the classifier which is not been previously done by other researchers in this area. The proposed architecture is shown in. The approach starts with data collection which controls the process of collecting web log files created automatically in Moodle LMS based on the student actions. Then data preprocessing is done to acquire learning style from web log files. It includes weblog analysis in which students' behavior is extracted and using significant rules of decision tree, learning style is extracted based on FSLSM. Web log files contain all the behaviors that the learner performs in Moodle LMS. These logs are automatically created when the learners use the system. It records all the activities in the form of chats, forums, quizzes, exercises, assignments, exam delivery, frequency of accessing course materials etc. Initially, the web log files of each learner are collected and filtered out by the activities and their corresponding actions. Data Preprocessing Data preprocessing includes web log analysis in which learner actions are extracted based on the three dimensions (processing, perception and input) of FSLSM.

The proposed work is experimented on around 100 online learners taking five online computing courses created in Moodle LMS. The learners registered for these five online courses had no previous knowledge on the topics taught in the course, since all these courses are delivered as basic units. These courses cover rich course topics represented in different types of elements such as texts, hyperlinks, assignments and quizzes etc. More number of assignments is presented to the learners for each topic as per the requirement of the course and the learners can access as many assignments based on their need. Also, learners are given more number of quizzes including formative and summative assessments, in which the learners can participate themselves at any time. Any number of quizzes can be attended by the learners whenever they want to practice the questions to

test their knowledge. Also, interaction among learners and educators are provided with the help of forum discussion and chat room facilities in order to discuss and share the ideas and views about the topic by posting and replying to others. All the learner activities performed in Moodle LMS while learning are stored automatically in the web log files. The log files automatically record the learner activities such as the type of learning content preferred (text, PowerPoint, hyperlinks), the number of assignments done, the number of quizzes attended with number of revisions and delivery time and participation in discussion forums and chat facilities etc. which are used to detect each learners learning style automatically. These actions are extracted from the log files and then processed in terms of the category using decision tree learning algorithm to detect each learner learning style. Algorithm Input: Weblog files of each learner Output: Learning style of each learner in three dimensions of FSLSM Test on each possible value of decision node and descend the appropriate branch e. Test repeated until a leaf node is reached f. Return the classification of leaf node which is the learning style. Evaluation is done by determining the performance of decision tree classifier for around 100 students using classification accuracy. The classification accuracy determines the percentage of correct instances placed in the correct category. Equation (1) gives the formula used for calculating classification accuracy and error rate for the three dimensions of FSLSM model: Accuracy ¼ Correct Predictions=Total Number of Predictions Error Rate ¼ Wrong Predictions=Total Number of Predictions ð1Þ The achieved experimental results shown in ranges from 75 to 95% shows a high precision for the three dimensions of FSLSM, thereby proving its suitability for identifying the learning style of learners with an average accuracy of 87% and a very less error rate of 13%.

By analyzing the evaluation results obtained, decision tree classifier is able to classify perception dimension with a high precision of 95% as most learners did most of the exams and exercises provided for the course and hence significant variance not found in this dimension. The input dimension is classified with a precision of 90% as most of the learners are interested in using the power point slides and hyperlinks instead of using the theoretical notes for their study and hence shown less variance in the predicted class. The processing dimension is classified with precision of 75% as the learners found to use less use of forums and chats facilities and hence variance found in this dimension. The main reason for the variance is that most of the learners have used the facilities such as chat,

email and forum very less. As a conclusion, learners should be encouraged to use the facilities of forums, chat rooms, email etc. in order that the active learners may be discovered and to use collaborative tasks such as assignments and exercises. The number of assignments and exams could also be increased for an effective analysis of learning style. Confusion Matrix. The evaluation of classification results is done by a confusion matrix to illustrate the accuracy of decision tree classifier solution. The confusion matrix contains information about actual and predicted classifications which is used to determine the performance of the decision tree classifier. In confusion matrix, for a single prediction, there are four different outcomes: true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN) for a two-class case with classes "1" ("yes") and "0" ("no"). A false positive is when the outcome is incorrectly classified as "yes" (or "positive"), when it is in fact "no" (or "negative"). A false negative is when the outcome is incorrectly classified as negative when it is in fact positive. True positives and true negatives are obviously correct classifications. Input Dimension: Table 4 shows the confusion matrix obtained for the prediction of three different learning styles (Neutral, Extremely_Visual and Extremely_Verbal) of input dimension. Shows the confusion matrix obtained for the prediction of five different learning styles (Extremely Reflective, Medium Reflective, Neutral, Medium Active, Extremely Active) of processing dimension. The values in the diagonal of Table 5 would always be the true positives (TP).

Ling Xiao Li, Siti Soraya and Abdul Rahman [5] came up with Students' learning style detection using Tree augmented Naïve Bayes. Generally, a learning style model classifies students according to where they fit in several scales that identify the ways in which they receive and process information [17].

In this paper, we consider Felder–Silverman learning style model (FSLSM), because of the main reasons as follows:

- FSLSM is the most widely used learning style model. Shockley (1) & Russell [3] analyzed the use of learning style model in the adaptive learning system over the past decade and found that the usage amount of FSLSM model ranks the first (50%), much higher than the second Kolb's model (8.6%). The findings of this study are also consistent with those of Akbulut & Cardak [18].
- FSLSM provides more detailed descriptions than other learning style models while its reliability and accuracy have also been proven [(2)19].

• FSLSM provides a high operational index of learning style (ILS) instrument, which includes 44 questions: 11 questions for each dimension, where each question has two answers to be chosen from, to detect both the preference and the degree of preference.

FSLSM divides the learning styles into four dimensions:

(i) procession (active/reflective),

(ii) perception (sensing/intuitive),

(iii) input (visual/verbal)

(iv) understanding (sequential/global).

Procession: active students do not learn much in situations that require them to be passive. Whereas reflective students do not learn much in situations that provide no opportunity to think about the information being presented. Active students work well in groups; reflective students work better by themselves or at most, with one other person.

Perception: sensing students prefer facts, data and experimentation, whereas intuitive students prefer principles and theories. Sensing students are patient with detail but do not like complications, whereas intuitive students are bored by details and welcome complications.

Input: visual students remember best what they see, and this includes pictures, diagrams, timelines, films, and demonstrations. Verbal students remember much of what they hear or read and speak.

Understanding: sequential students follow linear reasoning processes when solving problems. Global students make intuitive leaps and may be unable to explain how they come up with solutions. Sequential students can work with the material when they understand it partially or superficially, while global students may have great difficulty in doing so.

Bayesian network is an uncertain relationship representation and reasoning model based on probability analysis and graph theory. It is a directed acyclic graph where nodes represent random variables and arcs represent the probabilistic correlation between variables [21].

The absence of edges in a Bayesian network denotes statements of independence. A Bayesian network encodes the following statement of independence about each random variable: a variable is independent of its non-descendants in the network given the state of its parents [22]. A Bayesian network also represents a particular probability distribution, the joint distribution over all the variables represented by nodes in the graph.

This distribution is specified by a set of conditional probability tables (CPT).

Each node has an associated CPT that specifies this quantitative probability information. Such table specifies the probability of each possible state of the node given each possible combination of states of its parents. For nodes without parents, probabilities are not conditioned on other nodes. These are called the prior probabilities of these variables.

Past studies [8] found that Bayesian networks are very suitable to be employed as the detection technique for adaptive education domain. However, the general Bayesian network is too complex for small datasets and easy to overfit [5]. Naive Bayesian avoids this problem. The naive Bayesian classifier is an effective classifier due to two advantages that it has over other classifiers. Firstly, it is easy to be constructed as the structure is given a priority besides no structure learning procedure is required. Secondly, the classification process is very efficient. Both advantages are derived by assuming that all features are independent of each other. The simple structure only contains two layers, the classification node as the parent node of all other nodes. No other connection is allowed in the naive Bayesian network as shown in figure 1. The only connections link between node c and all leaf nodes x1, x2, x3, x4; the naive Bayes assumes that all leaf nodes are conditionally independent. But the conditional independence assumption in the naive Bayes is rarely true in reality. In an adaptive educational domain, the naive Bayesian assumption is (nearly) always violated because the variables are often interconnected.

The results showed that the tree augmented naive has higher precision than the Bayesian network. This is because the tree augmented naive algorithm loosens the conditional independence assumption which is consistent with the reality (the interconnection between variables). Comparing with naive Bayesian, tree augmented naive Bayesian allows the additional edges between the attributes of the network to capture correlations among them [29].

Furthermore, each attribute can have augmenting edge which encodes statistical dependencies between attributes; therefore, the joint probability of tree augmented naive Bayesian count on the probabilities conditioned not only on class but also from the attribute of parent node [30]. During students' online learning process, many internal connections existed between learning objects within the same learning style dimension, such as ‘online chat’ often appeared together with ‘forum’ section; when the correlation of the interconnection is higher, the result of tree augmented naive achieves better. On the

other hand, the tree augmented naive algorithm takes slightly more time than the Bayesian network. This is because the tree augmented naive needs to build the tree based on Bayesian network tree.

In this study, 36 students' data were used for training the classifier. Nevertheless, when looking at the possible variables, for example, in perception dimension, which has 4 different features, 81 (34) possible different states exist because each feature can have 3 states. Using only 36 students as input data might affect the precision of the detected results. Meanwhile, from another point of view, the precision of the results using the proposed approach could be further improved when running in big dataset environment. Another limitation of the current study is that the results of the experiment were only tested on Moodle platform with a specific subject. The consistency of performance needs to be tested when it runs with different learning management platforms or other online courses. Our future work will involve exploring further the performance in different environments.

We have evaluated the capability of tree augment naive Bayesian to model and detect students' learning styles. The results obtained are positive. Since the tree augmented naive Bayesian network retains the structural features of naive Bayes and relaxes its independence assumption, we could make classifications with higher accuracy. Experimental results prove that the proposed method is more accurate than the results obtained using the Bayesian network.

Although the experiment only assessed restricted numbers of students, the results obtained provide valuable data about students' learning behaviours with regards to online courses. These data will be used in future to enhance students' learning style modelling. For future work, the experiment will be carried out on a larger scale in order to validate the results obtained so far and to test the performance consistency.In summary, provided that we take into account issues on learning style detection, the proposed tree augmented naive Bayesian model enables us to discover students' learning styles in a highly precise manner.

