A STUDY ON LEARNING STYLES

Summer Project Report Submitted

in partial fulfillment of the requirements for the award

of the Degree of **Master of Business Administration**

By

DUVVURU LOKESH

Reg.No.121901103

Project Guide

Mr. D. Krishnamoorthy B. Tech., MBA., UGC NET., (Ph.D)

Assistant Professor

Saveetha School of Management

SAVEETHA SCHOOL OF MANAGEMENT

SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES

2020

CERTIFICATE

This is to certify that the project work entitled "**A STUDY ON LEARNING STYLES**" is a Bonafide work done by Mr. **DUVVURU LOKESH** a student of 2019-21 Batch of MBA, in partial, fulfilment of the requirements for the, award of the Degree of **Master of Business Administration**. This is an original work done by the candidate under my supervision and guidance.

Project Guide

Dean

DECLARATAION

I, **DUVVURU LOKESH** (Regd. No.121901103) hereby declare that the project work entitled "A STUDY ON LEARNING STYLES" is a Bonafide work done by me in partial fulfilment of the requirements for the award of the degree of Master of Business Administration. This is my original work and it has not been previously formed the basis for the award of any other Degree, Diploma, Fellowship or any other similar title.

Place:

Date:

DUVVURU LOKESH

(121901103)

ACKNOWLEDGEMENT

I express my deep sense of gratitude to the management of Saveetha Institute of Medical and Technical Sciences for permitting me to undertake this project work.

I am extremely thankful to the Dean and Professor Dr. Prasanna Sivanandam, Saveetha School of Management, for her kind co-operation and support in completing my project work.

I am grateful to my project guide Mr. D. Krishnamoorthy, Assistant professor, for his co-operation and guidance in completing my project work.

I am thankful to other faculty members of SSM for their support in completion of the project work.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

As the education industry moving into new era of online learning, it is beneficial for both the students and instructors to classify students based on their learning styles as proposed by the Fleming and Bonwell (2009). They explain learning style is a term of reference to individual's preferred way of gathering, organizing and thinking about the information provided. VARK model is the preferred way to examine the individual's preference on taking in and giving out the information. Learning styles are different ways of learning. They include teaching and learning methods, unique to each individual that allows her/him to grasp more information. The idea of learning styles came into prominence in 1970s and it has been popular study by many social scientists and academicians.

Learning styles are affected by numerous variables, for example, individual experience, various insights and character factors, such as an inclination for learning alone or in a group. Our learning style will impact how we adapt to ordinary assignments throughout our life, for example, using a guide or preparing a dinner. A valuable guide to help comprehends this idea better is the means by which we figure out how to utilize another bit of technology. We can move toward it either by sitting alone, perusing directions from start to finish previously or take a 'hands on' approach like squeezing the various controls to find through experimentation or learn by observing others utilizing the equivalent. This model assists with reflecting about how learning inclinations shift among people. This said however, conditions may likewise decide how every individual discovers some new information. Such models help us to consider how we have inclinations for the way in which we learn. Hence, understanding learning styles approaches helps us to think about a person's prevailing or favored perspective thus helping us to learn better in lesser time as proposed by Sreenidhi and Helena (2017).

No student or instructor is limited to only one mode for communication input and output. Even in this way, it is fascinating to take note of that there are some prevailing inclinations and a few voids among different students and instructors. A few students and teacher lean toward not just a solid inclination for one specific mode yet additionally relative shortcomings in different modes. For taking in our surroundings we have a tendency to use our senses - sight, hearing, taste, bit and smell. In educational learning we have a tendency to typically use our sight, our speech and our hearing with less importance placed on style, bit and smell. Some learners like to use all their senses at once by experiencing their learning and this comes under kinesthetic preferences (Fleming & Bonwell,2009, pg.1).

As illustrated by Fleming and Bonwell (2009), importance of the VARK model is such that both students and teachers can utilize it to improve learning experience and teaching experience respectively. The acronym of VARK stands for Visual, Auditory, Read/Write and Kinesthetic. These are the four sensory modalities which help a learner to learn information. For instructors preferred sensory modalities of the students help them to adapt in the classroom or any platform used for teaching. In order to classify the large number of students based on their learning styles there are excellent possibilities by leveraging the growing information technology especially

predictive models from machine learning and deep learning. By using the predictive models there is possibility to build a user interface to classify learning styles.

Adaptive learning systems may use this knowledge to offer more precise personalization by identifying the learning patterns of students, leading to increased satisfaction and decreased learning time. In addition, students can directly benefit from a more precise recognition of learning styles, by being able to exploit their strengths in terms of learning styles, and by recognizing their limitations. In addition, teachers may use this knowledge about the learning style to offer more detailed guidance to students, which, as described before, becomes more useful for students as the recognition of the learning style becomes more detailed as well. In addition, in the same classroom, students with common learning styles will work together to enhance their learning experience and support the teachers with their techniques. Additionally, other stakeholders in the education ecosystem, such as teachers, administrators and parents, can make use of such an approach to improve education in general as proposed Gomede, Barros and Mendes (2020).

According to Gomede, Barros and Mendes (2020), computational variables related to the learning style can be adapted from various sources such as questionnaires, databases and registers of the educational institutions. Moreover, these variables give us the source of input of information for example diagrams, workshops, textbooks and recordings. Then the output is extracted as type of learning style such as Visual, Kinesthetic, Read/Write and Auditory respectively for the input. For this purpose, a computer intelligence tool can come in handy for both the students and instructors to determine the learning strategies and teaching methods respectively. As Li and Rahman (2018) points out although combined detection methods have been successful in the institutions, instructors have to use a lengthy questionnaire to collect the data from the students. This has been a huge drawback as it consumes more time and inefficient as it involves in both static and dynamic approaches.

In the favor of the above context, this study will explore the data mining process from the raw data collected from the college students through questionnaire which is an integral part before the analysis to determine possibility to use different descriptors. This study primarily aims on selecting a suitable predictive model that can be adapted by the educational institutions as a user interface. Each predictive model will be summarized using the different metrics such as precision, recall, f1 score and accuracy. After training the model with some part of the data it tested and evaluated on rest of the data. At the end confusion matrix is plotted for analysis. At the end best performing model will saved to build it as user interface.

1.1 THEORETICAL FRAMEWORK

A student can struggle in performing in the exams if his learning strategies does not align with his learning style or an instructor can fail to make his/her students to grasp the information he/she is teaching due improper understanding about the students' sensory modalities. Both the situation needs efficient learning strategy. This personalized learning strategy can be determined through a VARK model as proposed by Fleming and Bonwell (2009). As Li and Rahman (2018) illustrates the need for a dynamic model to classify the students learning style, there is a need for the tool that can interpret personal information and inputs from the student

and predict his/her learning style. Student's learning style helps instructor to improvise his/her teaching methods according to sensory modalities of a student.

1.1.1 DEFINITIONS

Keefe (1979) defines learning styles as "the characteristic cognitive, affective and physiological behaviors that serve as relatively stable indicators of how learners perceive, interact with and respond to the learning environment."

Dunn et al. (1978) defines learning styles as "the way in which each person absorbs and retains information and/or skills; regardless of how that process is described, it is dramatically different for each person".

Sims and Sims (1990, cited in Reid, 2002) put forward that learning styles are typical ways a person behaves, feels, and processes information in learning situations. Therefore, learning style is demonstrated in that pattern of behavior and performance by which an individual approaches educational experience.

Oxford et al. (1991) briefly defines the learning style as the general approaches' students used to learn a new subject or tackle a new problem.

Tan Dingliang (1995) defines learning styles as: "the way that a learner often adopts in the learning process, which includes the learning strategies that have been stabilized within a learner, the preference of some teaching stimuli and learning tendency."

Reid (1995) summarizes definitions of learning styles as internally based characteristics of individuals for the intake or understanding of new information. Essentially learning styles are based upon how a person perceives and processes information to facilitate learning.

Fleming and Bonwell (2009) "The term learning style is frequently used in schools, universities and colleges and there are a variety of books about it. A learning style refers to an individual's preferred ways of gathering, organizing, and thinking about information."

1.1.2 DIVISION OF LEARNING SYLES OR LEARNERS

1.1.2.1 VISUAL LEARNERS

According to Fleming and Bonwell (2009) visual learners prefer learning through seeing. They learn best by using diagrams, graphs, flow charts, and all the symbolic arrows, circles, hierarchies, and other tools that teachers use to represent what is presented in words. They use layout, title, pattern, design, and color are important to establish meaning. And they predominantly visualize context in their mind as illustrated by Sreenidhi and Helena (2017).

The main traits exhibited by visual learners as established by Hussain (2017) and Fleming and Bonwell (2009) are,

- Pay attention to body actions, such as gestures, facial expressions and corporal language.
- Visualize novel ideas through coding colors.
- Learning via sketches and other visual instruments that require a sense of sight.
- During learning, highlight essential points.

- Visual learners are more conscious of their immediate surroundings and their location in space with a clear visual preference.
- When reading, they appear to daydream, and they are likely to imagine what they are seeing.

Tools which are predominantly used by the instructors while teaching to the visual learners illustrated by Hussain (2017) are,

- Graphical content to boost their learning.
- The use of color coding to coordinate notes can be imperative when adding them.
- Highlighters in texts and other research materials to make the points prominent.
- Instead of writing down, draw ideas on the frame of mind by pictures.
- Drawing for clarification of definition.
- For visual learners, word searching, matching activities and puzzles can be successful.
- For teaching lessons, the use of visuals is important, i.e. photographs, charts, pictures and diagrams.

1.1.2.2 AUDITORY LEARNERS

According to Fleming and Bonwell (2009) Auditory learners use listening skills to learn best. A preference for information that is spoken or heard is characterized by this perceptual mode. Learners say that they learn best through conversation, oral input, email, cell phone chat, text, discussion boards, oral presentations, lectures, tutorials, and talking to other students and teachers via this modality. They have what they hear more obviously down pat than what they interpret and see. They learn information by listening to things. When they hear those things, they leap at the ideas and concepts explains Hussain (2017).

The main traits exhibited by Auditory learners as established by Hussain (2017) and Fleming and Bonwell (2009) are,

- Verbal inspiration diversifies their learning.
- Their customary leaning trend can be group research and reading the notes aloud.
- Rather than being told to look up something, they tend to explore and find out the concepts.
- Ask individuals for advice while they are disturbed.
- Using rhymes to recall stuff in a better way.
- Their learning ability can be enhanced by seminars, audio and speaking books.
- They relate to harmonic and rhythmical intelligence.

Tools which are predominantly used by the instructors while teaching to the Auditory learners illustrated by Hussain (2017) and Sreenidhi and Helena (2017) are,

- For their improved learning, grouping in small or larger amounts can be advantageous.
- A handy strategy can be encouragement to learn autonomously.
- Teaching skills by inculcating beats, themes, songs and rhythms into the curriculum.
- Encourage the idea of data.
- Arrange debate among learners.

- Let them have the right to submit questions and facilitate group discussion.
- Make audio streaming feasible.
- Activities such as brainstorming, jingles, songs, jokes and tales are welcomed.
- By using the speech recognition platform available on PCs, an auditory learner may also benefit from it.

1.1.2.3 READ/WRITE LEARNER

According to Fleming and Bonwell (2009) Read/Write learners use repetition of written words to learn information. This preference is for data that is shown either read or written as words. Typically, that means those who want textbooks. Not surprisingly, this modality has a clear preference for many scholars and high-achieving learners. Such students put emphasis on language consistency and are keen to use quotations, lists, texts, books, and manuals. Their reverence for words is high.

The main traits exhibited by Read/Write learners as established by Hussain (2017) and Fleming and Bonwell (2009) are,

- These students are PowerPoint adductors, quotes, dictionaries and internet adductors.
- Efficient at coordinating notes during class.
- Enjoy reading and composing.
- They belong to the conventional research group.
- Practice the textbook reading approach as a traditional learning mode.
- Perform well to prevent distraction in a pretty environment.
- Learn by interacting with what is written.

Tools which are predominantly used by the instructors while teaching to the Read/Write learners illustrated by Hussain (2017) are,

- Use of detailed presentations in PowerPoint.
- Usage of the mechanism of reading textbooks.
- Establish a note-writing habit.
- Promote the community of textbooks.
- Operations such as thinking, writing, presenting and doing need to be integrated.

1.1.2.4 KINESTHETIC LEARNER

According to Fleming and Bonwell (2009) kinesthetic learner either learns through practical learning or experiential learning. These type of learns always tend to experiment to grasp the information. kinesthetic learning means "Learning by doing," but that is an oversimplification that is often abstract, especially for school, college and university learning. For learners with a kinesthetic preference, it can still be made available. This mode uses multiple senses to take in their surroundings and to experience and learn new things (sight, touch, taste and smell). Some theorists claim that movement is necessary for this mode, but it is the fact that appeals most to the situation.

The main traits exhibited by Read/Write learners as established by Hussain (2017) and Fleming and Bonwell (2009) are,

- Strong students instead of being passive.
- Hands up, while ignorant of the responses.
- Using the pointing method when reading.
- Demonstrate motions while speaking.
- Also touching fellows to attract recognition, they prefer closeness with others.
- Remember stuff again and again by publishing.
- Using your hands to do stuff.
- They're organized and synchronized well.
- Have an inquisitive nature.
- Their key traits are acting, miming, performing and crafting.

Tools which are predominantly used by the instructors while teaching to the Kinesthetic learners illustrated by Hussain (2017) are,

- Organize events such as learning games, sculpture, painting and drawing.
- Inculcate drills, field trips and workshops.
- Using techniques for skimming, scraping and memorizing.
- Illustrate details by diagrams.
- Concentrate on practical work instead of theory.
- Using role play and simulation.
- Making then create spreadsheets, scale models and real-life projects.

1.1.2.3 AUTOMATIC DETECTION PROCESS

As Gomede, Barros and Mendes (2020) proposed that automatic process is divided into three stage problem. First collection of data through questionnaire to prepare a dataset with target variable. Second is to collect the information from learner's portal as descriptors. And at last

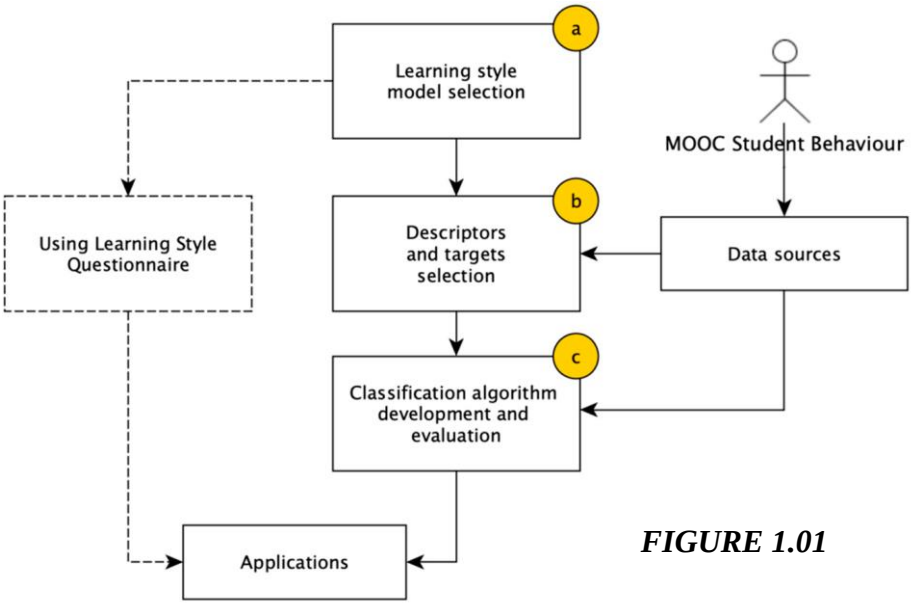


FIGURE 1.01

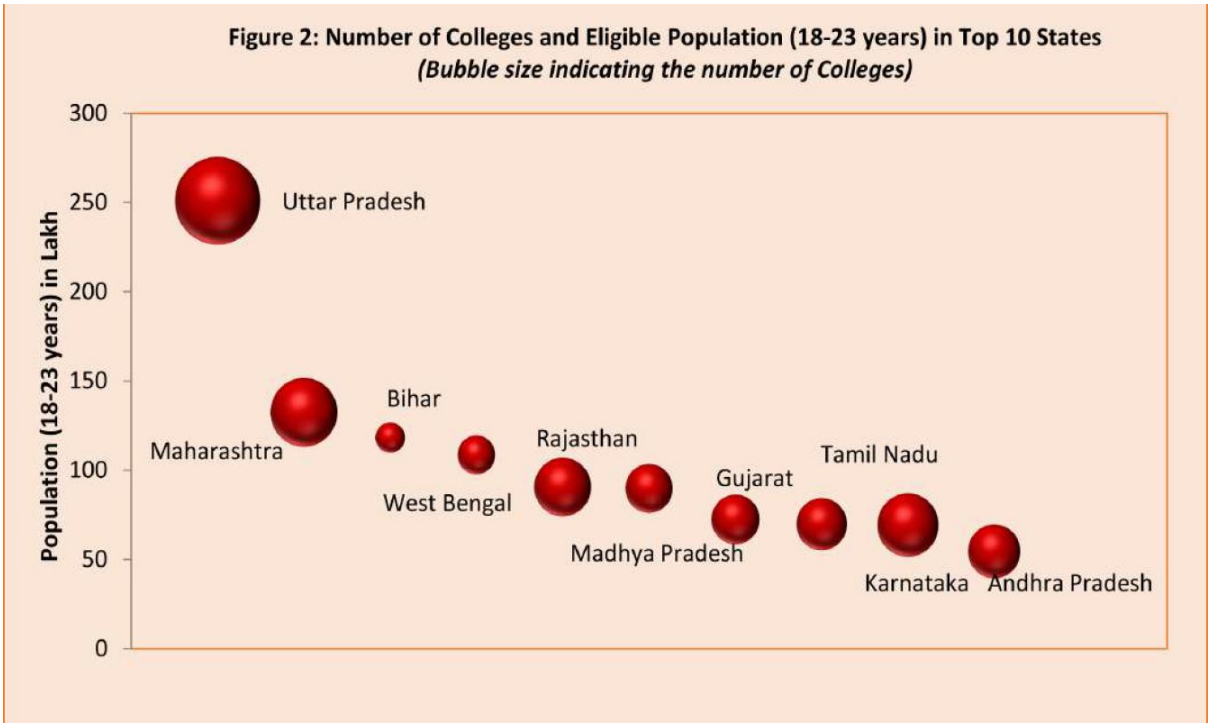
train the dataset and make predictions through a user application. As Li and Rahman (2018) illustrates with default learning styles, students join the method. When they communicate with the system, their learning styles will be modified differently based on their learning behavior.

