Impacts of Machine Learning in the Computer Gaming Industry

Table of Contents

Table of Figures

Abbreviations

AI	Artificial Intelligence
ARPG	Action Role Playing Game
CNN	Convolutional Neural Network
DLSS	Deep Learning Super Sampling
EMGCD	Explainable Multi-View Game Cheating Detection
FPS	First Person Shooter
GAIL	Generative Adversarial Imitation Learning
GAN	Generative Adversarial Network
GBM	Gradient Boosting Machine
LGBM	Light Gradient Boost Machine
ML	Machine Learning
MLPNN	Multi Layer Perceptron Neural Network
MMO	Massively Multiplayer Online
MMORPG	Massively Multiplayer Online Role Playing Game
MOBA	Multiplayer Online Battle Arena
MVAN	Multi-View Attention Network
NLP	Natural Language Processing
NN	Neural Network
NPC	Non-Player Character
PCG	Procedural Content Generation
PPO	Proximal Policy Optimisation
RL	Reinforcement Learning
RMT	Real Money Transfer
RNN	Recurrent Neural Network
RPG	Role Playing Game
RTS	Real-time Strategy
TCNN	Temporal Convolutional Neural Network
TPS	Third Person Shooter
TSM	Traditional Statistical Methods
XGBoost	Extreme Gradient Boosting

A. Part-1: Impacts of Machine Learning in the Computer Gaming Industry

1. Introduction

Computer gaming has become one of the most exciting and creative industries globally, with annual revenues exceeding $200 billion in recent years (Arai,2022). Most smart games incorporate Artificial Intelligence(AI), including Machine Learning(ML) and Reinforcement Learning(RL), which are popular among researchers and gaming community (Arai,2022;Sun *et al.*,2019). In order to understand the impact of ML in the computer gaming industry, it's important to have an understanding of ML and other traditional approaches.

2. Machine Learning and Related Fields

ML is a subset of AI in which computer software learns complex patterns without explicit instruction to infer a user-defined problem (Nabi,2019;Takano,2021). ML comprises a dataset, a model and a function that evaluates model accuracy (Jung,2022). It is capable of making predictions on unseen data with high accuracy (Murdoch *et al.*,2019).

The fundamental ML concepts are derived from other fields of science, including Linear Algebra, Optimization, Probability and Statistics, Information Theory and Theoretical Computer Science (Jung,2022). For example, Regularization is taken from Linear Algebra and is used to reduce the coefficients' size when fitting a model to data (Brownlee,2019). Another example is the Generative Adversarial Network(GAN), a class of ML methods that uses probabilistic models to generate simulated data (Jung,2022).

There is often a misconception that ML and statistics are synonymous (Stewart,2020). To clarify this statement, it is important to understand the difference between Statistics and Statistical methods(ibid). Statistics involves the mathematical analysis of data whereas Statistical methods are used in statistics to infer relationships within the data and predict future values(ibid). Having understood the difference between statistics and statistical models, the next section will cover the difference between ML and statistical methods in detail.

2.1. Difference between Machine Learning and Traditional Statistical Methods

Fig.1 illustrates the key differences between ML and Traditional Statistical Methods(TSM). Firstly, the objectives of ML and TSM are different. The former focus on learning the data without explicit instruction, identifying patterns and relationships in datasets, and making accurate predictions based on the learning (Rajula et al.,2020;Takano,2021). While the latter focus on inferring variable relationships through hypothesis testing and choosing an appropriate model to generalise the findings to a larger population (Rajula et al.,2020). An ML model in the training phase learns how to infer from a subset of data, and its accuracy is tested with unseen data in the testing phase whereas TSM involves no training or testing phases (Stewart,2020). TSM aims to characterize the data and outcome variable relationship, rather than to predict the outcome variable(ibid).

	Machine Learning	Traditional Statistical Methods
Purpose	Focus on making accurate predictions, identifying patterns and derive new insights from data.	Focus on inferring relationships between variables, hypothesis testing, choose appropriate model to generalise the findings to a larger population.
Training and Testing	An ML model in the training phase learns how to infer from a subset of data, and its accuracy is tested with unseen data in the testing phase.	No training or testing is involved.
Input Size	Prediction process is highly data-intensive and often uses a wide range of input parameters.	Considers only a few important input parameters for calculations and input dataset is small or medium sized.
Datatype	Effectively handle unstructured and high-dimensional data including images. It can handle non-linear relationships better than traditional statistical methods.	Best suited for continuous, categorical, ordinal, time-series and binary data.
Transparency and Complexity	Often highly complex and difficult to interpret, and in some models such as neural networks, calculations that led to the final decision are not transparent.	Easy to understand and transparent calculations that led to the final decision.
Assumptions	Free of a priori assumptions. Based on large dataset and leverages algorithms to identify patterns in data.	Based on strong assumptions on data such as normality, variance, independence, and linearity.
Suitable for	More number of predictors and very less observations.	Limited number of variables and predictors, and the number of observations is much greater than the number of variables.
Human Intervention	Less human intervention since processes are automated and require less expert knowledge.	Require human experts for model selection and result interpretation.