Muhammad Said Hasibuan, Lukito Edi Nugroho, Paulus Insap Santosa [6] came up with Detecting Learning Styles with Artificial Neural Networks in 2019. The learning style detection process can be divided into conventional and automatic forms (Hasibuan et al.,

2017). Detection of conventional learning styles uses the questionnaire provided by each learning style. Questionnaires include Learning Style Instrument (LSI) for Kolb's learning style, Learning Style Questionnaire (LSQ) for Honey and Mumford style of learning, Index of Learning Style (ILS) for the Felder-Silverman learning style model and questionnaire for Visual, Aural, Read and Kinesthetic (VARK) learning style (Kolb, 1984; Honey & Mumford, 1992; Fleming & Mills, 1992; Felder & Silverman, 1988). The detection of automatic learning style can be divided into two, namely data-driven and literature-based. Data-driven is the imitation of a questionnaire that leads to one of the artificial intelligence methods. Some research has been done with artificial intelligence i.e. Bayesian Network, Decision Tree and Neural Network, NB Tree algorithm, Hidden Markov and Genetic Algorithm (Özpolat & Akar, 2009; García, Amandi, Schiaffino & Campo, 2005; Cha, Kim, Park, Yoon, Jung & Lee, 2006; Yannibelli, Godoy & Amandi, 2006). -86- Journal of Technology and Science Education – https://doi.org/10.3926/jotse.540 Patricio detected the learning style of Felder-Silverman Learning Style Model (FSLSM) using Bayesian Network performed by Patricio (García et al., 2005). The process of detection is only done to the three dimensions that have FSLSM learning style, that is, perception, processing and understanding. The results of the detection of learning styles performed by Patricio have a level of accuracy below 77%. The same thing was done by Özpolat to detect the learning style of FSLSM by using NB Tree Classification (Özpolat & Akar, 2009). But in the study by Özpolat, the FSLSM learning style detection process used four dimensions: perception, processing, understanding and input. The results of the Özpolat study have an accuracy of below 73.3%. Furthermore, a study by Cha et al. detected the FSLSM learning style using the Decision Tree and Hidden Markov approaches (Cha et al., 2006). Cha et al. used Decision Tree and Hidden Markov to detect FSLSM learning styles with four dimensions, namely, perception, processing, understanding and input. The results of Cha et al. showed detection accuracy below 83%, a value which is better than the studies by Özpolat and García. Detection of the next learning style is Literature-Based; this work is based on learner visits to teaching materials. The learning system will retrieve the log data of the learner's visit to the teaching materials to determine the learning style (Rita et al., 2002; Nam, 2013; Popescu, Badica & Trigano, 2008). The data taken includes the length of visits to teaching materials and the pattern of visits.

Subsequent research by Hamtini has successfully detected the Visual Aural Kinethestic (VAK) learning style (Hamtini, 2015). This study captures learners' visits to learning

materials (contents, case studies, examples, exercises and assessments) and assesses learner, visit and answers behaviours. The results of this VAK learning style detection yielded 52.78% accuracy. Research by Ahmad et al. detected FSLSM learning style using a literature-based technique (Ahmad, Tasir, Kasim & Sahat, 2013). The dimensions of learning styles studied were the Active and Reactive dimensions. Both Active and Reactive dimensions work with Threshold and timing. The results of this study have an accuracy of 79.63%. Another study was conducted by Graf et al., who detected FSLSM learning styles using a literature-based method (Graf, Viola & Kinshuk, 2007). This study uses the pattern of interaction of learners when accessing content, outline and examples. When a learner visits content, outline and example, this study measures the time and number of visits. The result of FSLSM learning style detection conducted by Graf et al. yielded 79.33% detection accuracy. A study by Liyanage also detects FSLSM learning styles with a literature-based approach. Liyanage uses two methods: questionnaire and rule-based (Pitigala Liyanage, Gunawardena & Hirakawa, 2013). This study is similar to other literature-based research that uses interaction results of learners to outlines, contents, examples, self-assessments and exercises. The results of the use of an approach based on questionnaires and rules showed 77.5% accuracy. A similar study to Liyanage's, that focused on detecting learning styles with the Learning Management System, was done by Abdullah, Alqahtani, Aljabri, Altowirgi and Fallatah (2015). Abdullah et al. (2015) detected all learning styles of FSLSM with an accuracy of up to 90% but only for processing dimensions. Research on other literature-based learning style detection was done by Dung et al (Dung & Florea, 2012).

Their research detects FSLSM learning style by constructing LMS POLCA. LMS POLCA provides learning materials accessible to learners. LMS POLCA provides a variety of teaching materials that every learner can access. Learning interaction process of this learning material will determine learning style of learners. -87- Journal of Technology and Science Education – https://doi.org/10.3926/jotse.540 LMS POLCA will calculate the time the system provides compared with the actual time accessed by the learner. The results of this comparison will be the basis of decision making on learning styles.

This research has succeeded in developing a model of learning style detection by using a prior knowledge approach. The prior knowledge generation process uses the Latent Semantic Indexing (LSI) approach. The LSI approach has succeeded in generating learners' learning

styles more accurately than previous detection models. This successful because the detection process is carried out at the beginning of a new topic. Previous studies detected only the beginning of learning by using a vark questionnaire that did not relate to prior learning knowledge. After the process of generating learning styles with LSI is completed, the next step is to predict learning style by using ANN. To ensure that the vark learning style is appropriate, confirmation is made by giving online questions by adopting a vark learning style questionnaire. Further research is expected to develop a detection and prediction system that is integrated either with the Learning Management System or e-learning system. Intelligent dimensions are closely related to the ability of learners who can record, store and recall previous teaching materials. Meanwhile, the attitude dimension shows the attitude of learners who seek to have knowledge. Learner attitude can be divided into two categories, namely, able to increase motivation or demotivation. Both of the above dimensions relate to the prior knowledge of the learner. This research builds a learning style detection model using an internal approach, i.e. prior knowledge. So far there are three methods to generate prior knowledge: these are brainstorming technique, cognitive map and Know-Want To Know-Learned (KWL) chart. These three methods have limitations such as inefficient use of time, accuracy of improper detection results and insufficient subjectivity. The solution to the generation problem is that this research uses Latent Semantic Indexing (LSI) to generate prior knowledge, and the result of this generation will predict learning style. Learning style prediction is done by using the Artificial Neural Network (ANN) method.

Their research detects FSLSM learning style by constructing LMS POLCA. LMS POLCA provides learning materials accessible to learners. LMS POLCA provides a variety of teaching materials that every learner can access. Learning interaction process of this learning material will determine learning style of learners. -87- Journal of Technology and Science Education – https://doi.org/10.3926/jotse.540 LMS POLCA will calculate the time the system provides compared with the actual time accessed by the learner. The results of this comparison will be the basis of decision making on learning styles. This study claims to have produced learning style detection accuracy up to 79.54%.

2.3 EXISTING SYSTEM

Currently in many schools and other educational institutes traditional learning methods are being used to teach the students. Similarly, models and other text materials are used to teach students about human anatomy. The lack of visualization makes it difficult for them to understand the complex anatomy structures. We can overcome these challenges by using 3d visualization, where students can view the human anatomy models in 3d and can also zoom in zoom out and get a clear view of the minute details which are otherwise not visibleto the naked eye.

Some of the existing works include an AR application that can be used to learn anatomy of the human ear. Most of the previous works are a maker-based AR application where theuser scans any kind of marks (e.g., QR code). Markers were used to detect and allow any assigned image to recognized and then to be displayed on the tablets screen. A marker canbe any image on any surface which it will be scanned using the camera. For the markers a device database was created using Vuforia database and in the database the new targets were entitled and identified.

2.4 DRAWBACKS OF EXISTING SYSTEM

- Real datasets have not been used for the validation of the proposed models
- In some publications, the models have not been compared to others for accuracy, precision, and recall
- Most of the papers that used the FSLM learning style model mostly skipped either on or two dimensions of the model
- Most of the publications reviewed worked with Kolb’s learning style inventory and FSLM model: a very few of them used Gardner’s theory of multiple intelligence

3. SOFTWARE REQUIREMENT ANALYSIS

3.1 INTRODUCTION

Personalization has been a part of computing since the past few decades, and each system currently being developed provides some sort of personal touch to the users. E-learning systems have evolved a lot, from simple computerized instruction and quizzes to adaptive virtual environments. Learning styles are a category under a broader umbrella of learner characteristics; learner characteristics are the characteristics of the learner that affect the learning process. Learning style is the preferred way of using one's ability to learn Different psychologists have given us different models of learning styles

3.1.1 DOCUMENT PURPOSE

The goal is to give makers of learning style prediction tool advice on how to go about the building of the model and deliver the requirements accordingly.

3.1.2 DEFINITIONS

Learning style

Learner's style is grouped into four types mainly; Visual, auditory, kinesthetic and Read/Write. Each type of learners learns primarily through one of the main receiving senses, visual, listening, or by doing. Learner style influences the learning process and learner's achievement. It is better to select suitable learning tool for the learner according to his learning style.

FSLM model

The Felder-Silverman learning styles model describes **sensing learners as those who prefer to deal in facts**. When problem solving, they rely on tried-and-true methods and formulas. They lean toward real-world scenarios. Intuitive learners, on the other hand, are interested in innovation and novelty.

Kolb's' model

The Kolb states that learning involves the acquisition of abstract concepts that can be applied flexibly in a range of situations. In Kolb's theory, the impetus for the development of new concepts is provided by new experiences.

Feature Selection

In machine learning and statistics, feature selection, also known as variable selection, attribute selection or variable subset selection, is the process of selecting a subset of relevant features (variables, predictors) for use in model construction. Feature selection techniques are used for several reasons:

- simplification of models to make them easier to interpret by researchers/users.
- shorter training times.
- to avoid the curse of dimensionality.
- improve data's compatibility with a learning model class.

encode inherent symmetries present in the input space.

Hybrid Model

Hybrid ML models are made through integration of ML methods, with other ML methods, and/or with other soft computing, optimization techniques to improve the method in various aspects. While the ensemble methods are made using various grouping techniques such as bagging or boosting to use more than one ML classifier.

Boosting

Boosting is an ensemble modeling technique which attempts to build a strong classifier from the number of weak classifiers. It is done building a model by using weak models in series. Firstly, a model is built from the training data. Then the second model is built which tries to correct the errors present in the first model. This procedure is continued, and models are added until either the complete training data set is predicted correctly or the maximum number of models are added.

AdaBoost

AdaBoost was the first successful boosting algorithm developed for the purpose of binary classification. AdaBoost is short for Adaptive Boosting and is a very popular boosting technique which combines multiple “weak classifiers” into a single “strong classifier”. It was formulated by Yoav Freund and Robert Schapire. They also won the 2003 Gödel Prize for their work.

Random Forest

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

TensorFlow

TensorFlow is an end-to-end open-source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries and community resources that lets researchers push the state-of-the-art in ML and developers easily build and deploy ML powered applications.

Pandas

pandas is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool, built on top of the Python programming language.

3.1.3 REQUIREMENT ANALYSIS

Requirement analysis is also known as requirement engineering. It's a process where we determine user expectations for a modified or new software. Sometimes it is also referred to as requirement capture or requirements gathering in the context of software engineering. Requirement analysis encompasses the tasks involved in determining the conditions or needs to meet for a change do new project or product, considering the various stakeholders potential conflicting requirements, and documenting, analyzing, and managing system or software requirements. Bridging the difference software design and system requirements engineering is a software engineering task.

The analysis of requirements is crucial to the failure or success of a software or systemsproject.Also,therequirementsshouldbetraceable,testable,measurable,actionable,documented, related to identified business needs or opportunities, and detailed enough forsystemdesign.Organizationscanuserequirementanalysistoidentifytheneedsofstakeholders.Simultaneously,itenablesthe development team to communicate with stakeholders in a common language that everyone understands(such as flowcharts, models, and charts) rather than pages full of texts.

We gather all the requirements in a document called Software Requirements Specification. After gathering the requirements, the user stories are shared for approval with the stakeholders. This document can be understood by non-technical and technical users. The changes that are made are documented and go through a protocol and then approved. Requirement's analysis is a collaborative effort that necessitates knowledge of hardware-software, and human factors engineering, as well as interpersonal skills.

A software requirement is the capability of the software that as system component or system must have to satisfy specification, contract, standard or any other documentation that is imposed. The goal is the create a software that meets the customers' needs and is of high quality and to must be able to finish on time in the given budget.