1.2 INDUSTRY PROFILE

According to the MHRD Department of higher education, below points extensively encompasses the profile of the higher education industry in India.

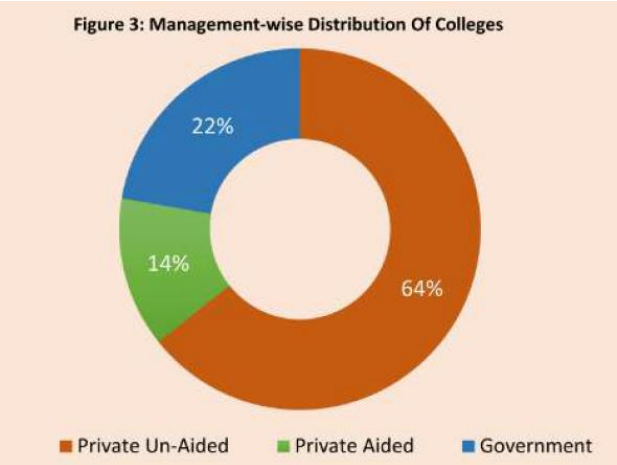
During the survey, 993 universities, 39931 colleges and 10725 stand-alone institutions were identified on the AISHE web portal, and 962 universities, 38179 colleges and 9190 stand-alone institutions responded. There are 298 affiliated institutions, i.e. colleges. There are 385 are privately owned universities. The rural area is home to 394 colleges. There are 16 women-only universities, 3 in Rajasthan, 2 in Tamil Nadu and 1 in Andhra Pradesh, Assam, Bihar, Delhi, Haryana, Himachal Pradesh, Karnataka, Maharashtra, Odisha, Uttarakhand and West Bengal respectively.

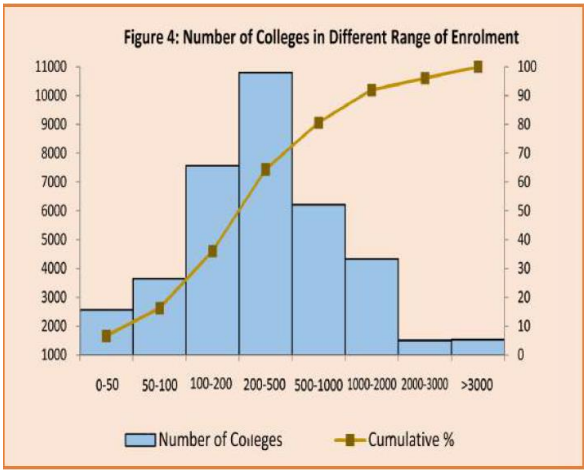
FIGURE 1.02



In addition to 1 Central Open University, 14 State Open Universities and 1 State Private Open University, 110 Dual Mode Universities also offer distance-based education and a maximum of 13 are located in Tamil Nadu. Other categories include 548 General, 142 Technological, 63 Agriculture & Allied, 58 Medical, 23 Law, 13 Sanskrit and 9 Language Universities, and 106 Universities. Uttar Pradesh, Maharashtra, Karnataka, Rajasthan, Haryana, Tamil Nadu, Gujarat and Madhya Pradesh are the top 8 states in terms of the largest number of colleges in India. With 880 colleges, Bangalore Urban district tops in terms of number of colleges, followed by Jaipur with 566 colleges. Around 32.2 per cent of colleges are in the top 50 districts. 77.8 percent of colleges are run privately; 64.3 percent are private-unassisted and 13.5 percent are

FIGURE 1.03

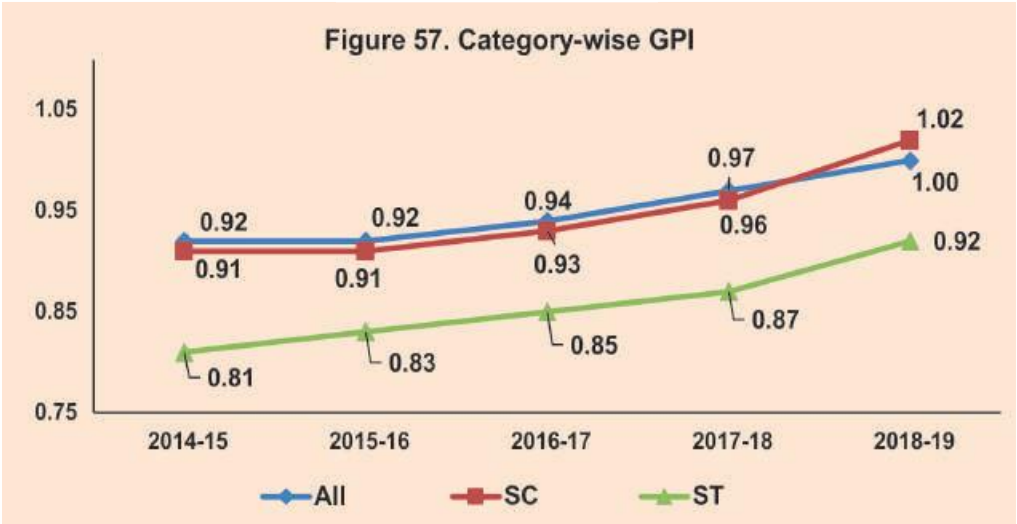




private-assisted. Andhra Pradesh & Uttar Pradesh have 88% of private-unaided colleges, and Tamil Nadu has 87% of private-unaided colleges, while Assam has 16.0%. 16.3% of the Colleges are having enrolment less than 100 and only 4% Colleges have enrolment more than 3000.

Total enrollment in higher education, with 19.2 million males and 18.2 million females, was estimated at 37.4 million. 48.6 percent of the overall enrollment is female. The Gross Enrolment Ratio (GER) in India's higher education is 26.3%, which is estimated for the age group of 18-23 years. For the male population, GER is 26.3 percent and 26.4 percent for females. It is 23 percent for Scheduled Castes and 17.2 percent for Scheduled Tribes, compared to 26.3 percent for the national GER. Around 10.62 percent of the overall enrolment in higher education is distance enrolment, of which 44.15 percent are female students. Around 79.8% of students are enrolled in the program at the undergraduate level. 1,69,170 Ph.D. students are enrolled, which is less than 0.5% of the maximum enrolment of students. Maximum numbers of Students are enrolled in B.A. programme followed by B.Sc. and B.Com. programmes. 10 Programmes out of approximately 187 cover 80.3% of the total students enrolled in higher education.

FIGURE 1.04



With 39.3 percent, the share of Group-C is the largest for non-teaching workers, led by Group-D with 28.3 percent. Group-A and Group-B account for 15.4% and 17% of non-teaching posts, respectively. For 100 male non-teaching workers, the total number of females is 49. With 23,765 males and 17,048 females, 40,813 students were awarded Ph.D. level degrees in 2018. M.A. at post graduate level. The cumulative pass number for students is followed by M.Sc. and M.B.A.

When it comes to monetary profile of the education industry in India According to the education sector report on Indian education, with nearly 29 per cent of India's population in the age group of 0 to 14 years, India's education market provides a tremendous potential. The higher education segment in India is projected to rise to Rs 2,44,824 crore by 2025 (US$ 35.03 billion). In the English Proficiency Index 2019, India was ranked 34 among the 100 nations. It is expected that growing internet penetration would aid in the distribution of education. Internet penetration in India crossed 54.29 percent as of December 2019.

For the introduction of capacity creation programmes, the central government plans to disburse US$ 1 billion to states. The 2015 Expertise India Project aims to educate 400 million Indian adolescents by 2022. There were 14,602 Vocational Training Institutes in India as of January 2020. More than one crore (10 million) young people annually have gained from the Capability India scheme. The government given Rs 400 crore (US$ 57.23 billion) to 'World Class Organizations' under the 2019-20 Union Budget. In the Times Higher Education (THE) Emerging Economies University Rankings 2020, 56 institutes from India were described.

India's education sector is also a strategic focus for the country. Since 2002, the government has permitted 100% Foreign Direct Investment (FDI) in the education sector through an automatic path. According to the data released by the Department for Industry Promotion and Internal Trade (DPIIT), overall FDI inflow in India's education sector stood at US$ 3.24 billion between April 2000 and March 2020. In the first six months of 2020, venture capital in ed-tech space soared from US$ 108 million in the same timeframe last year to US$ 795 million.

The National Education Policy 2020 was approved by the Cabinet chaired by Prime Minister Narendra Modi, making way for large-scale, structural reforms in both the school and higher education sectors. The strategy is focused on the key values of equality, opportunity, efficiency, affordability and sustainability and is consistent with the Sustainable Development Plan for 2030. The government has allocated Rs 59,845 crore (US$ 8.56 billion) to the Department of School Education and Literacy under the Union Budget 2020-21. In the Union Budget 2020-21, the Revitalization of Infrastructure and Programs in Education (RISE) by 2022 was declared with an outlay of Rs 3,000 crore (US$ 429.55 million).

After the digital revolution analytics industry has been making leaps and bound in India especially with the large population analytics industry is going to topple all the pen and paper implements in the country. According Khan, Shakil and Alam (2017) to the academic group has increasingly discussed the application of big data analytics in education and learning, which has led to new areas such as 'learning analytics' and 'big scholarly data analytics' emerging. Through incorporating social network analysis methods, much of the current learning analytics applications makes use of data linked to student experiences for analysis.

Several non-technical problems and drawbacks still remain when it comes to schooling. The growth of the system, unique to the Indian set-up, is influenced by the country's policies and administrative reforms. Furthermore, other concerns such as anonymity, financing and lack of knowledge surrounding the applicability of these structures in the actual scenario plague this area of research. Beyond this educational technology industry is booming at the rapid rate that is forcing to adapt technology-based learning in the higher educational institutions.

1.3 NEED FOR THE STUDY

As the curriculum of the higher education is leaning towards online mode, it is useful to determine what type of material a student needs to grasp the maximum information from the classes. Learning styles can provide the classification between the students. This classification can be further grouped on the basis of individual's learning style to provide the basis of the curriculum needed for each group of students.

Traditional practices of detecting the student's learning involves in preparing a lengthy questionnaire and asking subjects to fill them and classification is done manually. Major limitation of that process is that its time consuming. So, to overcome that limitation and make the process efficient there is need for efficient predictive model to classify learning styles. And filling the questionnaire can be limited to students with high understanding of the vocabulary that's present in the questions. To overcome this there is need for an application which can take the relevant answers can classify using the trained predictive model. To address the above needs this study is required.

1.4 OBJECTIVES OF THE STUDY

Objectives of this study are,

- To classify the students based on VARK learning styles based on their sensory modalities.
- Build a predictive model using the variables that can influence the learning styles.
- Explore the different models and to determine which can be more suitable for problem statement

1.5 LIMITATIONS

One of the limitations of this study is, training machine learning needs lot of pre labeled data. Predictive model learns the relationships in data using the dataset which is already classified, instead of classifying from the scratch. This leads to collection of data from two parts of the questionnaire. One part of questionnaire is to classify the students and second part is to use as the descriptors for the algorithm.

Other limitation of this study is unbalanced classes. When data is collected students are classified into 5 classes of learning styles. Each class should have to be almost equal for balanced learning. Machine learning algorithms also gives best results with more amount of data. In this case responses are limited to 396.

CHAPTER 2

REVIEW OF LITERATURE

1. Urval, Kamath, Ullal, Shenoy, Shenoy, and Udupa (2014) illustrates the influence of the gender and academic performance on the VARK learning styles among 415 medical students. They proved that there is no influence of the gender or previous academic performance on the type of learning styles. Most of the students in this study are classified as multimodal. Although most of the students came under Auditory and kinesthetic sensory modalities. They acknowledged their study helped in making the study materials better.

2. Zhang, Huang, Liu, Yin, Li, Yang and Xia (2020) proposed to define and classify the learning styles of students, by introducing a learning style classification method based on the deep belief network (DBN) for large-scale online education. They illustrate that DBLS provides better accuracy compared to traditional approaches.

3. Li and Rahman (2018) proposed a learning style detector using the Bayesian network using tree augmentation. Their experimental results proved that their method is more accurate than the results achieved using the Bayesian network. They proved that, if we take into account problems in the identification of learning styles, the proposed naive Bayesian tree augmented model helps us to discover the learning styles of students in a highly precise way.

4. Gomede, Barros and Mendes (2020) proposed a deep multi target prediction model that can classify Felder–Silverman learning styles. Their model consists of artificial neural network for the automatic detection of learning styles. The results obtained by them show that learning styles allow e-learning systems to improve the learning processes of students.

5. Khongpit, Sintanakul, and Nomphonkrang (2018) conducted a study on classification of the VARK learning styles among the computer science students. They illustrated that most of the computer science students around 53% are multimodal learners. In that most of the students fell in the category of bimodal of V and K. The study outcome shows that in order to enhance their motivation and understanding, the teaching material design should be achieved by designing imaginative activities and atmosphere in accordance with the learning abilities of the learners.

6. The meaning of the individualized "learning styles" that emerged in the 1970s is explained by Sreenidhi and Helena (2017). Many variables such as human knowledge, different intelligences, and personality factors such as a preference for learning alone or in a group affect learning styles. Our style of learning can affect how we perform daily tasks in our lives, such as reading a map, reading a book, a project plan, etc. They theorize individual’s learning style can make his/her life better.

7. Dantas and Cunha (2020) presents the debate on link between different learning styles from the kolb's model to Fleming's VARK model. They theorized based on their result selecting a particular learning style, in other words, relying on a single form of stimulus for the coordination of learning tasks as a presupposition for better learning, tends to be restrictive. While accepting individuality and the preferences of the subject, the most suitable approach for the construction of learning would be to provide students with various stimuli, similar to the different styles.

8. Maseleno, Hardaker, Sabani and Suhaili (2016) illustrated the multicultural perspective to the learning styles in higher education through diagnostic profiling. They created the data which consists of six sections which are culture, learning preferences, cognitive learning styles, creative skills, motivation and students back ground knowledge which can help in performing data analytics.

9. Labib, Pasina, Abdelhadi, Bayram and Nurunnabi (2019) experiments the Felder and Soloman's (1999) Index of Learning Styles (ILS) on the interior design students in Saudi Arabia. They introduce the clustering-based approach where they use agglomerative clustering algorithm and dendrogram to classify the students based on their personal data and preferences in learning styles. Their findings give 57% of students are classified as multiple learning styles.

10. Balasubramanian and Anouncia (2016) conducted a study which focuses specifically on developing a reinforcement model focused on the learners' cognitive skills (CS) for an integrated learning environment. The model approaches the problems in three ways; the first is to dynamically detect the Learning Style (LS) based on a learner's cognitive skills. The second emphasis is on mapping the Learning Object (LO) taxonomy of Bloom's cognitive abilities. The third objective is to build a reinforcement model to keep track of and provide input on the progress of the level of information competency.