Figure 1. Comparison between Machine Learning and Traditional Statistical Methods (Rajula et al.,2020;Stewart,2020; Takano,2021)

The ML model requires heavy computing power because it involves large amounts of high-dimensional data and parameters during the training and inferring phases for more accurate predictions, while the TSM doesn't require computation power and works with relatively small data sets and a limited number of parameters (Hassibi,2016;Thompson *et al.*,2022). While ML has been around for decades, the

lack of data availability and computational power has caused it to be overlooked for a long time (Stewart,2020). With the advent of cheaper computing power and an abundance of data, ML has evolved rapidly(ibid). Furthermore, ML models have varying levels of interpretability, for instance, Lasso Regression is easy to interpret, and Neural Networks (NN) are almost impossible to interpret(ibid).

TSM relies on strong assumptions about data, such as normality, variance, independence, and linearity, and follows analytical learning when data is scarce or small datasets are preferred requiring extensive prior knowledge of the problem and data(Rajula et al.,2020; Hassibi,2016). In contrast to TSM, ML makes no a priori assumptions and often follows a heuristic-driven approach that relies on inductive learning from a huge dataset without requiring extensive prior knowledge of the problem or data(Hassibi,2016).

To summarise, both ML and TSM are two distinct scientific fields, but they share many statistical concepts, so they should collaborate to achieve better results. Having understood the differences between ML and TSM, the next section will cover the reasons for the wider acceptance of the computer games worldwide and the relevance of AI in this field.

3. Reasons for the Popularity of Computer Games and the Relevance of AI in Computer Gaming Industry

Children and adults across the world enjoy computer gaming alike because there is a wide range of interesting game genres available through different devices, including mobile phones, that are engaging, challenging and rewarding, promoting psychological well-being and socialization (Adair,2022;Saltzman,2022). Fig.2 illustrates most popular computer game genres in the industry. The player experience offered by computer games is one of the key elements that influence the enjoyment level (Eshuis,2023). Furthermore, computer gaming is a cheaper alternative for socialising while living costs continue to rise (Emmanuel,2022).

There are many factors driving the use of ML in computer gaming, including the fact that games are complex, popular, interesting, highly interactive and engaging(Yannakakis and Togelius,2018). Due its increasing popularity, AI solutions are suitable to handle complex gaming contents and an enormous amount of play data

generated through computer games(ibid). The adoption of AI in computer gaming industry resulted in high-quality games driving drastic revenue growth globally(Media,2019). With the help of AI, the data captured through computer games about the players performance, emotions, gaming genre preference etc. can be analysed and utilised in the design, marketing and production of games, resulting in better player satisfaction(Yannakakis and Togelius,2018). The next paragraphs will cover the applications of ML in the computer gaming industry.