3.2 SYSTEM ARCHITECTURE

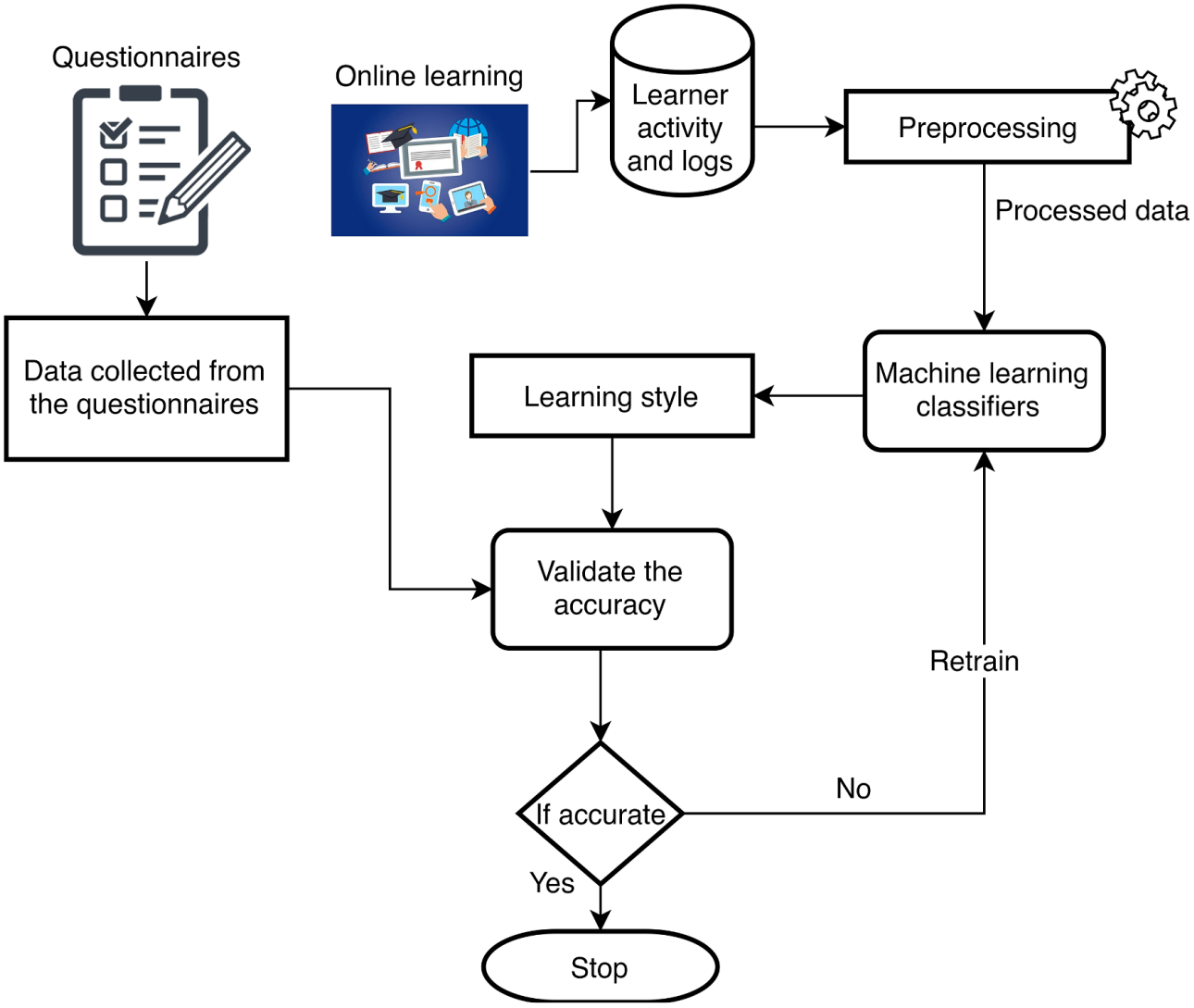

Fig 3.2.1: System Architecture

It is a conceptual model which defines a system's structure, views, and behavior. An architecture description is a formal description and representation of a system that is organized in a way that allows for reasoning about the system's structures and behaviors. A system architecture can be made up of system components and developed sub-systems that will work together to implement the overall system.

A system architecture focuses primarily on the internal interfaces between the system's components or subsystems, as well as the interface between the system and its external environment, particularly the user.

(In the case of computer systems, this latter, unique interface is known as the computer human interface, also known as the human computer interface, or HCI; formerly known as the man-machine interface.)

There The architecture as illustrated, comprises of the modules, questionnaires, online learning, learner activity and logs, preprocessing, machine learning classifiers, learning style, validating the accuracy.

The questionnaires collected help in the testing of the data's accuracy later on, which contains responses to different questions like amount of time spent on a module, the amount of knowledge gained or understood, and many others. The online learning platform is an interface where the students learn from. The data preprocessing includes feature selection, data normalization, removing null values and many more. The learning style is predicted using the machine learning classifiers and the data collected from questionnaires is checked against it to validate the accuracy of the model.

The inputs to the classification algorithm are the attributes in the normalized form; the data was observed from the learner logs, and values were normalized as discussed earlier. The output classes being the dominant value of intelligence. The attributes have been crafted after studying a lot of research/

in the area, a study of media associated with each type of intelligence can be found in (Kolås & Staupe, 2007). Another example of designing a rule-base for e-learning recommendations using multiple intelligences is developed in (Kaewkiriya et al., 2016). Igbrue and Pathak (Igbrue & Pathak, 2008) created e-learning content using the Multiple intelligences theory with an attempt to increase the learning efficiency of learners. Authors in (Mankad et al., 2011) used fuzzy logic and evolutionary rules to build a hybrid system to detect the multiple intelligence of the learner. The rules or parameters have been identified using the characteristics of learners with each dominant intelligence, the media selection, the time they spent on each media, and others.

The FSLM model is a 11 points scale. If the score on a scale is 1-3, then the user has mild preference, 5-7 is moderate preference and 9-11 is strong preference. The studies discussed in the literature review section give us a picture of what learners with learning

style do in online learning. If all/ most of the activities of the learner point towards the visual learning style, the learning style as predicted by the model is visual, and if all/ most of the activities of the learner point towards verbal learning style, the learning style as predicted by the model is verbal. This behavior of machine learning models is common; machine learning models help in predicting the class of a learner according to the inputs supplied. However, these classifiers do not indicate the tendency or intensity of any class. If both the activities/ actions are equal, then there is marking as visual-verbal. However, some ambiguities occur which will be resolved during our future work.

We collected data from graduate students studying a bachelor's degree in computer science. Metrics measures the quality of prediction performed by the machine learning algorithms. The most common metrics are accuracy, precision, recall, F1-score, cross-validation score, and area under the curve. There are metrics other than the mentioned above, but researchers use these while performing classification and prediction. The area under curve and cross-validation also explain to us whether the results are significant or not.

3.3 FUNCTIONAL REQUIREMENTS

- The device shall require the worker to authenticate.
- The device shall support a python 3.4+.
- The device shall support multi-thread rendering.
- The device shall have all latest packages as required.

The device shall provide Anaconda suite (Anaconda is a distribution of the Python and R programming languages for scientific computing, that aims to simplify package management and deployment. The distribution includes data-science packages suitable for Windows, Linux, and macOS.).

- It is free and open source
- It has more than 1500 Python/R data science packages
- Anaconda simplifies package management and deployment
- It has tools to easily collect data from sources using machine learning and AI
- It creates an environment that is easily manageable for deploying any project
- Anaconda is the industry standard for developing, testing and training on a single machine
- It has good community support- you can ask your questions there

The device shall provide a minimum supported versions of libraries like TensorFlow, SkLearn, etc. (Scikit-learn is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms).

3.4 SYSTEM ANALYSIS

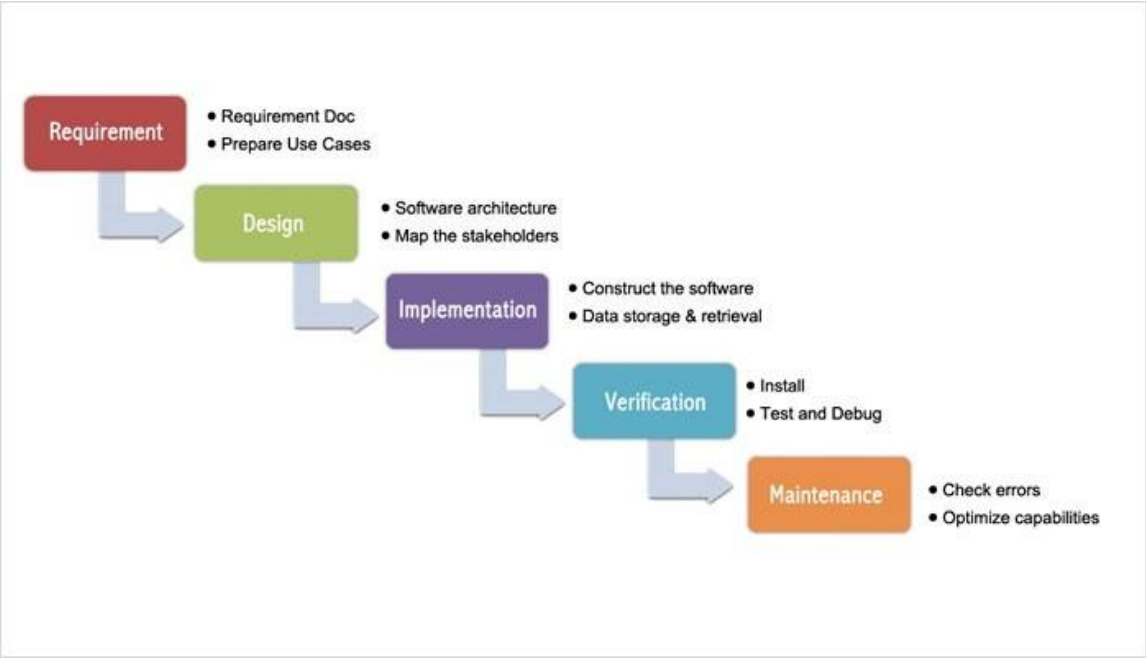

Fig 3.4.1: Waterfall Model

It was the first to be introduced as a Process Model. It's also known as a linear sequential life cycle model. It is extremely easy to apply and grasp. In this model, phases do not overlap and before each phase beings its previous phase much be completed.

The first SDLC approach used during software development was waterfall model.

The model shows that the development of software is linear and is a sequential process. Only after one phase of the development is completed, we can go to the next phase. The phases in this waterfall model do not overlap.

The steps in the waterfall model are explained below.

Requirements: At this point, the search is more intense and focused on the software's requirements. To understand the nature of the program to be created, the software engineer must first understand the information domain of the software, such as the functions required, user interface, and so on. This second activity (finding system requirements and software) must be documented and presented to the customer.

Design: Before coding begins, this stage process is used to change the above requirements as a representation in the form of "blueprint" software. The design must be capable of meeting the requirements outlined in the previous stage. This process, like the previous two, must be documented as the software configuration.

Implementation: The design was transformed into a form that machines could understand in order to be understood by a computer in some cases machine i.e., via the coding process into the programming language. This was the stage in which the programmer will put the technical design phase into action.

Verification: As with anything made, it must first be tested. The same is true for software. All software functions must be tested to ensure that the software is error-free, and the results must strictly adhere to the previously defined needs.

Maintenance: Because the software that is being created is not always exactly like that, software maintenance, including development, is required. When it runs, it may still have some minor errors that were not discovered previously, or if the need for additional features that were not previously available in the software arises. When there is a change in external companies, such as when the operating system or another device, development is required.

Useful factors: The Waterfall model has its advantages like simple to use. Additionally, while using the model all the system requirements can be defined as a whole, explicitly an at the start the product can rum without many issues.

It is economic to make changes in the early stages of the project when there are problems with system requirements then when the problems which arise in later stages.