11. Viloria, Gonzalez and Lezama (2019) research help students from various professions to understand the learning style preferences of college students and encourage teachers to go to college. After evaluating the learning preferences of students from different professions, the findings obtained led to the conclusion that college students have greater preferences for the reflector learning style followed by the analytical, pragmatist, and activist types of students from different professions. The overall findings indicate that in all examined occupations, the reflector learning style clearly predominates over all the others. Even if a particular learning style was not discovered, psychology, education and history students accept that the reflector style adopted by the pragmatist is favored, while journalism, humanities, and philosophy students prefer the reflector style, adopted by the theorist. The lower score was received by the activist learning style in all professions. They concluded by illustrating that optimum teaching methods should include the blend of different learning styles.

12. El aissaoui, El alami El madani, Oughdir and Allioui (2018) have suggested an approach to automatically classify the learning style based on the habits of the current learners and using techniques of online use mining and machine learning algorithms. The techniques of web usage mining were used to pre-process the log file collected from the E-learning environment and collect the sequences of the learners. In order to group them into 16 learning style combinations based on the Felder and Silverman learning style model, the captured learners' sequences were provided as an input to the K-modes clustering algorithm. Then, to predict a student's learning style in real time, the naive Bayes classifier was used.

13. V. Kolekar, M. Pai and Pai M.M (2018) presented approach to identify Felder and Silverman learning style. Using the Moodle platform, an e-learning application was created with the functionality to collect learners' usage data. The use data is used to group the learners according to FSLSM learning categories. On the portal, customization is created by designing an adaptive user interface for each learner based on the FSLSM learning style. Their portal recognizes the learning style of the students and then offers content and customizes the User Interface (UI) based on that learning style.

14. Bajaj and Sharma (2018) have built a system to help make adaptive education easily open to a diverse student audience in the future, through various cultural backgrounds, geographies and modes of education, conventional or eLearning. The structure offers a set of various attributes of student learning that can be monitored, based on which personalized learning can be delivered. Within a single tool, readily available collections would make it easy to determine which attributes of student learning can be selected for a specific learning environment. That would make it easier for the implementer to narrow down on potential models of learning theory that can be used to impart adaptive learning. This is the first system ever produced to compare various theories of learning and to compare classification strategies based on artificial intelligence based on the output of developed models. These models are dynamically constructed within the same tool, and statistical evaluation helps to define the most appropriate model for implementation in a given learning environment. The chosen learning theory and method of artificial intelligence can then be used to evaluate students ' learning styles. The framework suggests that a virtual teacher should be hosted in a cloud environment that interacts with learners in a scalable way to dynamically determine their learning styles using natural language processing APIs. Various learning content providers, either conventional schools or e-learning portals, will subsequently use the determined learning styles of students to provide adaptive education.

15. Švarcováa and Jelínkováa (2016) presents the results of the pilot portion of research aimed at defining learning styles among selected university students. The views on learning styles vary in many respects, as do the points of view themselves. Nevertheless, most opinions highlight the need for specific students and pupils to

respect and accept learning styles. The styles of learning are deeply rooted in the biological conditionality of the individual, so they are not easy to alter. They conclude stating that the pieces of knowledge need to be tailored to the teaching process so that the learning outcomes are as successful as possible.

16. Omar, Mohamad and Paimin (2014) researched the mixture of active, auditory, visual and sequential learning styles that students tend to learn in electrical engineering. The study also shows that students of electrical engineering at the polytechnic tend to learn socially, work in groups comfortably, and prefer to provide their friends with information, and be able to easily remember things. Research also shows that respondents selected the sensing pattern because engineering students were concerned with reasonable work accuracy. Overall, learning styles do not have an important correlation with academic achievement, but they can be used to describe the pattern of student-owned learning styles and can also be used by teachers to develop teaching methods.

17. Yee, Yunosb, Othmanc, Hassand, Tee and Mohamad (2015) examined the difference in learning styles among technical students at the level of higher order thinking skills (HOTS). A total of 375 technical students from four Malaysian technical universities were selected at random as samples. The Kolb Learning Styles Inventory and a set of questionnaires were used as research tools adapted from Marzano Rubrics for Specific Tasks or Situations. This is a quantitative analysis, and using SPSS software, the collected information was analyzed. The results showed that Doer was the most dominant learning style among technical students. The results also show that none of the students perceived the level of their thinking skills to be high. Only four Marzano HOTS are rated at the moderate level, namely, comparing, inductive reasoning, deductive reasoning and investigation. On contrary, nine Marzano HOTS are rated as poor. The study of Cramer V revealed that there is a very poor association between Kolb Learning Styles and the level of 13 Marzano HOTS. In addition, the results revealed that at the level of 13 Marzano HOTS, there is a statistically meaningful variation in Kolb Learning Styles. But, in Kolb Learning Styles, only two Marzano HOTS differ substantially. The recognition of learner learning patterns could also serve as an initial reference in building a more productive and advantageous teaching-learning environment for HOTS learning.

18. Maric, Penger, Todorovic, Djurica and Pintar (2015) carried out questionnaire research and included questions about the students and their styles of learning. To measure the different learning styles (reflectors, theorists, activists, pragmatists) across three Slovenian universities, they used the Honey and Mumford's learning style questionnaire (LSQ) to search for the learning style that prevails in each university and present the differences between them. They have found prevailing learning styles and the differences between them through the use of the LSQ questionnaire at the three Slovenian universities. Therefore, behavioral patterns should be more related to the ability of learners and their educational abilities in the future. Learning techniques are

often critical when students adapt to various forms of instructional processes at each university and faculty that are distinct.

19. Carol (2014) presents a paper that attempts to examine learning styles in Higher Education at the level of first year students. The target population selected is characterized as a comparative cohort by the first-year history students at the University of Bucharest, and a group of MA students from separate research strands. Kolb's Learning Patterns Questionnaire was the applied tool. Some uniformity of learning styles appears to suggest the main outcomes, but some essential variations are still evident. At the same time, the consequences for the education in historical history are discussed.

20. Klement (2014), based on the review of the collected study results, grouped students into four classes according to their preferred style of learning offered by the VARK classification. The main group of students consists of those who favors the form of motion (kinesthetic) learning, and the other three types of learning are described nearly equally. Within teaching planning, it would be advisable to respect this fact, which should then include and be enhanced by more elements or instructional resources that would allow a certain category of students to learn more effectively. He concluded by stating that among the male group of students is predominantly kinesthetic.

21. Mamat and Yusof (2013) discussed the implementation of teaching styles in an e-learning environment. The relation between the style of learning and the adaptive system is identified and linked. In order to facilitate the Customized Shared Learning (PCL) online learning environment, elements obtained from conventional learning style studies are tailored to the e-learning framework. PCL has integrated elements of Human Computer Interaction (HCI) into its environment. The learner model is smaller in terms of learning style, needs of learners and references, so it should be expanded to integrate more customised features into an adaptive system. Therefore, the sharing of knowledge / materials, questioning well formation and option for freedom support the design of depth. Therefore, the sharing of knowledge/materials, questioning well formation and freedom option support the design of deep customization into an adaptive system such as iYu to balance between Personalized Learning Environment (PLE) and collaborative learning. In addition, ideas of duplication, generalisation and time constraints are among the reasons that may cause the efficacy of the e-learning method to collapse. Furthermore, during a learning process or self-reflection event in the PCL framework, the learning style of learners is found to be dynamic. The results showed that, with the presentation of the iYu user interface, learners were able to reflect and discover themselves. In addition, the iYu system must also be checked with other course learners to distinguish the outcomes of data collection based on demographic context in order to preserve and optimise the efficiency of the iYu system and its efficacy.

22. Koh and Chua (2012) look at the variations in learning styles between students in mechanical engineering from various institutions and ages. Their paper provides assessments of the types of learning styles among engineering students in conjunction with this. In the study, most engineering students (68.44 percent) have a visual and visual-related learning style, with 51.23 percent of students having a visual learning style. In contrast, kinesthetic or kinesthetic-related learning styles are owned by a minority of them (15.98 percent), which is consistent with the figure reported by Koh (2008), leading to the suggestion that engineering students who have a kinesthetic or kinesthetic-related learning style range from 10 % to 25% of the population. Higher-level students (BEng) showed a single style of learning (80.55 percent for INTI), compared to graduate students (61.55 percent). The distribution of learning styles between the three universities is very close, reflecting the distribution of learning styles for students of mechanical engineering.

23. Othman and Amiruddin (2010) discusses the advantages of the VARK learning styles. They stated that the type of VARK learning style does not require intelligence or innate abilities, but is closely connected to how information or new knowledge is learned or understood. The type of VARK learning can also be viewed as a human methodology used to gain expertise, constructive skills and attitudes. As such, the style of VARK learning will create a interesting learning experience for learners and activate the senses of learners in learning.

24. Stirling and Alquraini (2017) undertook a study that seeks to develop understanding of Saudi nursing students' favourite learning styles that can help students to appreciate the quality of the course and, in turn, provide better patient care. 125 women nursing students who volunteered to participate in this research were administered a cross-sectional survey design. The majority of participants (80.5 percent) had some preference for kinesthetic learning. Of those with a dominant preference, 38.2 percent have a clear preference for kinesthetic learning, while 10.6 percent, 4.9 percent, and 2.4 percent favoured aural, reading / writing, and visual learning, respectively. In their kinesthetic preference, the learning styles of Saudi nursing learners were not significantly different from one group of Australian nursing students (p 1⁄4 0.85), but were significantly different from Saudi medical students in their kinesthetic preference ($p < 0.0001$). The kinesthetic learning style was the highest ranked preference for all groups of nursing students. Saudi nursing students' reported learning styles were more similar to other groups of nurses in key areas of learning preference than they were to other Saudi healthcare students.

CHAPTER 3
RESEARCH METHODOLOGY

Research methodology is the particular methods or strategies used to define, pick, process and interpret information on the subject. In a research paper, the methodology section helps the reader to objectively determine the general feasibility and reliability of the report. More specifically, analysis techniques allow one to find a solution to the problem Research methodology is a means to address the research dilemma systematically. It's a science of learning how experimental research is conducted. Essentially, this is the method by which researchers carry out their work of identifying, analyzing and forecasting the phenomena. It is supposed to set down the study work plan.

3.1 RESEARCH DESIGN

In this study exploratory research is adapted as the nature of the problem statement is to investigate different predictive models which can be adapted for the classification of the learning styles. This research also explores the entire process of building an automatic learning style detection model and to build a dataset for training and testing the learning style detector. This research also aims to explore the possibility of deploying the best predictive model as a user interface.

3.2 DATA COLLECTION

In this study data is collected as per the dataset collection process proposed by Gomede, Barros and Mendes (2020). In this study data collected is primarily from the fulltime college students. The same process proposes by Gomede, Barros and Mendes (2020) is adapted according to the objective of this study.

Data consists of three parts primary VARK data, demographic data and student preferences data. The dataset preparation process is explained in the diagram below.

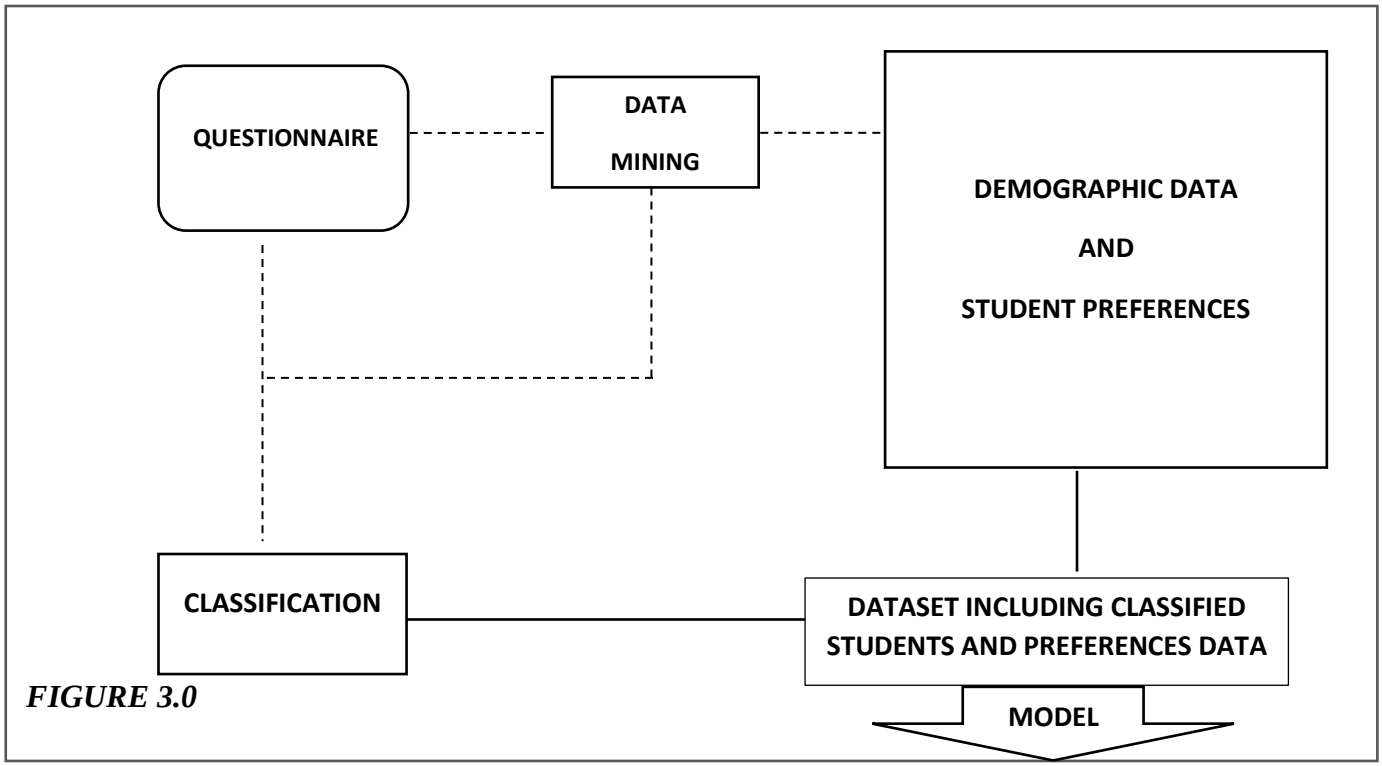


FIGURE 3.0

As shown in the diagram after students fill the questionnaire, primary section which consists of standard questions based on VARK version 8.01 questionnaire and student preferences after data mining is combined to classify the learning styles. Demographic and student preferences information is preprocessed and used form a dataset along with the classified learning style.

3.3 DATA COLLECTION INSTRUMENT

All the data collected in this study is using google forms as data collection instrument. Data is shared personally to participants through contacts.

3.4 SAMPLING

The data collection is done through the snowball sampling. Snowball sampling is where research participant recruits other participants for a study. It is used where potential participants are hard to find. In this case initially five participants are selected and those five participants are asked to select five participants each and so on. In this study this type of sampling can help include students from all backgrounds and multiple locations which in turn can help build a versatile predictive model.

3.5 QUESTIONNAIRE DESIGN

Questionnaire is the instrument that logs in the data to initiate an experiment. This study uses the questionnaire to capture the classification data ad well as the descriptors. As Gomede, Barros and Mendes (2020) suggested, Questionnaire consists of three parts primary VARK 8.0 questions, demographic questions and student preferences questions in the form of scenarios.

Demographic variables include name, mail id and college name which corresponds to each student. Then possible demographic descriptors like age, gender, college location, educational designation, educational stream, place grownup during schooling, school region, school type and schoolboard.

VARK 8.0 questionnaire from VARK-Learn Limited is included by altering as needed for this study. This part of the questionnaire is for classification and to form the target variable in the dataset with the student preferences. This part of the data is discarded after classification. The last part of the questionnaire consists of descriptor variables that gives weightage for visual, auditory, read/write and kinesthetic attributes of each learner. This part of the data is taken in the form of Likert scale and assigned weight as per attributes. These weightages serve as numerical descriptors in the dataset for determining the learning style.

3.6 TOOLS FOR ANALYSIS

As this study proposes a predictive model for learning style classification, understanding the data and performing exploratory data analysis is the important part in choosing the best model. For exploratory data analysis libraries like pandas, NumPy, seaborn and matplotlib are used with python programming language in jupyter notebook.

For statistical tests statsmodels and SciPy is imported in the python notebook to adapt to the python environment. For the algorithms and modeling analysis sklearn library is predominantly used in this study. Predictive models are imported from the sklearn, keras, XGBoost, Light GBM and TensorFlow libraries and frameworks.

3.6.1 LIBRARIES

3.6.1.1 PANDAS

Pandas is a fast, efficient, modular and easy-to-use open source data analysis and manipulation framework, developed on top of the Python programming language. This tool is used for most of the dataset preparation purpose and to frame tables in to analyze the dataset.

3.6.1.2 NUMPY

This library is dedicated for the operation and computation of the high multidimensional arrays. In this study a dataset is formed which is a multidimensional matrix. Numpy is used to solves the numerical operation on complex dataset.

3.6.1.3 MATPLOTLIB AND SEABORN

Matplotlib and seaborn are the data visualization tools which helps plot the data during the univariate, bivariate and multivariate analysis. These plots are used parallel to the frequency and mean analysis. And these libraries help in plotting the graphs which corresponds to the relationships and interactions between the variables in the dataset.

3.6.1.4 SKLEARN

For model selection metrics from the sklearn library accuracy, precision, recall and f1 scores are imported. These metrics are used for analyzing each predictive model.

3.6.2 STATISTICAL TOOLS

In this study statistical tools are used to check the interactions between the variables and to evaluate the models. The tools used to study the data and the interaction between the variables are,

1. Chi-square Test
2. One-way ANOVA
3. Five-point summary
4. Correlation summary
5. Frequency analysis

CHI-SQUARE TEST

Chi square test is used to examine the relationship between two categorical variables. In this study target variable is compared with the demographic variables to draw conclusion on interaction between learning styles and demographic variables like age, gender, college location, educational designation, educational stream, place grownup during schooling, school region, school type and schoolboard.

ONE-WAY ANOVA

ANOVA stands for "Variance Analysis" and is an omnibus measure, which means that it measures the average variation of both groups. One-way ANOVA, also referred to as one ANOVA factor, is a parametric measure used to measure for a statistically meaningful difference in result between 3 or more categories. In this study learning is tested with numerical descriptors of student preferences.

FIVE-POINT SUMMARY

FIVE POINT SUMMARY is used to display some descriptive statistics elements such as percentile, mean, standard deviation, quartiles, minimum and maximum values of each numerical variable of the data. In this study it is used for the mean analysis.

CORRELATION SUMMARY

CORRELATION SUMMARY is used to display the pairwise correlation of all numerical elements in the dataset. In this study this summary is used to examine the correlation between the numerical variables.

FREQUENCY ANALYSIS

Frequency of classes in the categorical variables, percentage of frequency and cumulative percentage is tabulated for the frequency analysis. Frequency analysis is a part of univariate analysis aiding in understanding the dataset better.

3.6.3 VISUALIZATION TOOLS

To perform the univariate analysis histograms are used to determine the skewness in the data and normalization. Bar charts are used to assist the frequency tables to analyses the categorical variables.

For bivariate analysis lmplot which shows the scatter plot with regression line. This plot helps to determine the linear relation between the numerical descriptors of same kind. To aid the relationship between the categorical variables, categorical plots are used in this study.

Pair plot is used to perform the multivariate analysis. Pair plot gives all the linear relationships plots and gives histograms in the diagonal. Along with pair plot, correlation chart is used to aid correlation summary in multivariate analysis.

3.6.4 MODEL ANALYSIS AND EVALUATION TOOLS

For the evaluation of the classification model confusion matrix metrics are most commonly used. In this study we look at five metrics to evaluate a model.

1. Accuracy
2. Precision
3. Recall
4. f1 score
5. cross validation score

ACCURACY

Accuracy is the most intuitive success indicator which is essentially the ratio of correctly expected observation to overall observations. One might assume that, if we have good precision, our model is the best one. Yeah, consistency is a fantastic indicator, but only because you have symmetric datasets where false positive and false negative values are about the same. You therefore need to look at other metrics to determine the efficiency of your model.

Accuracy = **TP+TN/TP+FP+FN+TN**

PRECISION

Precision is the percentage of the positive observations correctly predicted to the overall positive observations predicted. High precision is related to the low false positive rate.

$$\text{Precision} = \mathbf{TP/TP+FP}$$

RECALL

Recall is the ratio of correctly predicted positive observations to the all observations in actual class predictions. Recall is all called as sensitivity.

$$\text{Recall} = \mathbf{TP/TP+FN}$$

F1 SCORE

F1 score is the weighted average for Accuracy and Recall. Therefore, this score takes into account both false positive and false negatives. Intuitively, it's not as easy to understand as accuracy, but F1 is typically more useful than accuracy, particularly if you have an unequal class distribution. Accuracy performs better if false positives and false negatives have equal costs. If the cost of false positives and false negatives is somewhat different, it is best to look at both Accuracy and Recall.

$$\text{F1 Score} = 2*\mathbf{(Recall * Precision) / (Recall + Precision)}$$

CROSS VALIDATION SCORE

In this study stratified k-fold is used and with five folds to get average accuracy after training and testing for five runs. Stratified k-fold cross-validation is the same as k-fold cross-validation, but in stratified k-fold cross-validation, stratified sampling is performed instead of random sampling.

In the above formulae TP refers to true positives, FP refers to False positives, TN refers to true negatives and FN refers to false negatives.

TRUE POSITIVES (TP)

These are the positive values accurately predicted, which means that the value of the real class is yes and the value of the predicted class is yes.

TRUE NEGATIVES (TN)

These are the accurately predicted negative values, which means that the value of the real class is no and the value of the predicted class is no.

FALSE POSITIVES (FP)

When actual class is no and predicted class is yes.

FALSE NEGATIVES (FN)

When actual class is yes but predicted class in no.

False positives and false negatives exist when the real class is in contrast with the expected class.

CHAPTER 4

DATA ANALYSIS

4.1 FREQUENCY ANALYSIS

4.1.1 ANALYSIS OF AGE

TABLE 4.01 AGE

Age	frequency	percentage	cum percentage
20 to 25	225	56.82%	56.82%
25 to 30	92	23.23%	80.05%
<20	61	15.40%	95.45%
30+	18	4.55%	100.00%

INTERPRETATION

Table 4.01 shows the frequency analysis of the age. From the table it is clear that majority of respondents are age group 20 to 25 (225) followed by age group 25 to 30 (92), age group less than 20 (61) and age group greater than 30 (18).

4.1.2 ANALYSIS OF GENDER

TABLE 4.02 GENDER

Gender	frequency	percentage	cum percentage
Female	207	52.30%	52.27%
Male	189	47.70%	100.00%

INTERPRETATION

Table 4.02 shows the frequency analysis of the gender. From the table it is clear that majority of the respondents are female (207) and male (189).

4.1.3 ANALYSIS OF EDUCATIONAL DESIGNATION

TABLE 4.03 EDUCATIONAL DESIGNATION

Educational Designation	frequency	percentage	Cum percentage
Under Graduation	234	59.10%	59.09%
Post-Graduation	158	39.90%	98.99%
Doctorial	4	1.00%	100.00%

INTERPRETATION

Table 4.03 shows the frequency analysis of the educational designation. From the table it is clear that majority of the students are under graduates (234) followed by post graduates (158) and doctorial students.

4.1.4 ANALYSIS OF EDUCATIONAL STREAM

TABLE 4.04 EDUCATIONAL STREAM

Educational Stream	frequency	percentage	Cum percentage
Engineering	204	51.50%	51.52%
Management	57	14.40%	65.91%
Arts	54	13.60%	79.55%
sciences	26	6.60%	86.11%
medicine	23	5.80%	91.92%
others	10	2.50%	94.44%
computer science	6	1.50%	95.96%
commerce	5	1.30%	97.22%
law	3	0.80%	97.98%
Mathematics	2	0.50%	98.48%
Finance	2	0.50%	98.99%
statistics	1	0.30%	99.24%
economics	1	0.30%	99.49%
Accountancy	1	0.30%	99.75%
humanities	1	0.30%	100.00%

INTERPRETATION

Table 4.04 shows the frequency analysis of the educational stream. From the table it is clear that majority if the respondents are studying engineering (204) followed by management (57), arts (54), sciences (26) and medicine (23). Rest of the educational streams are less than 6 percent in the whole respondent pool. It is inferred that majority of the respondents are of engineering background.

4.1.5 ANALYSIS OF PLACE GROWNUP IN

TABLE 4.05 PLACE GROWN UP IN

Place Grow up in	frequency	percentage	Cum percentage
City	194	49.00%	48.99%
Town	85	21.50%	70.45%
Rural	63	15.90%	86.36%
City;Rural	22	5.60%	91.92%
City;Town	15	3.80%	95.71%
City;Rural;Town	9	2.30%	97.98%
Rural;Town	8	2.00%	100.00%

INTERPRETATION

Table 4.05 shows the frequency analysis of place respondent grown up in. From this table it is clear that majority of the respondents grown up in city (194). It is followed by people grown up in town (85) and rural (63). It is inferred that majority of respondents grew up in city.

4.1.6 ANALYSIS OF SCHOOL REGION

TABLE 4.06 SCHOOL REGION

School region	frequency	percentage	Cum percentage
City	183	46.20%	46.21%
Town	106	26.80%	72.98%
Rural	76	19.20%	92.17%
City;Town	10	2.50%	94.70%
City;Rural	10	2.50%	97.22%
Rural;Town	9	2.30%	99.49%
City;Rural;Town	2	0.50%	100.00%

INTERPRETATION

Table 4.06 shows the frequency analysis of school region. From this table it is clear that majority of the respondents' school region is in city (183). It is followed by town (106) and rural (76). It is inferred that majority of respondents' school region is in city.

4.1.7 ANALYSIS OF SCHOOL TYPE

TABLE 4.07 SCHOOL TYPE

School type	frequency	percentage	Cum percentage
Private	300	75.80%	75.76%
Govt	83	21.00%	96.72%
Other	13	3.30%	100.00%

INTERPRETATION

Table 4.07 shows the frequency analysis of school type. From this table it is evident that majority of respondents are of school type private (300) and followed by govt (83) and other (13).

4.1.8 ANALYSIS OF SCHOOL BOARD

TABLE 4.07 SCHOOL BOARD

School board	frequency	percentage	Cum percentage
State Board	296	74.70%	74.75%
Central Board	93	23.50%	98.23%
Other	7	1.80%	100.00%

INTERPRETATION

Table 4.08 shows the frequency analysis of school board. From this table it is evident that majority of respondents are of state board (296) and followed by central board (93) and other (7).

4.1.9 ANALYSIS OF STYLE (INDEPENDENT VARIABLE)

TABLE 4.09 STYLE

Style	frequency	percentage	Cum percentage
A	136	34.30%	34.34%
K	131	33.10%	67.42%
Multi	55	13.90%	81.31%
V	45	11.40%	92.68%
R	29	7.30%	100.00%

INTERPRETATION

Table 4.09 shows the frequency analysis of the style variable extracted from first part of questionnaire. From table it is clearly evident that majority of respondents prefer A (136) and followed by K (131), Multi (55), V (45) and R (29). It is inferred that majority of respondents prefer A.

4.1.9 ANAYSIS OF LEARNING STYLE (TARGET VARIABLE)

TABLE 4.10 LEARNING STYLE

Learning style	frequency	percentage	Cum percentage
Multi	246	62.12%	62.12%
K	64	16.16%	78.28%
A	44	11.11%	89.39%
V	32	8.08%	97.47%
R	10	2.53%	100.00%

INTERPRETATION

Table 4.10 shows the frequency analysis of the learning style. From table it is clearly evident that majority of respondents prefer Multi (246) and followed by K (64), A (44), V (32) and R (10). It is inferred that majority of respondents prefer Multi.

4.2 FIVE POINT SUMMARY WITH MEAN

TABLE 4.11 FIVE POINT SUMMARY

Detail	Student id	V1	A1	R1	K1	V2	A2	R2	K2
count	396	396	396	396	396	396	396	396	396
mean	198.5	3.180	3.309	3.264	3.281	3.906	3.729	3.601	3.848
std	114.459	0.954	0.911	0.927	0.913	0.615	0.648	0.705	0.672
min	1	1	1	1	1	1.833	1.667	1.167	1.333
25%	99.75	2.583	2.667	2.667	2.667	3.667	3.333	3	3.458
50%	198.5	3.333	3.333	3.333	3.333	4	3.833	3.667	4
75%	297.25	4	4	4	4	4.333	4.167	4.167	4.333
max	396	5	5	5	5	5	5	5	5

INTERPRETATION

Table 4.11 shows the five-point summary of the numerical descriptors in the dataset. From the table it can be interpreted that V2 has the highest mean pertaining to majority of the customers scoring more in V2. Standard deviation of the V2 is minimum (0.615) and quartiles range from 3.667 to 4.333. It is inferred that V2 has majority scoring as well as more values closer to mean.

4.3 ANOVA

TABLE 4.12 ANOVA RESULTS

S.NO	Variable	F statistic	p value
1	V1	2.572448	0.037461
2	A1	3.051259	0.01699
3	R1	4.854752	0.000786
4	K1	5.062573	0.000549
5	V2	3.042365	0.017244
6	A2	0.536994	0.708639
7	R2	1.937822	0.103411
8	K2	2.60547	0.035493

INTERPRETATION

Table 4.12 shows the ANOVA results.

Null Hypothesis: Variable has no effect on style

Alternate Hypothesis: Variable has effect on style

Here significance level is specified as 0.05. If the variable has p value less than 0.05 null hypothesis is rejected can alternate hypothesis is accepted. From the table it is inferred that V1, A1, R1, K1, V2 and K2 has an effect on style. A2 and R2 has no effect on style. We can interpret style variable from the first part of questionnaire has an effect on the numerical descriptors.

4.4 CHI SQUARE TEST

TABLE 4.13 CHI SQUARE TEST RESULTS

S.NO	Variable	chisquare value	p value
1	Gender	5.547826	0.235557
2	Age	24.93714	0.015124
3	Place grownup in	30.4778	0.169416
4	Educational designation	5.668069	0.684357
5	school region	28.70998	0.231237
6	school Type	5.311445	0.723829
7	school board	11.96928	0.15258
8	college location	191.3169	0.001251
9	educational stream	64.91396	0.193906

INTERPRETATION

Table 4.13 shows the chi-square test results.

Null Hypothesis: Variable has no effect on learning style

Alternate Hypothesis: Variable has effect on learning style

Here significance level is specified as 0.05. If the variable has p value less than 0.05 null hypothesis is rejected can alternate hypothesis is accepted. From the table it is evident that age and college location has an effect on the learning style (dependent variable). All other demographic variables have no effect on learning style.

4.5 CORRELATION

4.14 CORRELATION SUMMARY

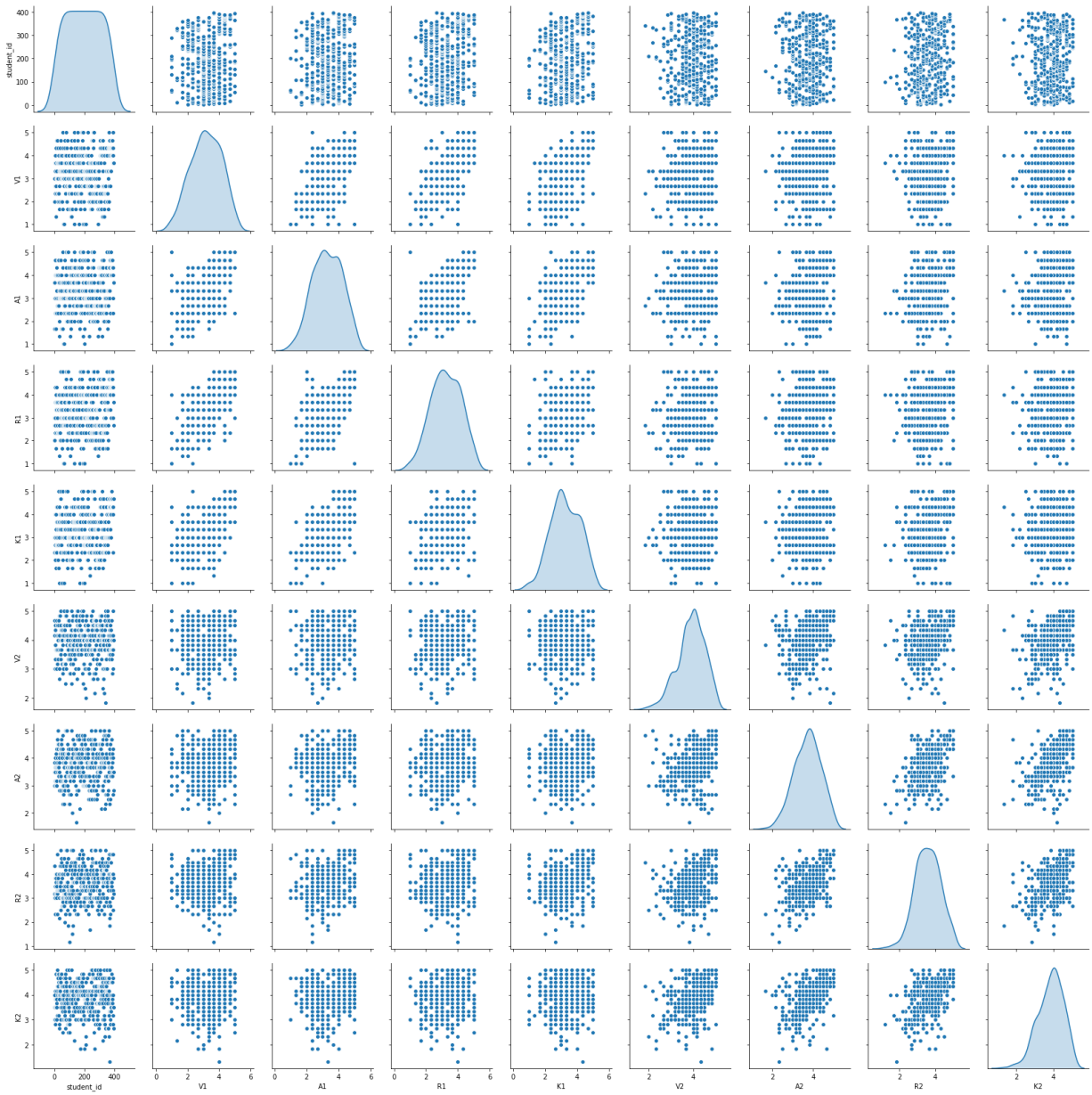
	student_id	V1	A1	R1	K1	V2	A2	R2	K2
student_id	1	0.0854	0.0902	0.1671	0.1758	0.0192	0.0238	0.1493	0.0032
V1	0.0854	1	0.6595	0.6996	0.6194	0.0921	0.1193	0.0833	0.0583
A1	0.0902	0.6595	1	0.6590	0.6675	0.1132	0.1936	0.2003	0.1259
R1	0.1672	0.6996	0.6590	1	0.5901	0.0969	0.1805	0.1495	0.0851
K1	0.1758	0.6194	0.6675	0.5901	1	0.0856	0.1679	0.1903	0.0741
V2	0.0192	0.0921	0.1132	0.0969	0.0856	1	0.2744	0.3380	0.4066
A2	0.0238	0.1193	0.1936	0.1805	0.1679	0.2744	1	0.5149	0.5197
R2	0.1493	0.0833	0.2003	0.1495	0.1903	0.3380	0.5149	1	0.4413
K2	0.0032	0.0583	0.1259	0.0851	0.0741	0.4066	0.5197	0.4413	1

INTERPRETATION

Table 4.14 shows the correlation summary. From the table it is clear that none of the numerical descriptors are negatively correlated to each other. There is strong correlation between V1 and R1, V1 and A1 and V1 and K1. Other than that, there is no significant correlation between the variables. It is inferred that V1, K1, A1 and K1 are better numerical descriptors for the target variable.