3.5 NON-FUNCTIONAL REQUIREMENTS

3.5.1 Performance Requirements

The Camera lag should be no more than 0.2 seconds. The app will support 10,000 simultaneous users. The app should be available at all the times. After clicking any button, user shall receive a response within 1 second.

3.5.2 Design Constraints

The AR needs to be lightweight enough that it can run with minimal delay on a standard Android smartphone. The app should be as small a size as possible because storage on phones is important. Information provided by the app about the content may be limited to what is available on the internet. The app will not have access to any student's private data.

3.5.3 Software System Attributes

- **Operability:** Degree to which a product or system has attributes that make it easy to operate and control.
- **Performance**: Performance relative to the number of resources used under stated conditions.
- **Security**: Degree to which a product or system protects information and data, so that persons or other products or systems have the degree of data access appropriate to their types and levels of authorization.
- **Reliability**: The Application should remain reliable by providing the correct details and information anytime, without any cross references.

3.6 SOFTWARE REQUIREMENT SPECIFICATION

Anaconda Suite

Anaconda, with over 25 million users worldwide, the open-source Individual Edition (Distribution) is the easiest way to perform Python/R data science and machine learning on a single machine. Developed for solo practitioners, it is the toolkit that equips you to work with thousands of open-source packages and libraries.

Anaconda Individual Edition is the world's most popular Python distribution platform with over 25 million users worldwide. You can trust in our long-term commitment to supporting the Anaconda open-source ecosystem, the platform of choice for Python data science.

Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda distribution that allows users to launch applications and manage conda packages, environments, and channels without using command-line commands. Navigator can search for packages on Anaconda Cloud or in a local Anaconda Repository, install them in an environment, run the packages and update them. It is available for Windows, macOS and Linux.

Spyder

Spyder is an open-source cross-platform integrated development environment (IDE) for scientific programming in the Python language. Spyder integrates with a number of prominent packages in the scientific Python stack, including NumPy, SciPy, Matplotlib, pandas, IPython, SymPy and Cython, as well as other open-source software. It is released under the MIT license.

Spyder is a powerful scientific environment written in Python, for Python, and designed by and for scientists, engineers, and data analysts. It features a unique combination of the advanced editing, analysis, debugging and profiling functionality of a comprehensive development tool with the data exploration, interactive execution, deep inspection and beautiful visualization capabilities of a scientific package Furthermore, Spyder offers built-in integration with many popular scientific packages, including NumPy, SciPy,

Pandas, IPython, Console, Matplotlib, SymPy, and more. Beyond its many built-in features, Spyder can be extended even further via third-party plugins Spyder can also be used as a PyQt5 extension library, allowing you to build upon its functionality and embed its components, such as the interactive console or advanced editor, in your own software.

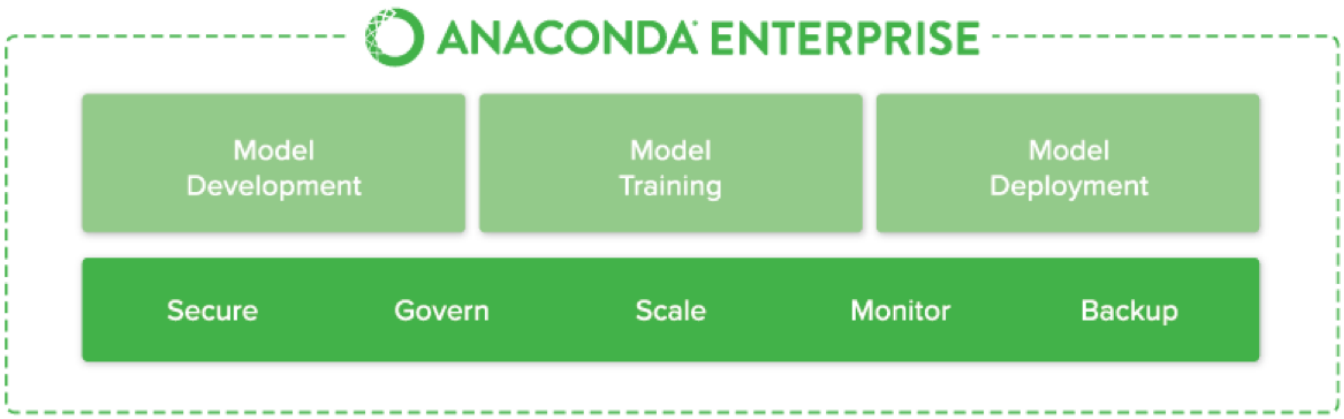

Fig 3.6: Anaconda Suite

3.7 SOFTWARE REQUIREMENTS

- Software's : Anaconda Suite, Python, Git
- Operating System : Windows family/ Linux based
- Technology : Machine Learning
- IDE : Spyder

3.8 HARDWARE REQUIREMENTS

- PC with Dual Core Processor or more
- 4 GB RAM or more
- 180 GB Hard Disk Space or more
- Active Internet Connection

4. PROPOSED SYSTEM

4.1METHODOLOGY

The study initially focused on studying learning style models, their dimensions, values, and combinations of values to identify the learner's learning styles. A study analyzed the performance of learners when they learnt using resources according to their learning style. The study is promising as it proves that presenting learners with customized resources to learn betters their performance. There is a study on linking multiple intelligences and online learning activities helped us identify what learners with various learning styles preferred to do in an online learning system. Students' typical choice of resources, the time taken to learn each concept using their choice, their performance in assessments and other learner attributes were observed. We then extracted indicators from learner's online behavior that will help us to predict their learning style. After the extraction of learner behavior from learner logs, we preprocessed it. The processed data was then passed as input to the machine learning classifiers. The Recognition system is based on machine learning classifiers which predict the learning style of the learner based on the inputs provided. The indicators and attributes are mentioned in table 3 and table 4. The extraction of indicators is necessary because we want to eliminate the use of questionnaires to identify the learning style. To prove the elimination of the use of questionnaires, we evaluate the consistency of questionnaire-based identification and the predicted learning style using the proposed attributes and trained model. The predicted learning style can be used to personalize the learning resources and recommend them specific resources that cater to their learning styles so that learning is optimal.

The proposed methodology is an extension of our work done. We proposed attributes to recognize the type of intelligence based on the theory of Gardner's theory of multiple intelligences. We proposed a neural network architecture to recognize the intelligence, and the intelligences was linked to learning styles. In this paper, we extend the work. Our earlier article didn't implement the proposed work; when we wanted to extend and peruse the work using deep learning, we were not able to achieve the desired accuracy. Deep learning works best with a large dataset, our dataset was small and the trained network over fitted.

To avoid this overfitting and reduce the use of computational power, we preferred not to use deep learning, and hence we made feature selection and removed attributes that had a negative correlation with the output class. With a more in-depth study of literature and the earlier identified attributes, we then shortlisted some attributes. After removing some attributes from our previous work, we add new attributes. We also observe attributes that help to identify a learner's behavior in a discussion forum. The research proposed in this paper follows the research design shown in Figure 1. This module is a part of a more extensive system that performs the sequencing of learning resources.

The research follows an experimental research design; the contributors of data are the usual behavior of the learner with a specific learning style and student behavior attributes in the E-learning system. Data is collected both through literature review and through online logs of learners in the system. The data collected from learners of various classes intended to check whether our trained model can correctly predict their learning style using the proposed attributes or not. For instance, learner 'A' filled the questionnaire initially. A's behavior is tracked in the online learning system. The indicators to extract the proposed attributes are taken from log files. The trained model then uses the indicators to predict A's learning style.

Finally, to evaluate the consistency of the proposed method, the results of the questionnaire and the predicted value is equated. If they are equal, the model is consistent, if not, the model is not consistent. The percentage of consistency for each dimension is discussed in the discussion section. The data and the study do not aim to compare the learning styles of learners from different classes. The study rather focusses on developing a machine learning model to predict the learning style of learners. The data is then cleaned, pre-processed; machine learning models are trained and tested to make predictions. The predictor predicts the learning style of the learner, which is the first outcome. The outcome of exciting patterns in learner behavior with respect to learning style is achieved by carefully observing the learner logs in different situations.

The research methodology followed to predict the learning style, which summarizes the steps taken to detect the learning style of the learner. The actions performed are typical to any data science-oriented machine learning technique used for classification and prediction.

The learners were presented with questionnaires before they began a course, on condition of anonymity, about two hundred students of various classes in a university participated in the experiment. They then started a course on Moodle, which was modified to observe the multiple attributes. The learner activity is monitored from the log facility in Moodle. The unprocessed data was collected, and processing was done to clean the data. The processed data was then used to train and test the classifiers. The classifiers we implemented were decision trees, Support Vector Machines, K-Nearest Neighbor, Naïve Bayes, Linear Discriminant Analysis, Random Forest, and Logistic Regression.

These algorithms are commonly used for classification in machine learning; when the models are trained with enough data, they can be used for prediction. A short account of these algorithms and their working is discussed in the literature review section. Using some attributes from our previous work and proposing new attributes, the following attributes were identified. The attributes in table 2 and 3 may not be enough for this research, but they were selected because they can be either easily observed or derived from the learner logs. Making the attributes complex will create issues in the data collection phase, and a lot of coding should be done to make the interface observe more attributes. We try to analyze the learner's learning style according to two different learning style theories. The first theory is the theory of multiple intelligences, and the next one is the Felder-Silverman model. Both these models are deemed efficient by sufficient proof in the literature to provide personalization in E-learning.

We have made the following assumptions: We have used the relative time in comparison to the total files designated for content. Values are coded. If a learner spends 75% or more of the appointed time, value is 3. If the time spent is between 50% -75% value is 2, and below 50% value is 1. Table 2 contains the attributes used for the deduction of the learner's dominant intelligence defined in Gardner's theory of multiple intelligence. The common characteristic we used in different intelligence deduction is the learning gain which is defined as:

$$Learninggain = (\Sigma PostTestscore - \Sigma PreTestscore)/\Sigma PreTestscore$$ (

(Sum of pre-test score subtracted from the sum of post-test score) divided by the sum of pre-test score)

4.1.1Tools/ Software Used

4.1.1.1Spyder

Unity Spyder is an open-source cross-platform integrated development environment (IDE) for scientific programming in the Python language. Spyder integrates with several prominent packages in the scientific Python stack, including NumPy, SciPy, Matplotlib, pandas, IPython, SymPy and Cython, as well as other open-source software. It is released under the MIT license.

Initially created and developed by Pierre Raybaut in 2009, since 2012 Spyder has been maintained and continuously improved by a team of scientific Python developers and the community.

Spyder is extensible with first-party and third-party plugins, includes support for interactive tools for data inspection and embeds Python-specific code quality assurance and introspection instruments, such as Pyflakes, Pylint and Rope. It is available cross-platform through Anaconda, on Windows, on macOS through MacPorts, and on major Linux distributions such as Arch Linux, Debian, Fedora, Gentoo Linux, openSUSE and Ubuntu.

Spyder uses Qt for its GUI and is designed to use either of the PyQt or PySide Python bindings. QtPy, a thin abstraction layer developed by the Spyder project and later adopted by multiple other packages, provides the flexibility to use either backend.

4.1.1.2SkLearn

The Scikit-learn (formerly scikits. learn and also known as sklearn) is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

The scikit-learn project started as scikits. learn, a Google Summer of Code project by David Cournapeau. Its name stems from the notion that it is a "SciKit" (SciPy Toolkit), a separately developed and distributed third-party extension to SciPy. The original codebase was later rewritten by other developers. In 2010 Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort and Vincent Michel, all from the French Institute for Research in Computer Science and Automation in Rocquencourt, France, took leadership of the project and made the first public release on February the 1st 2010. Of the various scikits, scikit-learn as well as scikit-image were described as "well-maintained and popular" in November 2012. Scikit-learn is one of the most popular machine learning libraries on GitHub.