4.6 PAIR PLOT

FIGURE 4.1 PAIR PLOT



INTERPRETATION

Figure 4.1 shows the pair plot of the dataset. From the plot it is evident that there are only few linear relationships in the close to the diagonal. It can be inferred that this dataset is likely to perform with tree-based models as there is very few linear relationships.

The diagonal of the pair plot shows normalized curves between the variables. It can be inferred that most of the diagonal plots are negatively skewed. From the above inferences one can say that tree-based models are likely to perform well than linear models.

CHAPTER 5

MODEL ANALYSIS

Modeling for a classification involves in preprocessing of data. All the categorical variables are assigned labels using the label encoder. The whole dataset is split into training and testing sets by stratifying using the target variable. From the data analysis part, it is inferred that tree-based models are likely to perform better than the linear models. For the tuning of the hyperparameters grid search cross validation is used.

5.1 MODEL RESULTS

TABLE 5.01 MODEL RESULTS

S.NO	Method	accuracy	recall	precision	f1_score	Cross Val score
0	Logistic Regression	0.625	0.625	0.57357	0.57156	0.643924
1	Gaussian Naive bayes	0.55	0.55	0.51526	0.52799	0.563196
2	K Nearest Neighbour	0.6375	0.6375	0.50606	0.52290	0.507532
3	Support Vector Machines	0.65	0.65	0.67673	0.57272	0.621203
4	Decision tree	0.6	0.6	0.57686	0.58734	0.563038
5	Random Forest	0.7	0.7	0.74683	0.62761	0.648924
6	Gradient Boost	0.6875	0.6875	0.695	0.68169	0.669082
7	XG Boost	0.6875	0.6875	0.69300	0.67494	0.626329
8	Light GBM	0.6875	0.6875	0.73901	0.67402	0.661677
9	Extra trees Classifier	0.6375	0.6375	0.54290	0.54256	0.654051
10	Voting Classifier	0.7	0.7	0.73447	0.67559	0.686899
11	Neural Network classifier	0.62184	0.62184	0.55646	0.55111	0.67193

INTERPRETATION

Table 5.01 shows the results of the models.

All values lie between 0 to 1. It can be read as 0 to 100 percentage.

From the table it is evident that random forest classifier (70%) and voting classifier (70%) has the highest accuracy. And there is no contradiction from the findings in the data analysis part. It is evident that tree-based models performed well. As the dataset has imbalance in the target class precision, recall and f1 score are much better interpreters. As the voting classifier produces the better recall (70%), precision (73%) and f1 score (67.5%), one can infer that it is the better performing model than the random forest classifier. The cross-validation score is naturally higher for the voting classifier (68.5%) as it is the combination algorithm of the top six best performing models. Neural network classifier (Fully connected Neural networks with 2 hidden layers and optimizer stochastic gradient descent) gives the results lower than the traditional models as it uses linearity. It is inferred that deep learning models requires more data to withstand the unbalanced target class in the dataset.

CHAPTER 6

SUMMARY OF FINDINGS, SUGGESTIONS AND CONCLUSION

6.1 SUMMARY OF FINDINGS

1. The frequency table of age shows that majority of respondents are age group 20 to 25 (225) followed by age group 25 to 30 (92), age group less than 20 (61) and age group greater than 30 (18).
2. The frequency table of gender shows that majority of the respondents are female (207) and male (189).
3. The frequency table of educational designation shows that majority of the students are under graduates (234) followed by post graduates (158) and doctorial students.
4. The frequency table of educational stream shows that majority if the respondents are studying engineering (204) followed by management (57), arts (54), sciences (26) and medicine (23).
5. The frequency table of place grown up in shows that majority of the respondents grown up in city (194). It is followed by people grown up in town (85) and rural (63).
6. The frequency table of school region shows that majority of the respondents' school region is in city (183). It is followed by town (106) and rural (76).
7. The frequency table of school type shows that majority of respondents are of school type private (300) and followed by govt (83) and other (13).
8. The frequency table of school board shows that majority of respondents are of state board (296) and followed by central board (93) and other (7).
9. The frequency table of style shows that majority of respondents prefer A (136) and followed by K (131), Multi (55), V (45) and R (29).
10. The frequency table of style shows that majority of respondents prefer A (136) and followed by K (131), Multi (55), V (45) and R (29).
11. The frequency table of learning style shows that majority of respondents prefer Multi (246) and followed by K (64), A (44), V (32) and R (10). It is inferred that majority of respondents prefer Multi.
12. Five-point summary of the numerical descriptors shows that V2 has the highest mean pertaining to majority of the customers scoring more in V2. Standard deviation of the V2 is minimum (0.615) and quartiles range from 3.667 to 4.333.
13. ANOVA test shows that V1, A1, R1, K1, V2 and K2 has an effect on style. A2 and R2 has no effect on style. We can interpret style variable from the first part of questionnaire has an effect on the numerical descriptors.
14. Chi square test shows that age and college location have an effect on the learning style (dependent variable). All other demographic variables have no effect on learning style.
15. Pair plot shows that tree-based models are likely to perform better than linear models.
16. Model summary shows that voting classifier (with top 6 base models excluding Support vector machines) performs better than other models with accuracy (70%), recall (70%), precision (73%) and f1 score (67.5%). It is inferred that tree-based models performs better than the linear models and models gives lower results than expected due to insufficient data point. Also, deep learning algorithm (Fully connected Neural networks with 2 hidden layers and optimizer stochastic gradient descent) performs better with more data.

6.2 SUGGESTIONS

As the online platform is being reformed as the norm in the educational sector, the learning styles classification helps personalize the learning experience. On that note the scope for the learning style classification is wide especially beyond the sensory modalities there are many behavioral traits that has to be included. Growth in analytics sector gives away the readily available predictive algorithms. But algorithms that are specific to the student analytics or educational analytics may obtain better results regarding the classification of students based on their learning styles and traits. Processing the information based on the modalities applies for any human being, there is gap in study of learning style in corporate training and development process that may be explored.

6.3 CONCLUSION

This study shows the ability of the machine learning algorithms to ascertain the relationships between the data. The learning styles are important part of the student's way of processing the information during the education. Voting classifier may require the blend of computationally expensive algorithms to obtain better results. But better results trump that disadvantage. This study explored the relationship between demographic factors like school and place people grew up and learning styles. Results proved to contradict those factors. I conclude that with growth of big data learning style classification though blend of model algorithms or stacked algorithms like voting classifier can be used to adapt to a user application.

BIBLIOGRAPHY

1. Norasmah Othman and Mohd Hasril Amiruddin (2010). Different Perspectives of Learning Styles from VARK Model. International Conference on Learner Diversity, Procedia - Social and Behavioral Sciences, Vol.7, pp.652-660.
2. Yit Yan Koh and Yaw Long Chua (2012). The Study of Learning Styles among Mechanical Engineering students from Different Institutions in Malaysia. Procedia - Social and Behavioral Sciences, Vol.56, pp.636-642.
3. Nuzulla Mamat and Norazah Yusof (2013). Learning Style in A Personalized Collaborative Learning Framework. Procedia - Social and Behavioral Sciences, Vol.103, pp.586-594.
4. Căpiță Carol (2014). Procedia - Social and Behavioral Sciences, Vol.180, pp.256-261.
5. Milan Klement (2014). How do my students' study? An analysis of students` of educational disciplines favourite learning styles according to VARK classification. Procedia - Social and Behavioral Sciences, Vol.132, pp.384-390.
6. Norasyikin Omar, Mimi Mohaffyza Mohamad and Aini Nazura Paimin (2014). Dimension of Learning Styles and Students' Academic Achievement. Procedia - Social and Behavioral Sciences, Vol.204, pp.172-182.
7. Rathnakar P. Urval, Ashwin Kamath, Sheetal Ullal, Ashok K. Shenoy, Nandita Shenoy and Laxminarayana A. Udupa (2014). Assessment of learning styles of undergraduate medical students using the VARK questionnaire and the influence of sex and academic performance. Adv Physiol Educ, pp,216–220.
8. Miha Maric, Sandra Penger, Ivan Todorovic, Nina Djurica and Rok Pintar (2015). Differences in Learning Styles: A comparison of Slovenian Universities. Procedia - Social and Behavioral Sciences, Vol.197, pp.175-183.
9. M. H. Yee, J. Md. Yunos, W. Othman, R. Hassan, T. K. Tee and Mimi Mohaffyza Mohamadf (2015). Disparity of Learning Styles and Higher Order Thinking Skills among Technical Students. Procedia - Social and Behavioral Sciences, Vol.204, pp.143-152.
10. V. Balasubramanian and S. Margret Anouncia (2016). Learning style detection based on cognitive skills to support adaptive learning environment – A reinforcement approach. Ain Shams Engineering Journal, Vol.9, pp.895-907.
11. Andino Maseleno, Glenn Hardaker, Noraisikin Sabani and Nabilah Suhaili (2016). Data on multi-cultural education and diagnostic information profiling: Culture, learning styles and creativity. Data in Brief, Vol.9, pp.1048-1051.
12. Eva Švarcová and Kristýna Jelínková (2016). Detection of Learning Styles in the Focus Group. Procedia - Social and Behavioral Sciences, Vol.217, pp.177-182.
13. Sreenidhi S.K and Tay chinyi Helena (2017). Styles of Learning Based on the Research of Fernald, Keller, Orton, Gillingham, Stillman, Montessori and Neil D Fleming. International journal for innovative research in multidisciplinary field, Vol.3.
14. Bridget V. Stirling and Wadha A. Alquraini M.N (2017). Using VARK to assess Saudi nursing students' learning style preferences: Do they differ from other health professionals. Journal of Taibah University Medical Sciences, Vol.12, pp.125-130.
15. Richa Bajaj and Vidushi Sharmab (2018). Smart Education with artificial intelligence-based determination of learning styles. Procedia Computer Science Vol.132, pp.834-842.

16. Ouafae el Aissaoui, Yasser el Alami el Madani, Lahcen Oughdir and Youssouf el Allioui (2018). Combining supervised and unsupervised machine learning algorithms to predict the learners' learning styles. Procedia Computer Science, Vol.148, pp.87–96.
17. Veena Khongpit, Krich Sintanakul, and Thanyarat Nomphonkrang (2018). The VARK Learning Style of the University Student in Computer Course. International Journal of Learning and Teaching, pp.102-106.
18. Ling Xiao Li and Siti Soraya Abdul Rahman (2018). Students' learning style detection using tree augmented naive Bayes. Royal society open science.
19. Sucheta V. Kolekar, Radhika M. Pai and Manohara Pai M.M (2018). Adaptive User Interface for Moodle based E-learning System using Learning Styles. Procedia Computer Science, Vol.135, 2018, pp.606-615.
20. Wafa Labib, Irene Pasina, Abdelhakim Abdelhadi, Goze Bayrama and Mohammad Nurunnabib (2019). Learning style preferences of architecture and interior design students in Saudi Arabia: A survey. MethodsX, Vol.6, pp.961-967.
21. Amelec Viloria, Ingrid Regina Petro Gonzalez and Omar Bonerge Pineda Lezama (2019). Learning Style Preferences of College Students Using Big Data. Procedia Computer Science, Vol.160, pp.461-466.
22. Lucimar Almeida Dantas and Ana Cunha (2020). An integrative debate on learning styles and the learning process. Social Sciences & Humanities Open, Vol.2.
23. Everton Gomede, Rodolfo Miranda de Barros and Leonardo de Souza Mendes (2020). Use of Deep Multi-Target Prediction to Identify Learning Styles. Appl. Sci.
24. Hao Zhang, Tao Huang, Sanya Liu1, Hao Yin, Jia Li1, Huali Yang and Yu Xia (2020). A learning style classification approach based on deep belief network for largescale online education. Journal of Cloud Computing. Vol.9.

ANNEXURE-1

QUESTIONNAIRE

A STUDY ON LEARNING STYLES

Many people recognize that each person prefers different learning styles and techniques. Learning styles group common ways that people learn. Everyone has a mix of learning styles.

1. Name

2. Email

3. Age

   A. <20
   B. 20 to 25
   C. 25 to 30
   D. 30+

4. Gender

   A. Female
   B. Male
   C. Other

5. College Name

6. Name of city or Town where your college located

7. Your current educational designation

   A. Under Graduation
   B. Post Graduation
   C. Doctorial

8. In which stream are you studying (Example: Engineering, Medical, Arts...)

9. Place you grownup in? (Select Multiple options if needed) * (Check all that apply)

   A. City
   B. Rural
   C. Town

10. Region of your school (10+2 and before) (Select Multiple options if needed) (Check all that apply).

   A. City
   B. Rural
   C. Town

11. Type of high school (10+2)

A. Govt
B. Private
C. Other

12. Board of Education in high school (10+2)

A. State Board
B. Central Board
C. Other

13. You need to find the way to a movie theatre that a friend has recommended. You would:

A. find out where the shop is in relation to somewhere, I know.
B. ask my friend to tell me the directions.
C. write down the street directions I need to remember.
D. use a map.

14. You are not sure whether a word should be spelled dependent or dependant. You would:

A. see the words in your mind and choose by the way they look.
B. think about how each word sounds and choose one.
C. find it in a dictionary.
D. write both words on paper and choose one.

15. A website has a video showing how to make a special graph or chart. There is a person speaking, some lists and words describing what to do and some diagrams. you would learn most from:

A. seeing the diagrams.
B. listening.
C. reading the words.
D. watching the actions.

16. You are planning a holiday for a group. You want some feedback from them about the plan. You would:

A. describe some of the highlights.
B. use a map or website to show them the places.
C. give them a copy of the printed itinerary.
D. phone, text or email them.

17. When choosing a career or area of study, these are important for you:

A. Applying knowledge in real situations.
B. Communicating with others through discussion.
C. Working with designs, maps or charts.
D. Using words well in written communications.

18. You are going to cook something as a special treat(dish) for your friends. You would:

A. cook something you know without the need for instructions.
B. ask family members for suggestions.
C. look through the cookbook for ideas from the pictures.
D. use a cookbook where you know there is a good recipe

19. When you are learning you:

A. like to talk things through.
B. see patterns in things.
C. use examples and applications.
D. read books, articles and handouts.

20. A group of tourists want to learn about the parks or nature reserves in your area. You would:

A. talk about, or arrange a talk for them about parks or nature reserves.
B. show them internet pictures, photographs or picture books.
C. take them to a park or nature reserve and walk with them.
D. give them a book or pamphlets about the parks or nature reserves.

21. You are about to purchase a digital camera or mobile phone. Other than price, what would most influence your decision?

A. Trying or testing it.
B. Reading the details about its features.
C. It is a modern design and looks good.
D. The salesperson telling me about its features.

22. Remember a time when you learned how to do something new. Try to avoid choosing a physical skill, e.g. riding a bike. You learned best by:

A. watching a demonstration.
B. listening to somebody explaining it and asking questions.
C. diagrams and charts - visual clues.
D. written instructions – e.g. a manual or textbook.

23. You want to save more money and to decide between a range of options. you would:

A. consider examples of each option using my financial information.
B. read a print brochure that describes the options in detail.
C. use graphs showing different options for different time periods.
D. talk with an expert about the options.

24. You have a problem with your lungs. you would prefer that the doctor:

A. gave me something to read to explain what was wrong.
B. used a plastic model to show me what was wrong.
C. described what was wrong.
D. showed me a diagram of what was wrong.

25. You want to learn a new program, skill or game on a computer. You would:

A. read the written instructions that came with the program.
B. talk with people who know about the program.
C. use the controls or keyboard and explore.
D. follow the diagrams in the book that came with it.

26. When learning from the Internet you like:

A. videos showing how to do or make things.
B. interesting design and visual features.
C. interesting written descriptions, lists and explanations.
D. audio channels where I can listen to podcasts or interviews.

27. Other than price, what would most influence your decision to buy a book?

A. The way it looks is appealing.
B. Quickly reading parts of it.
C. A friend talks about it and recommends it.
D. It has real-life stories, experiences and examples.

28. You want to learn about a new project. you would ask for:

A. diagrams to show the project stages with charts of benefits and costs.
B. a written report describing the main features of the project.
C. an opportunity to discuss the project.
D. examples where the project has been used successfully.

29. You want to learn how to take better photos. you would:

A. ask questions and talk about the camera and its features.
B. use the written instructions about what to do.
C. use diagrams showing the camera and what each part does.
D. use examples of good and poor photos showing how to improve them.

30. After watching film you need to do a project. Would you prefer to:

A. draw or sketch something that happened in the film.
B. read a dialogue from the film,
C. write about the film.
D. act out a scene from the film.

31. Do you prefer a teacher who likes to use:

A. class discussions, online discussion, online chat and guest speakers.
B. a textbook and plenty of handouts.
C. an overview diagram, charts, labelled diagrams and maps.
D. field trips, case studies, videos, labs and hands-on practical sessions.

32. You have finished a competition or test and would like some feedback. You would like to have feedback:

A. using examples from what you have done.
B. using a written description of your results.
C. from somebody who talks it through with you.
D. using graphs showing what you had achieved.

33. A new movie has arrived in town. What would most influence your decision to go (or not go)?

A. it is similar to others you have liked.
B. hear friends talking about it.
C. you see a preview of it.
D. you read what others say about it online or in a magazine.

34. You are going to choose food at a restaurant or café. You would: choose something that you have had there before.

A. listen to the waiter or ask friends to recommend choices.
B. choose from the descriptions in the menu.
C. look at what others are eating or look at pictures of each dish.

35. You have to make an important speech at a conference or special occasion. You would:

A. make diagrams or get graphs to help explain things.
B. write a few key words and practice saying your speech over and over.
C. write out your speech and learn from reading it over several times.
D. gather many examples and stories to make the talk real and practical.

36. You want to find out about a house or an apartment. Before visiting it, you would want:

A. to view a video of the property.
B. a discussion with the owner.
C. a printed description of the rooms and features.
D. a plan showing the rooms and a map of the area.

37. You want to assemble a wooden table that came in parts (kit set). you would learn best from:

A. diagrams showing each stage of the assembly.
B. advice from someone who has done it before.
C. written instructions that came with the parts for the table.
D. watching a video of a person assembling a similar table.

ANSWER THE BELOW QUESTIONS (On the scale of 1 to 5)

** where (1) being LEAST often and (5) being MOST often ** select a number

38. How often do you Work in groups or with a study partner (i.e. discussions: listening, talking) for learning?

1 2 3 4 5

39. How often do you highlight important points in text; key words for learning?

1 2 3 4 5

40. How often do you read/review notes every day for learning?

1 2 3 4 5

41. How often do you skim through the reading material first to understand the theme or main idea for learning?

1 2 3 4 5

42. How often do you move around as u read aloud or study; walk and read; work in a standing position?

1 2 3 4 5

43. How often do you rewrite ideas and principles into other(own) words?

1 2 3 4 5

44. How often do you review assignments and text reading before class?

1 2 3 4 5

45. How often do you create flashcards for key information; be concise?

1 2 3 4 5

46. How often do you convert notes and translate words into symbols, diagrams, and/or pictures?

1 2 3 4 5

47. How often do you record notes, key information, and lectures; listen to recordings regularly? *

1 2 3 4 5

48. How often do you turn reactions, actions, charts, etc. into words and Organize diagrams/graphs into statements?

1 2 3 4 5

49. How often do you record notes in class and listen to them during exercising or while doing some other work?

1 2 3 4 5

SCENARIO 1

Read the scenario and answer the questions based on it

Days before exam you have go through all the chapters in this semester for a subject, will the following strategy help u score higher marks.

50. Going through Flash cards

A. Strongly agree
B. Agree
C. Neutral
D. Disagree
E. Strongly disagree

51. Listen to the recorded notes

A. Strongly agree
B. Agree
C. Neutral
D. Disagree
E. Strongly disagree

52. Write and rewrite the concepts

A. Strongly agree
B. Agree
C. Neutral
D. Disagree
E. Strongly disagree

53. Group study with friends

A. Strongly agree
B. Agree
C. Neutral
D. Disagree
E. Strongly disagree

54. Highlight important points

A. Strongly agree
B. Agree
C. Neutral
D. Disagree
E. Strongly disagree

55. Read the Notes aloud

A. Strongly agree
B. Agree
C. Neutral
D. Disagree

E. Strongly disagree

56. Rewrite the class notes

A. Strongly agree
B. Agree
C. Neutral
D. Disagree
E. Strongly disagree

57. Take frequent breaks while studying

A. Strongly agree
B. Agree
C. Neutral
D. Disagree
E. Strongly disagree

SCENARIO 2

Read the scenario and answer the questions based on it

You are planning to take a competitive exam next month hypothetically. U need to go through a blend of materials that ranges from mathematical concepts to pure descriptive concepts. Do you:

58. use charts, flashcards and mind-maps

A. Strongly agree
B. Agree
C. Neutral
D. Disagree
E. Strongly Disagree

59. Discuss questions/problems in a group or with a study-buddy

A. Strongly agree
B. Agree
C. Neutral
D. Disagree
E. Strongly Disagree

60. Write paragraphs, beginnings and endings

A. Strongly agree
B. Agree
C. Neutral
D. Disagree
E. Strongly Disagree

61. Limit information i.e. use key words, symbols

A. Strongly agree
B. Agree
C. Neutral
D. Disagree
E. Strongly Disagree

SCENARIO 3

Read the scenario and answer the questions

At the time of the final project in your course, you need to collect a lot of information pertaining to the topic you chose. U prefer collecting information from:

62. material consisting of lot of diagrams, charts, graphs and

A. Strongly agree
B. Agree
C. Neutral
D. Disagree
E. Strongly Disagree

63. discussing with faculty and various people

A. Strongly agree
B. Agree
C. Neutral
D. Disagree
E. Strongly Disagree

64. essays, articles, textbooks and manuals (Handbooks)

A. Strongly agree
B. Agree
C. Neutral
D. Disagree
E. Strongly Disagree

65. experiments, case studies, hands on exercises and field visits (includes industrial visits)

A. Strongly agree
B. Agree
C. Neutral
D. Disagree
E. Strongly Disagree

ANNEXTURE-2

DOCUMENTATION OF MODELS

LOGISTIC REGRESSTION

***class* sklearn.linear_model.LogisticRegression(*penalty='l2', *, dual=False, tol=0.0001, C =1.0, fit_intercept=True, intercept_scaling=1, class_weight=None, random_state=None, sol ver='lbfgs', max_iter=100, multi_class='auto', verbose=0, warm_start=False, n_jobs=None , l1_ratio=None*)**

Parameters

penalty*{'l1', 'l2', 'elasticnet', 'none'}, default='l2'*

Used to specify the norm used in the penalization. The 'newton-cg', 'sag' and 'lbfgs' solvers support only l2 penalties. 'elasticnet' is only supported by the 'saga' solver. If 'none' (not supported by the liblinear solver), no regularization is applied.

dual*bool, default=False*

Dual or primal formulation. Dual formulation is only implemented for l2 penalty with liblinear solver. Prefer dual=False when n_samples > n_features.

tol*float, default=1e-4*

Tolerance for stopping criteria.

C*float, default=1.0*

Inverse of regularization strength; must be a positive float. Like in support vector machines, smaller values specify stronger regularization.

fit_intercept*bool, default=True*

Specifies if a constant (a.k.a. bias or intercept) should be added to the decision function.

intercept_scaling*float, default=1*

Useful only when the solver 'liblinear' is used and self.fit_intercept is set to True. In this case, x becomes [x, self.intercept_scaling], i.e. a "synthetic" feature with constant value equal to intercept_scaling is appended to the instance vector. The intercept becomes intercept_scaling * synthetic_feature_weight..

class_weight*dict or 'balanced', default=None*

Weights associated with classes in the form {class_label: weight}. If not given, all classes are supposed to have weight one.

The "balanced" mode uses the values of y to automatically adjust weights inversely proportional to class frequencies in the input data as n_samples / (n_classes * np.bincount(y)).

random_state*int, RandomState instance, default=None*

Used when solver == 'sag', 'saga' or 'liblinear' to shuffle the data.

solver*{'newton-cg', 'lbfgs', 'liblinear', 'sag', 'saga'}, default='lbfgs'*

Algorithm to use in the optimization problem.

- For small datasets, 'liblinear' is a good choice, whereas 'sag' and 'saga' are faster for large ones.
- For multiclass problems, only 'newton-cg', 'sag', 'saga' and 'lbfgs' handle multinomial loss; 'liblinear' is limited to one-versus-rest schemes.
- 'newton-cg', 'lbfgs', 'sag' and 'saga' handle L2 or no penalty
- 'liblinear' and 'saga' also handle L1 penalty
- 'saga' also supports 'elasticnet' penalty
- 'liblinear' does not support setting penalty='none'

max_iter*int, default=100*

Maximum number of iterations taken for the solvers to converge.

multi_class*{'auto', 'ovr', 'multinomial'}, default='auto'*

If the option chosen is 'ovr', then a binary problem is fit for each label. For 'multinomial' the loss minimised is the multinomial loss fit across the entire probability distribution, *even when the data is binary*. 'multinomial' is unavailable when solver='liblinear'. 'auto' selects 'ovr' if the data is binary, or if solver='liblinear', and otherwise selects 'multinomial'.

verbose*int, default=0*

For the liblinear and lbfgs solvers set verbose to any positive number for verbosity.

warm_start*bool, default=False*

When set to True, reuse the solution of the previous call to fit as initialization, otherwise, just erase the previous solution. Useless for liblinear solver.

n_jobs*int, default=None*

Number of CPU cores used when parallelizing over classes if multi_class='ovr'". This parameter is ignored when the solver is set to 'liblinear' regardless of whether 'multi_class' is specified or not. None means 1 unless in a joblib.parallel_backend context. -1 means using all processors.

l1_ratio*float, default=None*

The Elastic-Net mixing parameter, with 0 <= l1_ratio <= 1. Only used if penalty='elasticnet'. Setting l1_ratio=0 is equivalent to using penalty='l2', while setting l1_ratio=1 is equivalent to using penalty='l1'. For 0 < l1_ratio <1, the penalty is a combination of L1 and L2.

GAUSIAN NAÏVE BAYES

class sklearn.naive_bayes.GaussianNB(*, priors=None, var_smoothing=1e-09)

Parameters

priors*array-like of shape (n_classes,)*

Prior probabilities of the classes. If specified the priors are not adjusted according to the data.

var_smoothing*float, default=1e-9*

Portion of the largest variance of all features that is added to variances for calculation stability.

K NEAREST NEIGHBOUR

class sklearn.neighbors.KNeighborsClassifier(n_neighbors=5, *, weights='uniform', algorithm='auto', leaf_size=30, p=2, metric='minkowski', metric_params=None, n_jobs=None, **kwargs)

Parameters

n_neighbors*int, default=5*

Number of neighbors to use by default for **kneighbors** queries.

weights*{'uniform', 'distance'} or callable, default='uniform'*

weight function used in prediction. Possible values:

- 'uniform' : uniform weights. All points in each neighborhood are weighted equally.
- 'distance' : weight points by the inverse of their distance. in this case, closer neighbors of a query point will have a greater influence than neighbors which are further away.
- [callable] : a user-defined function which accepts an array of distances, and returns an array of the same shape containing the weights.

algorithm*{'auto', 'ball_tree', 'kd_tree', 'brute'}, default='auto'*

Algorithm used to compute the nearest neighbors:

- 'ball_tree' will use **BallTree**
- 'kd_tree' will use **KDTree**
- 'brute' will use a brute-force search.
- 'auto' will attempt to decide the most appropriate algorithm based on the values passed to **fit** method.

Note: fitting on sparse input will override the setting of this parameter, using brute force.

leaf_size*int, default=30*

Leaf size passed to BallTree or KDTree. This can affect the speed of the construction and query, as well as the memory required to store the tree. The optimal value depends on the nature of the problem.

p*int, default=2*

Power parameter for the Minkowski metric. When p = 1, this is equivalent to using manhattan_distance (l1), and euclidean_distance (l2) for p = 2. For arbitrary p, minkowski_distance (l_p) is used.

metric*str or callable, default='minkowski'*

the distance metric to use for the tree. The default metric is minkowski, and with p=2 is equivalent to the standard Euclidean metric. See the documentation of **DistanceMetric** for a list of available metrics. If metric is "precomputed", X is assumed to be a distance matrix and must be square during fit. X may be a sparse graph, in which case only "nonzero" elements may be considered neighbors.

metric_params*dict, default=None*

Additional keyword arguments for the metric function.

n_jobs*int, default=None*

The number of parallel jobs to run for neighbors search. None means 1 unless in a **joblib.parallel_backend** context. -1 means using all processors.. Doesn't affect **fit** method.

SUPPORT VECTOR MACHINES

***class* sklearn.svm.SVC(*, *C=1.0, kernel='rbf', degree=3, gamma='scale', coef0=0.0, shrinking=True, probability=False, tol=0.001, cache_size=200, class_weight=None, verbose=False, max_iter= 1, decision_function_shape='ovr', break_ties=False, random_state=None*)**

Parameters

C*float, default=1.0*

Regularization parameter. The strength of the regularization is inversely proportional to C. Must be strictly positive. The penalty is a squared l2 penalty.

kernel*{'linear', 'poly', 'rbf', 'sigmoid', 'precomputed'}, default='rbf'*

Specifies the kernel type to be used in the algorithm. It must be one of 'linear', 'poly', 'rbf', 'sigmoid', 'precomputed' or a callable. If none is given, 'rbf' will be used. If a callable is given it is used to pre-compute the kernel matrix from data matrices; that matrix should be an array of shape (n_samples, n_samples).

degree*int, default=3*

Degree of the polynomial kernel function ('poly'). Ignored by all other kernels.

gamma*{'scale', 'auto'} or float, default='scale'*

Kernel coefficient for 'rbf', 'poly' and 'sigmoid'.