4.1.1.3Numpy

Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.

Spyder At the core of the NumPy package, is the ndarray object. This encapsulates n-dimensional arrays of homogeneous data types, with many operations being performed in compiled code for performance. There are several important differences between NumPy arrays and the standard Python sequences:

- NumPy arrays have a fixed size at creation, unlike Python lists (which can grow dynamically). Changing the size of an ndarray will create a new array and delete the original.
- The elements in a NumPy array are all required to be of the same data type, and thus will be the same size in memory. The exception: one can have arrays of (Python, including NumPy) objects, thereby allowing for arrays of different sized elements.
- NumPy arrays facilitate advanced mathematical and other types of operations on large numbers of data. Typically, such operations are executed more efficiently and with less code than is possible using Python’s built-in sequences.
- A growing plethora of scientific and mathematical Python-based packages are using

NumPy arrays; though these typically support Python-sequence input, they convert such input to NumPy arrays prior to processing, and they often output NumPy arrays. In other words, to efficiently use much (perhaps even most) of today's scientific/mathematical Python-based software, just knowing how to use Python's built-in sequence types is insufficient - one also needs to know how to use NumPy arrays.

4.1.1.4SciPy

It is a Python-based ecosystem of open-source software for mathematics, science, and engineering SciPy refers to several related but distinct entities:

- The SciPy ecosystem, a collection of open-source software for scientific computing in Python.
- The community of people who use and develop this stack.
- Several conferences dedicated to scientific computing in Python - SciPy, Euro SciPy, and SciPy.in.
- The SciPy library, one component of the SciPy stack, providing many numerical routines.

It provides many user-friendly and efficient numerical routines, such as routines for numerical integration, interpolation, optimization, linear algebra, and statistics. SciPy is a community-driven project. Development happens on GitHub. SciPy is fiscally sponsored by NumFOCUS.

4.1.1.5Pandas

It is a fast, powerful, flexible and easy to use open-source data analysis and manipulation tool, It is built on top of the Python programming language.

- A fast and efficient Data Frame object for data manipulation with integrated indexing.
- Tools for reading and writing data between in-memory data structures and different formats: CSV and text files, Microsoft Excel, SQL databases, and the fast HDF5 format.
- Intelligent data alignment and integrated handling of missing data: gain automatic label-based alignment in computations and easily manipulate messy data into an orderly form.

- Flexible reshaping and pivoting of data sets.
- Intelligent label-based slicing, fancy indexing, and sub-setting of large data sets.
- Columns can be inserted and deleted from data structures for size mutability.
- Aggregating or transforming data with a powerful group by engine allowing split-apply-combine operations on data sets.
- High performance merging and joining of data sets.
- Hierarchical axis indexing provides an intuitive way of working with high-dimensional data in a lower-dimensional data structure.
- Time series-functionality: date range generation and frequency conversion, moving window statistics, date shifting and lagging. Even create domain-specific time offsets and join time series without losing data.
- Highly optimized for performance, with critical code paths written in Cython or C.
- Python with pandas is in use in a wide variety of academic and commercial domains, including Finance, Neuroscience, Economics, Statistics, Advertising, Web Analytics, and more.

4.1.1.6Matplotlib

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK. There is also a procedural "pylab" interface based on a state machine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged. SciPy makes use of Matplotlib.

Matplotlib was originally written by John D. Hunter. Since then, it has an active development community and is distributed under a BSD-style license. Michael Droettboom was nominated as matplotlib's lead developer shortly before John Hunter's death in August 2012 and was further joined by Thomas Caswell.

4.1.1.7Anaconda

The Anaconda is a distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment. The distribution includes data-science packages suitable for Windows, Linux, and macOS. It is developed and maintained by Anaconda, Inc., which was founded by Peter Wang and Travis Oliphant in 2012. As an Anaconda, Inc. product, it is also known as Anaconda Distribution or Anaconda Individual Edition, while other products from the company are Anaconda Team Edition and Anaconda Enterprise Edition, both of which are not free.

Open-source packages can be individually installed from the Anaconda repository, Anaconda Cloud (anaconda.org), or the user's own private repository or mirror, using the conda install command. Anaconda, Inc. compiles and builds the packages available in the Anaconda repository itself, and provides binaries for Windows 32/64 bit, Linux 64 bit and MacOS 64-bit. Anything available on PyPI may be installed into a conda environment using pip, and conda will keep track of what it has installed itself and what pip has installed.

4.2MODULES

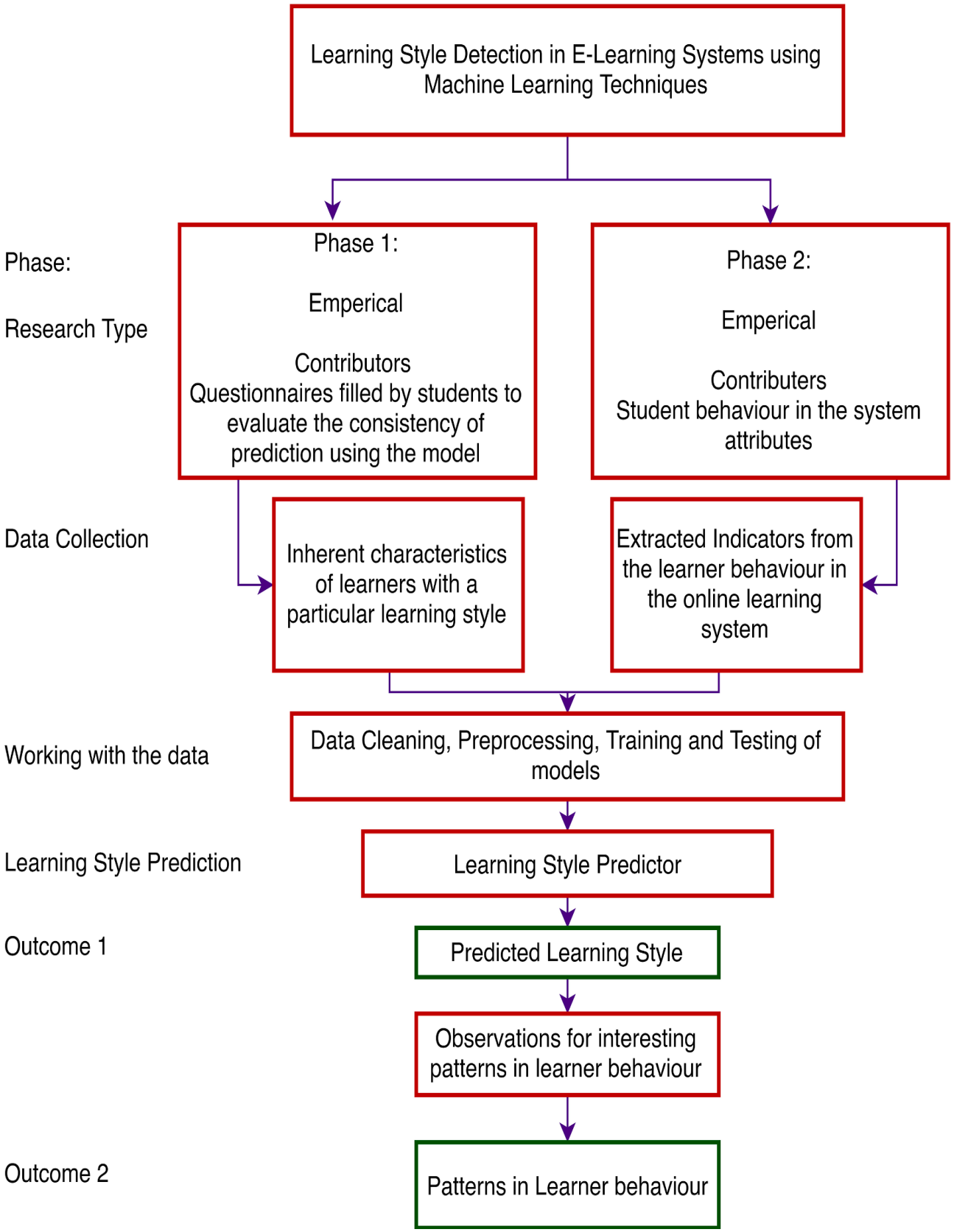

Fig 4.2: Modules in the experiment

Driver Framework

The Deep learning works best with a large dataset, our dataset was small and the trained network overfitted. To avoid this overfitting and reduce the use of computational power, we preferred not to use deep learning, and hence we made feature selection and removed attributes that had a negative correlation with the output class. With a more in-depth study of literature and the earlier identified attributes, we then shortlisted some attributes. After removing some attributes from our previous work, we add new attributes. We also observe attributes that help to identify a learner's behavior in a discussion forum. The research proposed in this paper follows the research design shown.

This module is a part of a more extensive system that performs the sequencing of learning resources. The research follows an experimental research design; the contributors of data are the usual behavior of the learner with a specific learning style and student behavior attributes in the E-learning system. Data is collected both through literature review and through online logs of learners in the system. The data collected from learners of various classes intended to check whether our trained model can correctly predict their learning style using the proposed attributes or not. For instance, learner 'A' filled the questionnaire initially. A's behavior is tracked in the online learning system. The indicators to extract the proposed attributes are taken from log files. The trained model then uses the indicators to predict A's learning style.

This module is a part of a more extensive system that performs the sequencing of learning resources. The research follows an experimental research design; the contributors of data are the usual behavior of the learner with a specific learning style and student behavior attributes in the E-learning system. Data is collected both through literature review and through online logs of learners in the system. The data collected from learners of various classes intended to check whether our trained model can correctly predict their learning style using the proposed attributes or not. For instance, learner 'A' filled the questionnaire initially. A's behavior is tracked in the online learning system. The indicators to extract the proposed attributes are taken from log files. The trained model then uses the indicators to predict A's learning style

Learning style detection in E-learning Systems using machine learning techniques:

There are 2 phases in Research Type Module

Phase ：1 Empirical, contributors

Questionnaires filled by students to evaluate the consistency of prediction using the model.

Phase: 2 Empirical, contributors

Student behavior in the system attributes

Module: Data Collection

Phase: 1 Inherent characteristics of learner with a particular learning style

Phase: 2 Extracted Indicators from the learner behavior in the online learning system

Module: Working with the data

In this module, we perform various data cleaning techniques, get the data preprocessed and ready for consumption to experiment upon it.

Eventually, training and testing of models.

Module: Learning style prediction.

In this module, a learning style predictor is involved.

Module: Outcome 1

This module depicts the one of the outcomes of our project. Predicting learning style.

The observations for interesting patterns in learner behavior are gone through thoroughly.

Module: outcome 2

This module depicts the other outcome of the project.

Capturing patterns in learner behavior.

4.3FUNCTIONALITIES

Functionality of Learner and activity logs

The learner and activity logs are the space where all the student-centric data is stored and used for the later stages of preprocessing and predicting. They consist of various attributes including, time spent on audio files, time spent on a podcast, time spent on reading material, time spent on studying concept maps, time spent on charts and graphs, time spent on studying abstract concepts, time spent on studying images, time spent on study using PowerPoint presentations, number of YouTube videos accessed related to the content among others.

Without Keeping an activity log is also one of the most effective ways of determining where time is wasted. Specific cases can then be isolated, and a method can be found to work more efficiently. Keeping track of and analyzing activities can lead to considerable time gains. Afterwards, time can be better categorized, resulting in a more efficient and productive working schedule.