- if gamma='scale' (default) is passed then it uses 1 / (n_features * X.var()) as value of gamma,
- if 'auto', uses 1 / n_features.

coef0*float, default=0.0*

Independent term in kernel function. It is only significant in ‘poly’ and ‘sigmoid’.

shrinking*bool, default=True*

Whether to use the shrinking heuristic.

probability*bool, default=False*

Whether to enable probability estimates. This must be enabled prior to calling fit, will slow down that method as it internally uses 5-fold cross-validation, and predict_proba may be inconsistent with predict.

tol*float, default=1e-3*

Tolerance for stopping criterion.

cache_size*float, default=200*

Specify the size of the kernel cache (in MB).

class_weight*dict or ‘balanced’, default=None*

Set the parameter C of class i to class_weight[i]*C for SVC. If not given, all classes are supposed to have weight one. The “balanced” mode uses the values of y to automatically adjust weights inversely proportional to class frequencies in the input data as n_samples / (n_classes * np.bincount(y))

verbose*bool, default=False*

Enable verbose output. Note that this setting takes advantage of a per-process runtime setting in libsvm that, if enabled, may not work properly in a multithreaded context.

max_iter*int, default=-1*

Hard limit on iterations within solver, or -1 for no limit.

decision_function_shape*{‘ovo’, ‘ovr’}, default=’ovr’*

Whether to return a one-vs-rest (‘ovr’) decision function of shape (n_samples, n_classes) as all other classifiers, or the original one-vs-one (‘ovo’) decision function of libsvm which has shape (n_samples, n_classes * (n_classes - 1) / 2). However, one-vs-one (‘ovo’) is always used as multi-class strategy. The parameter is ignored for binary classification.

break_ties*bool, default=False*

If true, decision_function_shape='ovr', and number of classes > 2, predict will break ties according to the confidence values of decision_function; otherwise the first class among the tied classes is returned. Please note that breaking ties comes at a relatively high computational cost compared to a simple predict.

random_state*int or RandomState instance, default=None*

Controls the pseudo random number generation for shuffling the data for probability estimates. Ignored when probability is False. Pass an int for reproducible output across multiple function calls

DECISION TREES CLASSIFIER

***class* sklearn.tree.DecisionTreeClassifier(*, *criterion='gini', splitter='best', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features=None, random_state=None, max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, class_weight=None, presort='deprecated', ccp_alpha=0.0*)**

Parameters

criterion*{"gini", "entropy"}, default="gini"*

The function to measure the quality of a split. Supported criteria are "gini" for the Gini impurity and "entropy" for the information gain.

splitter*{"best", "random"}, default="best"*

The strategy used to choose the split at each node. Supported strategies are "best" to choose the best split and "random" to choose the best random split.

max_depth*int, default=None*

The maximum depth of the tree. If None, then nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples.

min_samples_split*int or float, default=2*

The minimum number of samples required to split an internal node:

- If int, then consider min_samples_split as the minimum number.
- If float, then min_samples_split is a fraction and ceil(min_samples_split * n_samples) are the minimum number of samples for each split.

min_samples_leaf*int or float, default=1*

The minimum number of samples required to be at a leaf node. A split point at any depth will only be considered if it leaves at least min_samples_leaf training samples in each of the left and right branches. This may have the effect of smoothing the model, especially in regression.

- If int, then consider min_samples_leaf as the minimum number.
- If float, then min_samples_leaf is a fraction and ceil(min_samples_leaf * n_samples) are the minimum number of samples for each node.

min_weight_fraction_leaf*float, default=0.0*

The minimum weighted fraction of the sum total of weights (of all the input samples) required to be at a leaf node. Samples have equal weight when sample_weight is not provided.

max_features*int, float or {“auto”, “sqrt”, “log2”}, default=None*

The number of features to consider when looking for the best split:

- If int, then consider max_features features at each split.
- If float, then max_features is a fraction and int(max_features * n_features) features are considered at each split.
- If “auto”, then max_features=sqrt(n_features).
- If “sqrt”, then max_features=sqrt(n_features).
- If “log2”, then max_features=log2(n_features).
- If None, then max_features=n_features.

random_state*int, RandomState instance, default=None*

Controls the randomness of the estimator. The features are always randomly permuted at each split, even if splitter is set to "best". When max_features < n_features, the algorithm will select max_features at random at each split before finding the best split among them. But the best-found split may vary across different runs, even if max_features=n_features. That is the case, if the improvement of the criterion is identical for several splits and one split has to be selected at random. To obtain a deterministic behaviour during fitting, random_state has to be fixed to an integer.

max_leaf_nodes*int, default=None*

Grow a tree with max_leaf_nodes in best-first fashion. Best nodes are defined as relative reduction in impurity. If None then unlimited number of leaf nodes.

min_impurity_decrease*float, default=0.0*

A node will be split if this split induces a decrease of the impurity greater than or equal to this value.

The weighted impurity decrease equation is the following:

N_t / N * (impurity - N_t_R / N_t * right_impurity - N_t_L / N_t * left_impurity)

where N is the total number of samples, N_t is the number of samples at the current node, N_t_L is the number of samples in the left child, and N_t_R is the number of samples in the right child. N, N_t, N_t_R and N_t_L all refer to the weighted sum, if sample_weight is passed.

min_impurity_split*float, default=0*

Threshold for early stopping in tree growth. A node will split if its impurity is above the threshold, otherwise it is a leaf.

class_weight*dict, list of dict or “balanced”, default=None*

Weights associated with classes in the form {class_label: weight}. If None, all classes are supposed to have weight one. For multi-output problems, a list of dicts can be provided in the same order as the columns of y.

The “balanced” mode uses the values of y to automatically adjust weights inversely proportional to class frequencies in the input data as n_samples / (n_classes * np.bincount(y))

presort*deprecated, default=’deprecated’*

This parameter is deprecated and will be removed in v0.24.

ccp_alpha*non-negative float, default=0.0*

Complexity parameter used for Minimal Cost-Complexity Pruning. The subtree with the largest cost complexity that is smaller than ccp_alpha will be chosen. By default, no pruning is performed. See Minimal Cost-Complexity Pruning for details.

RANDOM FOREST CLASSIFIER

***class* sklearn.ensemble.RandomForestClassifier(*n_estimators=100, *, criterion='gini', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features='auto', max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, bootstrap=True, oob_score=False, n_jobs=None, random_state=None, verbose=0, warm_start=False, class_weight=None, ccp_alpha=0.0, max_samples=None*)**

Parameters

n_estimators*int, default=100*

The number of trees in the forest.

criterion*{“gini”, “entropy”}, default=”gini”*

The function to measure the quality of a split. Supported criteria are “gini” for the Gini impurity and “entropy” for the information gain. Note: this parameter is tree-specific.

max_depth*int, default=None*

The maximum depth of the tree. If None, then nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples.

min_samples_split*int or float, default=2*

The minimum number of samples required to split an internal node:

- If int, then consider min_samples_split as the minimum number.
- If float, then min_samples_split is a fraction and ceil(min_samples_split * n_samples) are the minimum number of samples for each split.

min_samples_leaf*int or float, default=1*

The minimum number of samples required to be at a leaf node. A split point at any depth will only be considered if it leaves at least min_samples_leaf training samples in each of the left and right branches. This may have the effect of smoothing the model, especially in regression.

- If int, then consider min_samples_leaf as the minimum number.

- If float, then min_samples_leaf is a fraction and ceil(min_samples_leaf * n_samples) are the minimum number of samples for each node.

min_weight_fraction_leaf*float, default=0.0*

The minimum weighted fraction of the sum total of weights (of all the input samples) required to be at a leaf node. Samples have equal weight when sample_weight is not provided.

max_features*{"auto", "sqrt", "log2"}, int or float, default="auto"*

The number of features to consider when looking for the best split:

- If int, then consider max_features features at each split.
- If float, then max_features is a fraction and int(max_features * n_features) features are considered at each split.
- If "auto", then max_features=sqrt(n_features).
- If "sqrt", then max_features=sqrt(n_features) (same as "auto").
- If "log2", then max_features=log2(n_features).
- If None, then max_features=n_features.

max_leaf_nodes*int, default=None*

Grow trees with max_leaf_nodes in best-first fashion. Best nodes are defined as relative reduction in impurity. If None then unlimited number of leaf nodes.

min_impurity_decrease*float, default=0.0*

A node will be split if this split induces a decrease of the impurity greater than or equal to this value.

The weighted impurity decrease equation is the following:

N_t / N * (impurity - N_t_R / N_t * right_impurity - N_t_L / N_t * left_impurity)

where N is the total number of samples, N_t is the number of samples at the current node, N_t_L is the number of samples in the left child, and N_t_R is the number of samples in the right child. N, N_t, N_t_R and N_t_L all refer to the weighted sum, if sample_weight is passed.

min_impurity_split*float, default=None*

Threshold for early stopping in tree growth. A node will split if its impurity is above the threshold, otherwise it is a leaf.

bootstrap*bool, default=True*

Whether bootstrap samples are used when building trees. If False, the whole dataset is used to build each tree.

oob_score*bool, default=False*

Whether to use out-of-bag samples to estimate the generalization accuracy.

n_jobs*int, default=None*

The number of jobs to run in parallel. **fit**, **predict**, **decision_path** and **apply** are all parallelized over the trees. None means 1 unless in a **joblib.parallel_backend** context. -1 means using all processors.

random_state*int or RandomState, default=None*

Controls both the randomness of the bootstrapping of the samples used when building trees (if bootstrap=True) and the sampling of the features to consider when looking for the best split at each node (if max_features < n_features).

verbose*int, default=0*

Controls the verbosity when fitting and predicting.

warm_start*bool, default=False*

When set to True, reuse the solution of the previous call to fit and add more estimators to the ensemble, otherwise, just fit a whole new forest.

class_weight*{"balanced", "balanced_subsample"}, dict or list of dicts, default=None*

Weights associated with classes in the form {class_label: weight}. If not given, all classes are supposed to have weight one. For multi-output problems, a list of dicts can be provided in the same order as the columns of y.

Note that these weights will be multiplied with sample_weight (passed through the fit method) if sample_weight is specified.

ccp_alpha*non-negative float, default=0.0*

Complexity parameter used for Minimal Cost-Complexity Pruning. The subtree with the largest cost complexity that is smaller than ccp_alpha will be chosen. By default, no pruning is performed. See Minimal Cost-Complexity Pruning for details.

max_samples*int or float, default=None*

If bootstrap is True, the number of samples to draw from X to train each base estimator.

- If None (default), then draw X.shape[0] samples.
- If int, then draw max_samples samples.
- If float, then draw max_samples * X.shape[0] samples. Thus, max_samples should be in the interval (0, 1).

GRADIENT BOOSTING CLASSIFIER

***class* sklearn.ensemble.GradientBoostingClassifier(*, loss='deviance', learning_rate=0.1, n_estimators=100, subsample=1.0, criterion='friedman_mse', min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_depth=3, min_impurity_decrease=0.0, min_impurity_split=None, init=None, random_state=None, max_features=None, verbose=**

***0, max_leaf_nodes=None, warm_start=False, presort='deprecated', validation_fraction=0. 1, n_iter_no_change=None, tol=0.0001, ccp_alpha=0.0*)**

Parameters

loss*{'deviance', 'exponential'}, default='deviance'*

loss function to be optimized. 'deviance' refers to deviance (= logistic regression) for classification with probabilistic outputs. For loss 'exponential' gradient boosting recovers the AdaBoost algorithm.

learning_rate*float, default=0.1*

learning rate shrinks the contribution of each tree by learning_rate. There is a trade-off between learning_rate and n_estimators.

n_estimators*int, default=100*

The number of boosting stages to perform. Gradient boosting is fairly robust to over-fitting so a large number usually results in better performance.

subsample*float, default=1.0*

The fraction of samples to be used for fitting the individual base learners. If smaller than 1.0 this results in Stochastic Gradient Boosting. subsample interacts with the parameter n_estimators. Choosing subsample < 1.0 leads to a reduction of variance and an increase in bias.

criterion*{'friedman_mse', 'mse', 'mae'}, default='friedman_mse'*

The function to measure the quality of a split. Supported criteria are 'friedman_mse' for the mean squared error with improvement score by Friedman, 'mse' for mean squared error, and 'mae' for the mean absolute error. The default value of 'friedman_mse' is generally the best as it can provide a better approximation in some cases.

New in version 0.18.

min_samples_split*int or float, default=2*

The minimum number of samples required to split an internal node:

- If int, then consider min_samples_split as the minimum number.
- If float, then min_samples_split is a fraction and ceil(min_samples_split * n_samples) are the minimum number of samples for each split.

min_samples_leaf*int or float, default=1*

The minimum number of samples required to be at a leaf node. A split point at any depth will only be considered if it leaves at least min_samples_leaf training samples in each of the left and right branches. This may have the effect of smoothing the model, especially in regression.

- If int, then consider min_samples_leaf as the minimum number.

- If float, then min_samples_leaf is a fraction and ceil(min_samples_leaf * n_samples) are the minimum number of samples for each node.

min_weight_fraction_leaf*float, default=0.0*

The minimum weighted fraction of the sum total of weights (of all the input samples) required to be at a leaf node. Samples have equal weight when sample_weight is not provided.

max_depth*int, default=3*

maximum depth of the individual regression estimators. The maximum depth limits the number of nodes in the tree. Tune this parameter for best performance; the best value depends on the interaction of the input variables.

min_impurity_decrease*float, default=0.0*

A node will be split if this split induces a decrease of the impurity greater than or equal to this value.

The weighted impurity decrease equation is the following:

N_t / N * (impurity - N_t_R / N_t * right_impurity - N_t_L / N_t * left_impurity)

where N is the total number of samples, N_t is the number of samples at the current node, N_t_L is the number of samples in the left child, and N_t_R is the number of samples in the right child. N, N_t, N_t_R and N_t_L all refer to the weighted sum, if sample_weight is passed.

min_impurity_split*float, default=None*

Threshold for early stopping in tree growth. A node will split if its impurity is above the threshold, otherwise it is a leaf.

init*estimator or 'zero', default=None*

An estimator object that is used to compute the initial predictions. init has to provide **fit** and **predict_proba**. If 'zero', the initial raw predictions are set to zero. By default, a DummyEstimator predicting the classes priors is used.

random_state*int or RandomState, default=None*

Controls the random seed given to each Tree estimator at each boosting iteration. In addition, it controls the random permutation of the features at each split (see Notes for more details). It also controls the random spliting of the training data to obtain a validation set if n_iter_no_change is not None. Pass an int for reproducible output across multiple function calls.

max_features*{'auto', 'sqrt', 'log2'}, int or float, default=None*

The number of features to consider when looking for the best split:

- If int, then consider max_features features at each split.

- If float, then max_features is a fraction and int(max_features * n_features) features are considered at each split.
- If ‘auto’, then max_features=sqrt(n_features).
- If ‘sqrt’, then max_features=sqrt(n_features).
- If ‘log2’, then max_features=log2(n_features).
- If None, then max_features=n_features.

Choosing max_features < n_features leads to a reduction of variance and an increase in bias.

Note: the search for a split does not stop until at least one valid partition of the node samples is found, even if it requires to effectively inspect more than max_features features.

verbose*int, default=0*

Enable verbose output. If 1 then it prints progress and performance once in a while (the more trees the lower the frequency). If greater than 1 then it prints progress and performance for every tree.

max_leaf_nodes*int, default=None*

Grow trees with max_leaf_nodes in best-first fashion. Best nodes are defined as relative reduction in impurity. If None then unlimited number of leaf nodes.

warm_start*bool, default=False*

When set to True, reuse the solution of the previous call to fit and add more estimators to the ensemble, otherwise, just erase the previous solution.

validation_fraction*float, default=0.1*

The proportion of training data to set aside as validation set for early stopping. Must be between 0 and 1. Only used if n_iter_no_change is set to an integer.

n_iter_no_change*int, default=None*

n_iter_no_change is used to decide if early stopping will be used to terminate training when validation score is not improving. By default it is set to None to disable early stopping. If set to a number, it will set aside validation_fraction size of the training data as validation and terminate training when validation score is not improving in all of the previous n_iter_no_change numbers of iterations. The split is stratified.

tol*float, default=1e-4*

Tolerance for the early stopping. When the loss is not improving by at least tol for n_iter_no_change iterations (if set to a number), the training stops.

ccp_alpha*non-negative float, default=0.0*

Complexity parameter used for Minimal Cost-Complexity Pruning. The subtree with the largest cost complexity that is smaller than ccp_alpha will be chosen. By default, no pruning is performed.

XG BOOST CLASSIFIER

***class* xgboost.XGBClassifier(*objective='binary:logistic', use_label_encoder=True, **kwargs*)**

Parameters

- **n_estimators (*int*)** – Number of boosting rounds.
- **use_label_encoder (*bool*)** – (Deprecated) Use the label encoder from scikit-learn to encode the labels. For new code, we recommend that you set this parameter to False.
- **max_depth (*int*)** – Maximum tree depth for base learners.
- **learning_rate (*float*)** – Boosting learning rate (xgb's "eta")
- **verbosity (*int*)** – The degree of verbosity. Valid values are 0 (silent) - 3 (debug).
- **objective (*string or callable*)** – Specify the learning task and the corresponding learning objective or a custom objective function to be used (see note below).
- **booster (*string*)** – Specify which booster to use: gbtree, gblinear or dart.
- **tree_method (*string*)** – Specify which tree method to use. Default to auto. If this parameter is set to default, XGBoost will choose the most conservative option available. It's recommended to study this option from parameters document.
- **n_jobs (*int*)** – Number of parallel threads used to run xgboost. When used with other Scikit-Learn algorithms like grid search, you may choose which algorithm to parallelize and balance the threads. Creating thread contention will significantly slow dowm both algorithms.
- **gamma (*float*)** – Minimum loss reduction required to make a further partition on a leaf node of the tree.
- **min_child_weight (*float*)** – Minimum sum of instance weight(hessian) needed in a child.
- **max_delta_step (*int*)** – Maximum delta step we allow each tree's weight estimation to be.
- **subsample (*float*)** – Subsample ratio of the training instance.
- **colsample_bytree (*float*)** – Subsample ratio of columns when constructing each tree.
- **colsample_bylevel (*float*)** – Subsample ratio of columns for each level.
- **colsample_bynode (*float*)** – Subsample ratio of columns for each split.
- **reg_alpha (*float (xgb's alpha)*)** – L1 regularization term on weights
- **reg_lambda (*float (xgb's lambda)*)** – L2 regularization term on weights
- **scale_pos_weight (*float*)** – Balancing of positive and negative weights.
- **base_score** – The initial prediction score of all instances, global bias.