Many people do not realize that wasting a few minutes every day can cost a lot of time and money in the long term. After all, our brains are not very efficient when it comes to short amounts of time being spent on certain matters and the realization thereof.

This is not necessarily due to people's inability, but due to short-term information that the brain does not transmit to long-term memory storage. Keeping an activity log is therefore highly advisable.

Functionality of machine learning classifiers

This module helps to create a model to take the inputs as mentioned above and predict the learning style. In our case, we have used AdaBoost implementation of random forest algorithm. We have attempted to identify attributes that can be used to detect multiple intelligences and attributes pertaining to the Felder-Silverman model of learning styles. The model is a hybrid model of boosted random forest, which is made through integration of ML methods, with other ML methods, and/or with other soft computing, optimization techniques to improve the method in various aspects. The accuracy obtained was around 87.6%.

Functionality of preprocessor module

Data preprocessing is an integral step in Machine Learning as the quality of data and the useful information that can be derived from it directly affects the ability of our model to learn; therefore, it is extremely important that we preprocess our data before feeding it into our model.

This module helps in the preprocessing stage where the inputs are the attributes taken from the learner and activity logs on which data cleaning steps are deployed, mainly including feature selection, normalization, filling in the null values among others. The feature selection is primal to the overall accuracy as more redundant features don't contribute to the desired output. For the attributes identified for the theory of multiple intelligences, we collected a dataset with 998 samples from students taking online courses. Pre-processing of the data included Feature Selection using Fisher Score, filling missing values with appropriate methods, scaling the data using standard scaler.

4.4ADVANTAGES OF PROPOSED SYSTEM

- Real dataset extrapolated from the Central University in Uttarakhand for the study.
- This data is used as a benchmark for comparison with other related works.
- Usage of newer techniques like boosting, perceptrons
- Identification and inclusion of essential attributes to the model for more accurate prediction.
- Feature selection techniques like Fisher score improved the overall accuracy
- It is a hybrid combination of FSLM and Kolb's Learning Style models.

5. IMPLEMENTATION

5.1 Code For Feature Selection

```
"""
Created on Aug 17

@Author: akarsh
"""

import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split

def Diff(li1, li2):
    li_dif = [i for i in li1 + li2 if i not in li1 or i not in li2]
    return li_dif

df=pd.read_csv(r'G:\mini_project\label_data.csv',index_col=0)

X=df.iloc[:,:-1]
y=df.iloc[:,-1]

from sklearn.feature_selection import mutual_info_classif
import matplotlib.pyplot as plt

ranks= mutual_info_classif(X,y)

feat_importances = pd.Series(ranks,df.columns[0:len(df.columns)-1])
feat_importances.plot(kind='barh',color='teal')
plt.show()

features=[]
index=[]
for i in range(len(ranks)):
    if ranks[i]>=0.005:
        features.append(df.columns[i])
        index.append(i)
list(df.columns)

new_df=df.drop(columns =Diff(list(df.columns),features))

new_df['learning_style']=df['learning_style']

new_df.to_csv("G:\mini_project\data_fs.csv")
```

EXPLANATION:

Feature selection is the process of reducing the number of input variables when developing a predictive model. It is desirable to reduce the number of input variables to both reduce the computational cost of modelling and, in some cases, to improve the performance of the model.

Statistical-based feature selection methods involve evaluating the relationship between each input variable and the target variable using statistics and selecting those input variables that have the strongest relationship with the target variable. These methods can be fast and effective, although the choice of statistical measures depends on the data type of both the input and output variables.

Subset selection evaluates a subset of features as a group for suitability. Subset selection algorithms can be broken up into wrappers, filters, and embedded methods. Wrappers use a search algorithm to search through the space of possible features and evaluate each subset by running a model on the subset. Wrappers can be computationally expensive and have a risk of over fitting to the model. Filters are like wrappers in the search approach, but instead of evaluating against a model, a simpler filter is evaluated. Embedded techniques are embedded in, and specific to, a model.

Information gain calculates the reduction in entropy or surprise from transforming a dataset in some way.It is commonly used in the construction of decision trees from a training dataset, by evaluating the information gain for each variable, and selecting the variable that maximizes the information gain, which in turn minimizes the entropy and best splits the dataset into groups for effective classification.

Information gain can also be used for feature selection, by evaluating the gain of each variable in the context of the target variable. In this slightly different usage, the calculation is referred to as mutual information between the two random variables.

5.2CODE FOR ADABOOST RANDOM FOREST CLASSIFIER

```
# -*- coding: utf-8 -*-
"""
Created on  Aug 21

@author: akarsh
"""

import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split

df=pd.read_csv(r'G:\mini_project\data_fs.csv',index_col=0)


X=df.iloc[:,:-1]
y=df.iloc[:,-1]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=42)


from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

Scaled_X_train=scaler.fit_transform(X_train)
Scaled_X_test=scaler.transform(X_test)


parameter_space = {'random_state':[5,100],
                   'hidden_layer_sizes': [(10,30,10),(20,)],
                    'activation': ['tanh', 'relu'],
                    'solver': ['sgd', 'adam'],
                    'alpha': [0.0001, 0.05],
                    'learning_rate': ['constant','adaptive']}



from sklearn.ensemble import AdaBoostClassifier
from sklearn.ensemble import RandomForestClassifier
clf_ADB =
AdaBoostClassifier(random_state=0,base_estimator=RandomForestClassifier(random_state=0))
clf_ADB=clf_ADB.fit(X_train,y_train)

y_pred=clf_ADB.predict(X_test)
```

```
# _________ACCURACIES_____________________
import sklearn.metrics as metrics

print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
metrics.confusion_matrix(y_test, y_pred)

print("Accuracy:",metrics.accuracy_score(y_test, snn_predictions))
```

EXPLANATION:

The AdaBoost algorithm, short for Adaptive Boosting, is a Boosting technique used as an Ensemble Method in Machine Learning. It is called Adaptive Boosting as the weights are re-assigned to each instance, with higher weights assigned to incorrectly classified instances. Boosting is used to reduce bias as well as variance for supervised learning. It works on the principle of learners growing sequentially. Except for the first, each subsequent learner is grown from previously grown learners. In simple words, weak learners are converted into strong ones. The AdaBoost algorithm works on the same principle as boosting with a slight difference. Let's discuss this difference in detail.

An ensemble is a composite model, combines a series of low performing classifiers with the aim of creating an improved classifier. Here, individual classifier vote and final prediction label returned that performs majority voting. Ensembles offer more accuracy than individual or base classifier. Ensemble methods can parallelize by allocating each base learner to different-different machines. Finally, you can say Ensemble learning methods are meta-algorithms that combine several machine learning methods into a single predictive model to increase performance. Ensemble methods can decrease variance using bagging approach, bias using a boosting approach, or improve predictions using stacking approach.

5.3CODE FOR MLP Classifier

```
# -*- coding: utf-8 -*-
"""
Created on Fri Aug 18

@author: akarsh
"""

import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split

df=pd.read_csv(r'G:\mini_project\data_fs.csv',index_col=0)


X=df.iloc[:,:-1]
y=df.iloc[:,-1]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=42)


from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

Scaled_X_train=scaler.fit_transform(X_train)
Scaled_X_test=scaler.transform(X_test)


parameter_space = {'random_state':[5,100],
                   'hidden_layer_sizes': [(10,30,10),(20,)],
                    'activation': ['tanh', 'relu'],
                    'solver': ['sgd', 'adam'],
                    'alpha': [0.0001, 0.05],
                    'learning_rate': ['constant','adaptive']}

from sklearn.neural_network import MLPClassifier
from sklearn.model_selection import GridSearchCV
snn_classifier = MLPClassifier()
clf = GridSearchCV(snn_classifier, parameter_space, n_jobs=-1, cv=5)
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)
```

```
# _________ACCURACIES__________________
import sklearn.metrics as metrics

print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
metrics.confusion_matrix(y_test, y_pred)

print("Accuracy:",metrics.accuracy_score(y_test, snn_predictions))
```

EXPLANATION:

The Artificial Neural Networks or shortly ANN's are widely used today in many applications and, classification is one of them and there are many libraries and frameworks that are dedicated to building Neural Networks with ease. Most of these frameworks and tools, however, require many lines of code to implement when compared to a simple library from Scikit-Learn that we are going to learn now.

The MLPClassifier stands for Multi-layer Perceptron classifier which in the name itself connects to a Neural Network. Unlike other classification algorithms such as Support Vectors or Naive Bayes Classifier, MLPClassifier relies on an underlying Neural Network to perform the task of classification. One similarity though, with Scikit-Learns other classification algorithms is that implementing MLPClassifier takes no more effort than implementing Support Vectors or Naive Bayes or any other classifiers from Scikit-Learn.

The multilayer perceptron (MLP) is a class of feedforward artificial neural network (ANN). The term MLP is used ambiguously, sometimes loosely to mean any feedforward ANN, sometimes strictly to refer to networks composed of multiple layers of perceptrons (with threshold activation); see § Terminology. Multilayer perceptrons are sometimes colloquially referred to as "vanilla" neural networks, especially when they have a single hidden layer.

5.4CODE FOR Random Forest Classifier

```
# -*- coding: utf-8 -*-
"""
Created on Fri Aug 18

@author: akarsh
"""

import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split

df=pd.read_csv(r'G:\mini_project\data_fs.csv',index_col=0)


X=df.iloc[:,:-1]
y=df.iloc[:,-1]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=42)


from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

Scaled_X_train=scaler.fit_transform(X_train)
Scaled_X_test=scaler.transform(X_test)


parameter_space = {'random_state':[5,100],
                   'hidden_layer_sizes': [(10,30,10),(20,)],
                    'activation': ['tanh', 'relu'],
                    'solver': ['sgd', 'adam'],
                    'alpha': [0.0001, 0.05],
                    'learning_rate': ['constant','adaptive']}

from sklearn.ensemble import RandomForestClassifier
from sklearn.datasets import make_classification
from sklearn.model_selection import GridSearchCV

snn_classifier = RandomForestClassifier(max_depth=2, random_state=0)
clf = GridSearchCV(snn_classifier, parameter_space, n_jobs=-1, cv=5)
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)
```

```
# _________ACCURACIES_____________________
import sklearn.metrics as metrics

print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
metrics.confusion_matrix(y_test, y_pred)

print("Accuracy:",metrics.accuracy_score(y_test, snn_predictions))
```

EXPLANATION:

The Random Forest is a flexible, easy to use machine learning algorithm that produces, even without hyper-parameter tuning, a great result most of the time. It is also one of the most used algorithms, because of its simplicity and diversity (it can be used for both classification and regression tasks). In this post we'll learn how the random forest algorithm works, how it differs from other algorithms and how to use it.

The Random Forest is a supervised learning algorithm. The "forest" it builds, is an ensemble of decision trees, usually trained with the “bagging” method. The general idea of the bagging method is that a combination of learning models increases the overall result. Put simply: random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction.

The One big advantage of random forest is that it can be used for both classification and regression problems, which form most current machine learning systems. Let's look at random forest in classification, since classification is sometimes considered the building block of machine learning.
Random forest has nearly the same hyperparameters as a decision tree or a bagging classifier. Fortunately, there's no need to combine a decision tree with a bagging classifier because you can easily use the classifier-class of random forest. With random forest, you can also deal with regression tasks by using the algorithm's regressor.

Random forest adds additional randomness to the model, while growing the trees. Instead of searching for the most important feature while splitting a node, it searches for the best feature among a random subset of features. This results in a wide diversity that generally results in a better model.

5.5CODE FOR Decision Tree Classifier

```
# -*- coding: utf-8 -*-
"""
Created on Fri Aug 18

@author: akarsh
"""

import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split

df=pd.read_csv(r'G:\mini_project\data_fs.csv',index_col=0)


X=df.iloc[:,:-1]
y=df.iloc[:,-1]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=42)


from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

Scaled_X_train=scaler.fit_transform(X_train)
Scaled_X_test=scaler.transform(X_test)


parameter_space = {'random_state':[5,100],
                   'hidden_layer_sizes': [(10,30,10),(20,)],
                    'activation': ['tanh', 'relu'],
                    'solver': ['sgd', 'adam'],
                    'alpha': [0.0001, 0.05],
                    'learning_rate': ['constant','adaptive']}

from sklearn.tree import DecisionTreeClassifier
from sklearn.datasets import make_classification
from sklearn.model_selection import GridSearchCV

snn_classifier = RandomForestClassifier(max_depth=2, random_state=0)
clf = GridSearchCV(snn_classifier, parameter_space, n_jobs=-1, cv=5)
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)
```

```
# _________ACCURACIES__________________
import sklearn.metrics as metrics

print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
metrics.confusion_matrix(y_test, y_pred)

print("Accuracy:",metrics.accuracy_score(y_test, snn_predictions))
```

EXPLANATION:

The Decision Trees are a type of Supervised Machine Learning (that is you explain what the input is and what the corresponding output is in the training data) where the data is continuously split according to a certain parameter. The tree can be explained by two entities, namely decision nodes and leaves. The leaves are the decisions or the final outcomes. And the decision nodes are where the data is split.

The Decision tree learning, or induction of decision trees is one of the predictive modelling approaches used in statistics, data mining and machine learning. It uses a decision tree (as a predictive model) to go from observations about an item (represented in the branches) to conclusions about the item's target value (represented in the leaves). Tree models where the target variable can take a discrete set of values are called classification trees; in these tree structures, leaves represent class labels and branches represent conjunctions of features that lead to those class labels. Decision trees where the target variable can take continuous values (typically real numbers) are called regression trees.

The A decision tree is a simple representation for classifying examples. For this section, assume that all the input features have finite discrete domains, and there is a single target feature called the "classification". Each element of the domain of the classification is called a class. A decision tree or a classification tree is a tree in which each internal (non-leaf) node is labeled with an input feature. The arcs coming from a node labeled with an input feature are labeled with each of the possible values of the target feature or the arc leads to a subordinate decision node on a different input feature. Each leaf of the tree is labeled with a class or a probability distribution over the classes, signifying that the data set has been classified by the tree into either a specific class, or into a particular probability distribution (which, if the decision tree is well-constructed, is skewed towards certain subsets of classes).

6. TESTING

It is very important to set the correct environment and use different physical objects to test the application in different scenes with different lighting in the case of conventional software development, models are made testable by using a test oracle, such as testers, test engineers or testing mechanisms working alongside the test program. An oracle can verify the outcome of a test against expected values. ML models, however, are often considered untestable because of the difficulty in performing black box testing on them. Since ML models output some sort of prediction, there are no expected values against which to verify test outcomes.

For lack of a test oracle, pseudo-oracles are used. Pseudo-oracles denote the scenario when the outputs for the given set of inputs are compared with each other, and correctness is determined.

For example, to solve a problem, a program has been coded using two different implementations; one of them will be considered the main program. The input passes through both implementations. If the output is the same as or proportional to (i.e., it falls within a given range around a predetermined value) the main program, then the program can be considered working as expected or correctly.

Testers must determine the f, x and y values before testing the models. If need be, they may consult with the engineers who are building the product, they may read and write documentation, they may rely on their own experience by making assumptions and they may research materials online. Through those methods, testers will arrive at possible experiments and tests (H) and expected results (O). They will also help find functional bugs.

Testers clarify the understanding of the f, x, and y values through every iteration of the testing process.

6.1TYPES OF TESTING

6.1.1Manual Testing

In manual testing the testers test the software manually and they test without the use of any automated tool. This is done to detect errors or any kind of problems in the software application. It is used to find some of the critical errors in the software applications. . Manual software testing requires more effort, but it is necessary to verify the feasibility of automation. Manual testing concepts do not require any knowledge of testing tools. One of the fundamental aspects of software testing is "100% automation is impossible."

6.1.2Automated testing

Automated testing or test automation is a software testing technology that uses special automated testing software tools to run sets of test cases. Rather, the manual test is performed by a person sitting at the computer who carefully performs the test steps. The automated test software tests and compare expected and actual results and generate detailed test reports. Software test automation requires a large investment of capital and resources. Continuous development cycles will require repeated execution of the same test suite. Using test automation tools, this set of tests can be recorded and played back as needed. Once the test suite is automated, no human intervention is required. This improves the return on investment for test automation. The goal of automation is to reduce the number of test cases that will be run manually, not to completely eliminate manual testing.

6.2Software Testing Methods

6.2.1Black Box Testing

This testing technique is used when there is no knowledge on the inner workings of the application. The evaluator has nothing to do with the system architecture and cannot access the source code. Generally, when running a black box test, the evaluator will interact with the user interface of the system by providing input and checking output without knowing how and where the input works.

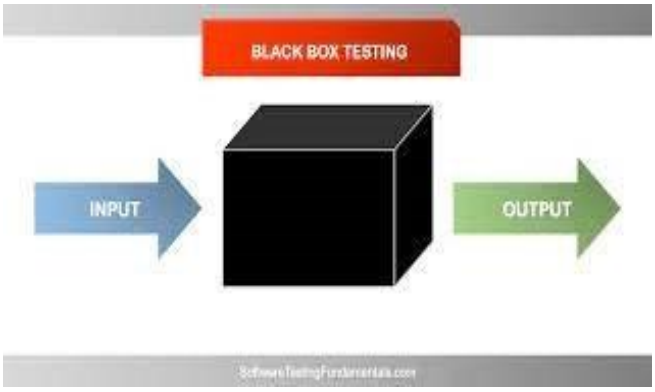

Fig 6.1.1: Black Box Testing

6.2.2Gray Box Testing

Gray box testing is a technique for testing applications with limited knowledge of the inner workings of the application. In software testing, it's important to know that the more you know, the better when testing applications. Mastery of a systematic field always gives the evaluator an advantage over those with limited knowledge in the field. In grey box testing the testers have the access to the databases. Using this information, the tester can prepare better test cases while making the testing plans.

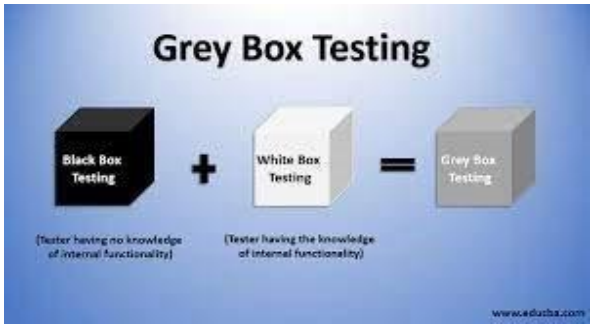

Fig 6.1.2: Grey Box Testing

6.2.3White Box Testing

White box testing is a software testing technique in which the internal structure, software's code and layout are tested to verify input and output processes and improve design, usability, and safety. In this type of testing the code is visible to the tester and that's why it is known as open box testing, Transparent box testing, Code-based testing and Glass box

testing.

It is one of the two parts of the software test box testing method. Corresponding black box testing involves testing from the perspective of the outside or the end user

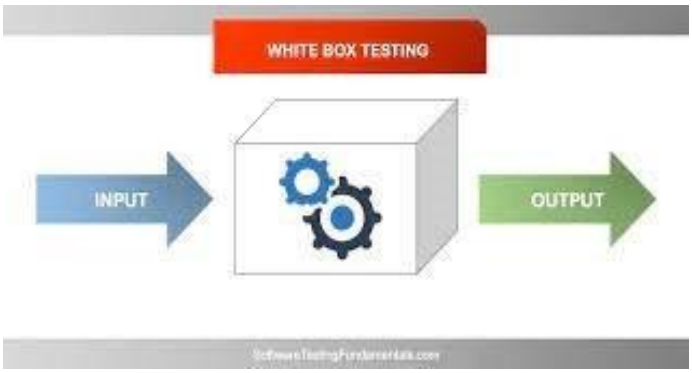

Fig 6.1.3: White Box Testing

6.3Testing Levels

6.3.1Non-functional Testing

The testing of the non-functional attributes is the non-functional testing. In this testing the programs non-functional specifications like user interface, stability, performance etc.

6.3.1.1Performance Testing

It is a non-functional testing technique which is used to access the quality of speed, stability and also the responsiveness of the application when the application is given a certain workload.

6.3.1.2Stress Testing

The software's reactions in an unnatural circumstance are monitored.

6.3.1.3Security Testing

In this the application is reviewed to detect any vulnerabilities, bugs or holes and what is the level of protection is also reviewed.

6.3.1.4Portability Testing

It is a process which states the ease with which we can move the software from one environment to another. Some of the attributes of this are co-existence and adaptability. This test helps us to identify the flaws that were not detected during the integration or the unit testing.

6.3.1.5Usability Testing

It is a technique which is used to identify the usability defects. This is done by making a small set of target end-users use the software and this testing focuses on the flexibility and the ease of use of the application.

6.3.2Functional Testing

This testing is used to validate the software system against the specifications. In this testing each function of the software is tested, and the output is verified against the functional requirements.

It mainly contains black box testing and checks if the errors are displayed according to the error conditions. To perform the functional testing firstly we need to understand the functional requirements of the software and then compute the results using input or test data then after executing the test cases we compare the actual and the computed results.

6.3.2.1Integration Testing

In this testing all the modules of the software are integrated logically and are tested as a group. Its main aim is to identify the defects in the interaction between these software modules when they are integrated.

6.3.2.2Regression Testing

Regression testing is just a selection of all or part of the test cases that have been executed, and re-execution to ensure that the existing functions work normally. This test is performed to ensure that new code changes will not have a secondary impact on existing functions. Make sure that the old code continues to work after making the latest code changes.

6.3.2.3Unit Testing

Unit testing is to test the accuracy of the isolated code. The white box testing is used for unit testing. It is often used to check of there are any issues in the code by developer himself and the main aim of this testing is to analyze and fix those defects.

6.3.2.4Alpha Testing

It is used to detect if there are any bugs before releasing the software to the public. It is used to discover the bugs which were not discovered during previous tests. It is called alpha testing because it is done when software is developed.

6.3.2.5Beta Testing

Beta testing is a user acceptability testing in which a product team distributes a nearly finished product to a set of target users in order to assess how well it performs in the real world.

6.4 TEST CASES

First, what are we trying to achieve when performing ML testing, as well as any software testing whatsoever? Quality assurance is required to make sure that the software system works according to the requirements. Were all the features implemented as agreed? Does the program behave as expected? All the parameters that you test the program against should be stated in the technical specification document.

Moreover, software testing has the power to point out all the defects and flaws during development. You don't want your clients to encounter bugs after the software is released and come to you waving their fists. Different kinds of testing allow us to catch bugs that are visible only during runtime.

However, in machine learning, a programmer usually inputs the data and the desired behavior, and the logic is elaborated by the machine. This is especially true for deep learning. Therefore, the purpose of machine learning testing is, first, to ensure that this learned logic will remain consistent, no matter how many times we call the programs.

Usually, software testing includes:

Unit tests. The program is broken down into blocks, and each element (unit) is tested separately.