LIGHT GBM CLASSIFIER

***class*lightgbm.LGBMClassifier(*boosting_type='gbdt', num_leaves=31, max_depth=-1, learning_rate=0.1, n_estimators=100, subsample_for_bin=200000, objective=None, class_weight=None, min_split_gain=0.0, min_child_weight=0.001, min_child_samples=20, subsample=1.0, subsample_freq=0, colsample_bytree=1.0, reg_alpha=0.0, reg_lambda=0.0, random_state=None, n_jobs=- 1, silent=True, importance_type='split', **kwargs*)**

Parameters

- **boosting_type (*string, optional (default='gbdt')*)** – 'gbdt', traditional Gradient Boosting Decision Tree. 'dart', Dropouts meet Multiple Additive Regression Trees. 'goss', Gradient-based One-Side Sampling. 'rf', Random Forest.
- **num_leaves (*int, optional (default=31)*)** – Maximum tree leaves for base learners.
- **max_depth (*int, optional (default=-1)*)** – Maximum tree depth for base learners, <=0 means no limit.
- **learning_rate (*float, optional (default=0.1)*)** – Boosting learning rate. You can use callbacks parameter of fit method to shrink/adapt learning rate in training using reset_parameter callback. Note, that this will ignore the learning_rate argument in training.
- **n_estimators (*int, optional (default=100)*)** – Number of boosted trees to fit.
- subsample_for_bin (*int, optional (default=200000)*) – Number of samples for constructing bins.
- **objective (*string, callable or None, optional (default=None)*)** – Specify the learning task and the corresponding learning objective or a custom objective function to be used (see note below). Default: 'regression' for LGBMRegressor, 'binary' or 'multiclass' for LGBMClassifier, 'lambdarank' for LGBMRanker.
- **class_weight (*dict, 'balanced' or None, optional (default=None)*)** – Weights associated with classes in the form {class_label: weight}. Use this parameter only for multi-class classification task; for binary classification task you may use is_unbalance or scale_pos_weight parameters. Note, that the usage of all these parameters will result in poor estimates of the individual class probabilities. You may want to consider performing probability calibration of your model. The 'balanced' mode uses the values of y to automatically adjust weights inversely proportional to class frequencies in the input data as n_samples / (n_classes * np.bincount(y)). If None, all classes are supposed to have weight one. Note, that these weights will be multiplied with sample_weight (passed through the fit method) if sample_weight is specified.
- **min_split_gain (*float, optional (default=0.)*)** – Minimum loss reduction required to make a further partition on a leaf node of the tree.
- **min_child_weight (*float, optional (default=1e-3)*)** – Minimum sum of instance weight (hessian) needed in a child (leaf).

- **min_child_samples (*int, optional (default=20)*)** – Minimum number of data needed in a child (leaf).
- **subsample (*float, optional (default=1.)*)** – Subsample ratio of the training instance.
- **subsample_freq (*int, optional (default=0)*)** – Frequence of subsample, <=0 means no enable.
- **colsample_bytree (*float, optional (default=1.)*)** – Subsample ratio of columns when constructing each tree.
- **reg_alpha (*float, optional (default=0.)*)** – L1 regularization term on weights.
- **reg_lambda (*float, optional (default=0.)*)** – L2 regularization term on weights.
- **random_state (*int, RandomState object or None, optional (default=None)*)** – Random number seed. If int, this number is used to seed the C++ code. If RandomState object (numpy), a random integer is picked based on its state to seed the C++ code. If None, default seeds in C++ code are used.
- **n_jobs (*int, optional (default=-1)*)** – Number of parallel threads.
- **silent (*bool, optional (default=True)*)** – Whether to print messages while running boosting.
- **importance_type (*string, optional (default='split')*)** – The type of feature importance to be filled into feature_importances_. If 'split', result contains numbers of times the feature is used in a model. If 'gain', result contains total gains of splits which use the feature.
- ****kwargs** – Other parameters for the model

EXTRATREES CLASSIFIER

***class* sklearn.ensemble.ExtraTreesClassifier(*n_estimators=100, *, criterion='gini', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features='auto', max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, bootstrap=False, oob_score=False, n_jobs=None, random_state=None, verbose=0, warm_start=False, class_weight=None, ccp_alpha=0.0, max_samples=None*)**

Parameters

n_estimators*int, default=100*

The number of trees in the forest.

criterion*{"gini", "entropy"}, default="gini"*

The function to measure the quality of a split. Supported criteria are "gini" for the Gini impurity and "entropy" for the information gain.

max_depth*int, default=None*

The maximum depth of the tree. If None, then nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples.

min_samples_split*int or float, default=2*

The minimum number of samples required to split an internal node:

- If int, then consider min_samples_split as the minimum number.
- If float, then min_samples_split is a fraction and ceil(min_samples_split * n_samples) are the minimum number of samples for each split.

min_samples_leaf*int or float, default=1*

The minimum number of samples required to be at a leaf node. A split point at any depth will only be considered if it leaves at least min_samples_leaf training samples in each of the left and right branches. This may have the effect of smoothing the model, especially in regression.

- If int, then consider min_samples_leaf as the minimum number.
- If float, then min_samples_leaf is a fraction and ceil(min_samples_leaf * n_samples) are the minimum number of samples for each node.

min_weight_fraction_leaf*float, default=0.0*

The minimum weighted fraction of the sum total of weights (of all the input samples) required to be at a leaf node. Samples have equal weight when sample_weight is not provided.

max_features*{"auto", "sqrt", "log2"}, int or float, default="auto"*

The number of features to consider when looking for the best split:

- If int, then consider max_features features at each split.
- If float, then max_features is a fraction and int(max_features * n_features) features are considered at each split.
- If "auto", then max_features=sqrt(n_features).
- If "sqrt", then max_features=sqrt(n_features).
- If "log2", then max_features=log2(n_features).
- If None, then max_features=n_features.

max_leaf_nodes*int, default=None*

Grow trees with max_leaf_nodes in best-first fashion. Best nodes are defined as relative reduction in impurity. If None then unlimited number of leaf nodes.

min_impurity_decrease*float, default=0.0*

A node will be split if this split induces a decrease of the impurity greater than or equal to this value.

The weighted impurity decrease equation is the following:

N_t / N * (impurity - N_t_R / N_t * right_impurity - N_t_L / N_t * left_impurity)

where N is the total number of samples, N_t is the number of samples at the current node, N_t_L is the number of samples in the left child, and N_t_R is the number of samples in the right child. N, N_t, N_t_R and N_t_L all refer to the weighted sum, if sample_weight is passed.

min_impurity_split*float, default=None*

Threshold for early stopping in tree growth. A node will split if its impurity is above the threshold, otherwise it is a leaf.

bootstrap*bool, default=False*

Whether bootstrap samples are used when building trees. If False, the whole dataset is used to build each tree.

oob_score*bool, default=False*

Whether to use out-of-bag samples to estimate the generalization accuracy.

n_jobs*int, default=None*

The number of jobs to run in parallel. fit, predict, decision_path and apply are all parallelized over the trees.

random_state*int, RandomState, default=None*

Controls 3 sources of randomness:

- the bootstrapping of the samples used when building trees (if bootstrap=True)
- the sampling of the features to consider when looking for the best split at each node (if max_features < n_features)
- the draw of the splits for each of the max_features

verbose*int, default=0*

Controls the verbosity when fitting and predicting.

warm_start*bool, default=False*

When set to True, reuse the solution of the previous call to fit and add more estimators to the ensemble, otherwise, just fit a whole new forest.

class_weight*{"balanced", "balanced_subsample"}, dict or list of dicts, default=None*

Weights associated with classes in the form {class_label: weight}. If not given, all classes are supposed to have weight one. For multi-output problems, a list of dicts can be provided in the same order as the columns of y.

ccp_alpha*non-negative float, default=0.0*

Complexity parameter used for Minimal Cost-Complexity Pruning. The subtree with the largest cost complexity that is smaller than ccp_alpha will be chosen. By default, no pruning is performed.

max_samples*int or float, default=None*

If bootstrap is True, the number of samples to draw from X to train each base estimator.

- If None (default), then draw X.shape[0] samples.
- If int, then draw max_samples samples.
- If float, then draw max_samples * X.shape[0] samples. Thus, max_samples should be in the interval (0, 1).

VOTING CLASSIFIER

***class* sklearn.ensemble.VotingClassifier(*estimators, *, voting='hard', weights=None, n_jobs=None, flatten_transform=True, verbose=False*)**

Parameters

estimators*list of (str, estimator) tuples*

Invoking the fit method on the VotingClassifier will fit clones of those original estimators that will be stored in the class attribute self.estimators_. An estimator can be set to 'drop' using set_params.

voting*{'hard', 'soft'}, default='hard'*

If 'hard', uses predicted class labels for majority rule voting. Else if 'soft', predicts the class label based on the argmax of the sums of the predicted probabilities, which is recommended for an ensemble of well-calibrated classifiers.

weights*array-like of shape (n_classifiers,), default=None*

Sequence of weights (float or int) to weight the occurrences of predicted class labels (hard voting) or class probabilities before averaging (soft voting). Uses uniform weights if None.

n_jobs*int, default=None*

The number of jobs to run in parallel for fit. None means 1 unless in a joblib.parallel_backend context. -1 means using all processors.

flatten_transform*bool, default=True*

Affects shape of transform output only when voting='soft' If voting='soft' and flatten_transform=True, transform method returns matrix with shape (n_samples, n_classifiers * n_classes). If flatten_transform=False, it returns (n_classifiers, n_samples, n_classes).

verbose*bool, default=False*

If True, the time elapsed while fitting will be printed as it is completed.

NEURAL NETWORK CLASSIFIER

compile*(optimizer='SGD', loss=None, metrics=None, loss_weights=None, weighted_metrics=None, run_eagerly=None, **kwargs)*

fit*(x=None, y=None, batch_size=None, epochs=1, verbose=1, callbacks=None, validation_split=0.0, validation_data=None, shuffle=True, class_weight=None, sample_weight=None, initial_epoch=0, steps_per_epoch=None, validation_steps=None, validation_batch_size=None, validation_freq=1, max_queue_size=10, workers=1, use_multiprocessing=False)*

Arguments

x -Input data. It could be:

- A Numpy array (or array-like), or a list of arrays (in case the model has multiple inputs).
- A TensorFlow tensor, or a list of tensors (in case the model has multiple inputs).
- A dict mapping input names to the corresponding array/tensors, if the model has named inputs.
- A tf.data dataset. Should return a tuple of either (inputs, targets) or (inputs, targets, sample_weights).
- A generator or keras.utils.Sequence returning (inputs, targets) or (inputs, targets, sample_weights). A more detailed description of unpacking behavior for iterator types (Dataset, generator, Sequence) is given below.

y- Target data. Like the input data x, it could be either Numpy array(s) or TensorFlow tensor(s). It should be consistent with x (you cannot have Numpy inputs and tensor targets, or inversely). If x is a dataset, generator, or keras.utils.Sequence instance, y should not be specified (since targets will be obtained from x).

batch_size - Integer or None. Number of samples per gradient update. If unspecified, batch_size will default to 32. Do not specify the batch_size if your data is in the form of datasets, generators, or keras.utils.Sequence instances (since they generate batches).

Epochs - Integer. Number of epochs to train the model. An epoch is an iteration over the entire x and y data provided. Note that in conjunction with initial_epoch, epochs is to be understood as "final epoch". The model is not trained for a number of iterations given by epochs, but merely until the epoch of index epochs is reached.

Verbose - 0, 1, or 2. Verbosity mode. 0 = silent, 1 = progress bar, 2 = one line per epoch. Note that the progress bar is not particularly useful when logged to a file, so verbose=2 is recommended when not running interactively (eg, in a production environment).

Callbacks - List of keras.callbacks.Callback instances. List of callbacks to apply during training. See tf.keras.callbacks.

validation_split - Float between 0 and 1. Fraction of the training data to be used as validation data. The model will set apart this fraction of the training data, will not train on it, and will evaluate the loss and any model metrics on this data at the end of each epoch. The validation data is selected from the last samples in the x and y data provided, before shuffling. This argument is not supported when x is a dataset, generator or keras.utils.Sequence instance.

validation_data - Data on which to evaluate the loss and any model metrics at the end of each epoch. The model will not be trained on this data. Thus, note the fact that the validation loss of data provided using validation_split or validation_data is not affected by regularization layers like noise and dropuout. validation_data will override validation_split. validation_data could be:

- tuple (x_val, y_val) of Numpy arrays or tensors
- tuple (x_val, y_val, val_sample_weights) of Numpy arrays
- dataset For the first two cases, batch_size must be provided. For the last case, validation_steps could be provided. Note that validation_data does not support all the data types that are supported in x, eg, dict, generator or keras.utils.Sequence.

shuffle - Boolean (whether to shuffle the training data before each epoch) or str (for 'batch'). This argument is ignored when x is a generator. 'batch' is a special option for dealing with the limitations of HDF5 data; it shuffles in batch-sized chunks. Has no effect when steps_per_epoch is not None.

class_weight - Optional dictionary mapping class indices (integers) to a weight (float) value, used for weighting the loss function (during training only). This can be useful to tell the model to "pay more attention" to samples from an under-represented class.

sample_weight - Optional Numpy array of weights for the training samples, used for weighting the loss function (during training only). You can either pass a flat (1D) Numpy array with the same length as the input samples (1:1 mapping between weights and samples), or in the case of temporal data, you can pass a 2D array with shape (samples, sequence_length), to apply a different weight to every timestep of every sample. This argument is not supported when x is a dataset, generator, or keras.utils.Sequence instance, instead provide the sample_weights as the third element of x.

initial_epoch - Integer. Epoch at which to start training (useful for resuming a previous training run).

steps_per_epoch - Integer or None. Total number of steps (batches of samples) before declaring one epoch finished and starting the next epoch. When training with input tensors such as TensorFlow data tensors, the default None is equal to the number of samples in your dataset divided by the batch size, or 1 if that cannot be determined. If x is a tf.data dataset, and 'steps_per_epoch' is None, the epoch will run until the input dataset is exhausted. When passing an infinitely repeating dataset, you must specify the steps_per_epoch argument. This argument is not supported with array inputs.

validation_steps - Only relevant if validation_data is provided and is a tf.data dataset. Total number of steps (batches of samples) to draw before stopping when performing validation at the end of every epoch. If 'validation_steps' is None, validation will run until the validation_data dataset is exhausted. In the case of an infinitely repeated dataset, it will run into an infinite loop. If 'validation_steps' is specified and only part of the dataset will be consumed, the evaluation will start from the beginning of the dataset at each epoch. This ensures that the same validation samples are used every time.

validation_batch_size - Integer or None. Number of samples per validation batch. If unspecified, will default to batch_size. Do not specify the validation_batch_size if your data is

in the form of datasets, generators, or keras.utils.Sequence instances (since they generate batches).

validation_freq - Only relevant if validation data is provided. Integer or collections_abc.Container instance (e.g. list, tuple, etc.). If an integer, specifies how many training epochs to run before a new validation run is performed, e.g. validation_freq=2 runs validation every 2 epochs. If a Container, specifies the epochs on which to run validation, e.g. validation_freq=[1, 2, 10] runs validation at the end of the 1st, 2nd, and 10th epochs.

max_queue_size - Integer. Used for generator or keras.utils.Sequence input only. Maximum size for the generator queue. If unspecified, max_queue_size will default to 10.

Workers - Integer. Used for generator or keras.utils.Sequence input only. Maximum number of processes to spin up when using process-based threading. If unspecified, workers will default to 1. If 0, will execute the generator on the main thread.

use_multiprocessing - Boolean. Used for generator or keras.utils.Sequence input only. If True, use process-based threading. If unspecified, use_multiprocessing will default to False. Note that because this implementation relies on multiprocessing, you should not pass non-picklable arguments to the generator as they can't be passed easily to children processes.

DOCUMENTATION CITATIONS

- Buitinck, L., Louppe, G., Blondel, M., Pedregosa, Fabian, Mueller, A., Grisel, O., … Ga"el Varoquaux. (2013). API design for machine learning software: experiences from the scikit-learn project. In ECML PKDD Workshop: Languages for Data Mining and Machine Learning, pp. 108–122.
- Abadi, Mart' in, Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., … others. (2016). Tensorflow: A system for large-scale machine learning. In Symposium on Operating Systems Design and Implementation. pp. 265–283.
- Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 785–794.
- Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, Tie-Yan Liu (2017). LightGBM: A Highly Efficient Gradient Boosting Decision Tree. Advances in Neural Information Processing Systems, vol.30. pp. 3149-3157