Regression tests. They cover already tested software to see if it doesn't suddenly break.

Integration tests. This type of testing observes how multiple components of the program work together.

Moreover, there are certain rules that people follow: don't merge the code before it passes all the tests, always test newly introduced blocks of code, when fixing bugs, write a test that captures the bug.

Machine learning adds up more actions to your to-do list. You still need to follow ML's best practices. Moreover, every ML model needs not only to be tested but evaluated. Your model should generalize well. This is not what we usually understand by testing, but evaluation is needed to make sure that the performance is satisfactory.

First, you split the database into three non-overlapping sets. You use a training set to train the model. Then, to evaluate the performance of the model, you use two sets of data:

Validation set. Having only a training set and a testing set is not enough if you do many rounds of hyperparameter-tuning (which is always). And that can result in overfitting. To avoid that, you can select a small validation data set to evaluate a model. Only after you get maximum accuracy on the validation set, you make the testing set come into the game.

Test set (or holdout set). Your model might fit the training dataset perfectly well. But where are the guarantees that it will do equally well in real-life? In order to assure that, you select samples for a testing set from your training set — examples that the machine hasn't seen before. It is important to remain unbiased during selection and draw samples at random. Also, you should not use the same set many times to avoid training on your test data. Your test set should be large enough to provide statistically meaningful results and be representative of the data set as a whole

Cross-validation

Cross-validation is a model evaluation technique that can be performed even on a limited dataset. The training set is divided into small subsets, and the model is trained and validated on each of these samples.

K-fold cross-validation

The most common cross-validation method is called k-fold cross-validation. To use it, you need to divide the dataset into kk subsets (also called folds) and use them kk times. For example, by breaking the dataset into 10 subsets, you will perform a 10-fold cross-validation. Each subset must be used as the validation set at least once.

Leave-one-out cross-validation

In this method, we train the model on all the data samples in the set except for one data point that is used to test the model. By repeating this process iteratively, each time leaving a different data point as a testing set, you get to test the performance for all the data.

The benefit of the method is low bias since all the data points are used. However, it also leads to higher variation in testing because we are testing the model against just one data point each time.

Write ML unit tests

This kind of ML testing is more like traditional testing: you write and run tests checking the performance of the program. Applying the tests, you catch bugs in different components of the ML program. For example, you can test that the hidden layers in a neural network are configured correctly. If you are interested in diving deeper into unit testing for different models.

7. OUTPUT SCREENS/RESULTS

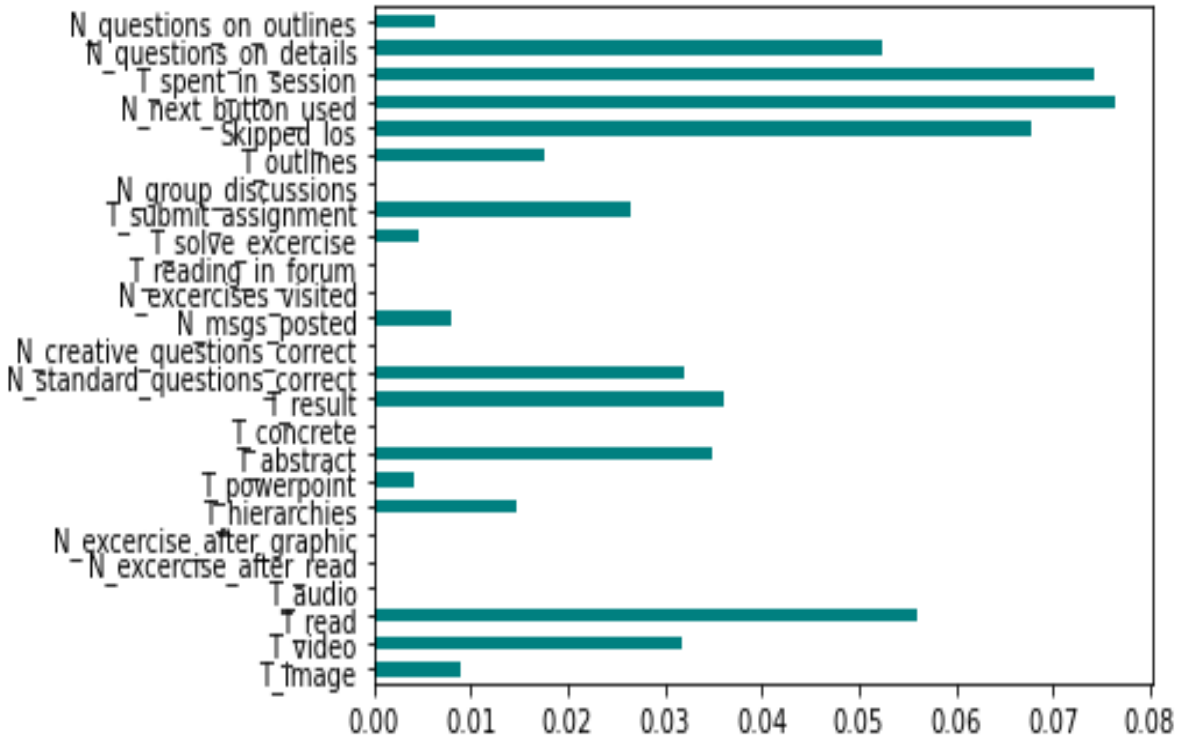

Fig 7.1: Fisher Scoring for all the features of the dataset

```
In [2]: runfile('G:/mini_project/clf.py', wdir='G:/mini_project')
Accuracy: 0.876

In [3]: metrics.confusion_matrix(y_test, y_pred)
Out[3]:
array([[117,   0,   0,   8],
       [  1,   0,   0,   0],
       [  3,   0,   0,   3],
       [ 16,   0,   0, 102]], dtype=int64)

In [4]:
```

Fig 7.2: Accuracy and Confusion Matrix for AdaBoost

TP: 219
Overall Accuracy: 87.6%

Class	n (truth)	n (classified)	Accuracy	Precision	Recall	F1 Score
1	137	125	88.8%	0.94	0.85	0.89
2	0	1	99.6%	0.0	0.0	0.0
3	0	6	97.6%	0.0	0.0	0.0
4	113	118	89.2%	0.86	0.90	0.88

Fig 7.3: Accuracy, Precision, Recall & F1Score for each class

	A	B	C	D	E
1	Models	Accuracy	Precision	Recall	F1Score
2	Decision Trees	80.13%	80%	80%	80%
3	Random Forest	80.15%	81%	80%	80%
4	Multi Layer Perceptron	83.2%	84%	83%	83.5%
5	AdaBoost Random Forest	87.6%	86%	87%	86.8%

Fig 7.4: Results for the different models tested during the research

8. CONCLUSION AND FURTHER WORK

We have attempted to identify attributes that can be used to detect multiple intelligences and attributes pertaining to the Felder-Silverman model of learning styles. Some of the attributes are time spent in learning through audio, the number of messages sent, and so on. Data was collected, and algorithms were run to validate the accuracy of classification. For the data collected to detect the dominating intelligence of the learner, the highest accuracy was achieved with AdaBoosted Random Forest. The lowest was with a decision tree, and that is 83.55%. For the data collected for the FSLM model, the accuracy in each dimension was calculated for the different algorithms. The highest accuracy was achieved when the Support Vector Machine was used in the input dimension. The consistency between manual marking and machine prediction was also calculated; the results are promising and are an improvement in this area.

This reduction and testing the same on a larger dataset. We also observed various patterns like change in learning styles of the learners according to the change in situations such as exam date nearing; change of learning style based on the difficulty of the concepts. We plan to use the transfer learning mode of machine learning to test the validity of the model proposed in the future by deploying the trained model of the algorithm to predict the learning style of the learners in a real-world environment. We also plan to generalize the patterns observed in various classes of learners. A limitation that we will overcome in the future is considering non-computer science learners learning online.

This reduction and testing the same on a larger dataset. We also observed various patterns like change in learning styles of the learners according to the change in situations such as exam date nearing; change of learning style based on the difficulty of the concepts. We plan to use the transfer learning mode of machine learning to test the validity of the model proposed in the future by deploying the trained model of the algorithm to predict the learning style of the learners in a real-world environment. We also plan to generalize the patterns observed in various classes of learners. A limitation that we will overcome in the future is considering non-computer science learners learning online.

9. REFERENCES

[1] Dr. Alkhuraiji, S., Cheetham, B., & Bamasak, O. (2011). Dynamic Adaptive Mechanism in Learning Management System Based on Learning Styles. In *2011 IEEE 11th International Conference on Advanced Learning Technologies* (pp. 215–217).

[2] Angeli, C., Valanides, N., Polemitou, E., & Fraggoulidou, E. (2016). An interaction effect between young children's field dependence-independence and order of learning with glass-box and black-box simulations: Evidence for the malleability of cognitive style in computer-supported learning. *Computers in Human Behavior, 61*, 569–583.

[3] Bernard, J., Chang, T. W., Popescu, E., & Graf, S. (2017). Learning style Identifier: Improving the precision of learning style identification through computational intelligence algorithms. *Expert Systems with Applications, 75*, 94–108

[4] Carmona, C., Castillo, G., & Mill´an, E. (2008). Designing a dynamic bayesian network for modeling students ' learning styles. *Eighth IEEE International Conference on Advanced Learning Technologies, 2008*, 346–350

[5] Carver, C. A., Howard, R. A., & Lane, W. D. (1999). Enhancing student learning through hypermedia courseware and incorporation of student learning styles. *IEEE Transactions on Education, 42*(1), 33–38

[6] Cassidy, S., & Eachus, P. (2000). Learning style, academic belief systems, self-report student proficiency and academic achievement in higher education. *Educational Psychology*, *20*(3), 307–322

[7] Crockett, K., Latham, A., & Whitton, N. (2017). On predicting learning styles in conversational intelligent tutoring systems using fuzzy decision trees. *International Journal of Human-Computer Studies, 97*, 98–115\

[8] Dr. Crockett, K., Latham, A., & Whitton, N. (2017). On predicting learning styles in conversational intelligent tutoring systems using fuzzy decision trees. *International Journal of Human-Computer Studies, 97*, 98–115

[9] Dr. García, P., Amandi, A., Schiaffino, S., & Campo, M. (2007). Evaluating Bayesian networks' precision for detecting students' learning styles. *Computers & Education, 49*(3), 794–80

[10] Julio Gardner, H., & Hatch, T. (1989). Intelligences multiple. *Educational Research, 18, Issue 8*

[11] Graf, S., Liu, T.-C., & Kinshuk.. (2010). Analysis of learners' navigational behaviour and their learning styles in an online course. *Journal of Computer Assisted Learning, 26*(2), 116–131

[12] Hmedna, B., Mezouary, A. El, & Baz, O. (2017). Identifying and tracking learning styles in MOOCs: A neural networks approach. *Advances in Intelligent Systems and Computing, 520*(2), 125–134

[13] Huang, E. Y., Lin, S. W., & Huang, T. K. (2012). What type of learning style leads to online participation in the mixed-mode e-learning environment? A study of software usage instruction. *Computers & Education, 58*(1), 338–349

[14] Jegatha Deborah, L., Baskaran, R., & Kannan, A. (2014). Learning styles assessment and theoretical origin in an E-learning scenario: A survey. *Artificial Intelligence Review, 42* (4), 801–819

[15] Kaewkiriya, T., Utakrit, N., & Tiantong, M. (2016). The design of a rule base for an e- Learning recommendation system based on multiple intelligences. *International Journal of Information and Education Technology, 6*(3), 206–210