Computer Game Genre	Description	Examples
Sandbox	This genre offers the player a virtual-world experience without any predefined strict goals or objectives. Players are allowed to explore and interact with system in different ways to accomplish a variety of tasks with the available resources.	• Grand Theft Auto (GTA) • Minecraft
Real-time Strategy (RTS)	Games in this genre often involve resource or map management to defeat opponents to achieve objectives. Players manage small organised groups such as civilisations or armies to compete in real-time with other players or bots using different tactics.	• Age of Empires • Command and Conquer
Shooters (FPS and TPS)	This genre has two subtypes – First Person Shooter (FPS) and Third Person Shooter (TPS). Players of the FPS are only able to see the hands and weapons used by the main character in the game while players of TPS can see the entire character and surroundings including the obstacles and opponents.	• Halo (FPS) • Gears of War (TPS)
Multiplayer Online Battle Arena (MOBA)	Similar to RTS, this genre involves resource management, map management and offers top-down view. The key difference between MOBA and RTS is that the former focuses on controlling a main character rather than group such as armies.	• Dota 2 • League of Legends • Smite
Massively Multiplayer Online(MMO)	This genre involves thousands of players playing simultaneously in the same virtual environment competing with each other or completing different quests etc.	• Lost Ark
Role Playing Game (RPG)	This genre allows players to choose a character to explore the virtual world to complete quests and level up with new capabilities or tools as the character gains more experience points (XP). There are many subtypes of this genre including MMORPG, ARPG etc. Massively Multiplayer Online RPG (MMORPG) is the combination of MMO and RPG and offers players to customise their characters. Action RPG (ARPG) focus on combat actions.	• Fallout 4 (RPG) • Neverwinter (MMORPG) • The Witcher 3 (ARPG)
Simulation and Sports	Simulation genre offers detailed simulation of real-world activities that enhance player experience. Sports genre is a subtype of this genre, which simulates sports such as basket ball, soccer etc., allowing players to compete against other players or bots.	• Forza Motorsport • FIFA
Puzzlers	This genre offers players with puzzles or brain teasers. It can be simple or advanced theme based games.	• The Talos Principle
Party Games	This genre offers fun, often loud and fast-paced mini games that can played in a social group such as friends or families.	• Super Mario Party
Action Adventure	This genre focus more on fighting and strategy through deep engrained storylines and tough gameplay. It often involves revealing mysteries or completing quests.	• Star Wars Jedi: Fallen Order • Assassin's Creed
Survival and Horror	Survival genre allows players to manage resources, often crafting or salvaging systems that help the main character to survive in the virtual environment. Horror genre is similar to survival genre whereas it involves horror aspects such as zombies.	• Don't Starve (survival) • Resident Evil (survival & horror)
Platformer	This genre allows players to control a character through an environment with a series of actions such as running, jumping, or climbing, and the difficulty level increases as the game progresses.	• Crash Bandicoot • Super Mario Bros

Figure 2.Popular Computer Game Genres (Duetzmann,2016;MasterClass,2021; Pavlovic,2020; Plarium,2022;Smith,2023; Wald,2022)

4. Applications of Machine Learning in the Computer Gaming Industry

Currently, there are several ML applications in computer gaming industry however a few are covered in the following sections.

4.1. Non-Player Characters (NPCs)

AI is used to enhance the player experience by personalised computer games (Pretty,Fayek and Zambetta,2023;Riedl and Bulitko,2021). Non-Player Character (NPC) is any character in the computer games that acts either as an ally or an enemy

without being controlled by the players, making the game more interesting and engaging (Pretty,Fayek and Zambetta,2023; Stephenson,2019). Dogmeat, a dog from the Fallout-4 game is an example for NPC (Pretty,Fayek and Zambetta,2023). NPC can range from simple animation without any dialogue to sophisticated characters with personality or backstories(ibid). Personalised NPC can respond less predictably by adjusting difficulty levels based on the player performance and state, boosting player engagement and motivating them to engage in a longer session(ibid).

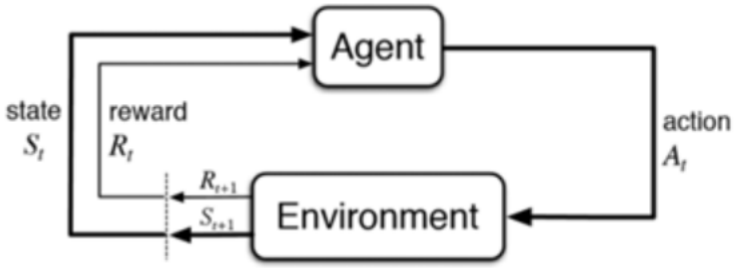


Figure 3.Working of Classic Reinforcement Learning(Rahman,2021)

RL is one of the ML techniques used to determine the optimal actions or behaviour of the NPC based on the higher cumulative reward for a given game context (Fröberg and Håkansson,2021; Pretty,Fayek and Zambetta,2023). As illustrated in Fig.3, RL involves an agent taking actions to receive states and rewards, with the goal of learning an optimal action to maximise the reward through trial-and-error method within an environment (Rahman,2021). For instance, in an action game, the player's state can be represented as the distance to the opponent, if the player defeats the opponent, then agent will be rewarded with positive value however if the opponent defeats the player, then it's rewarded with negative value where the state and rewards are defined by the game designer (ibid). Fig.4 illustrates ML techniques used to train and defeat smart NPCs.

ML Technique	Description	Citation
Generative Adversarial Imitation Learning(GAIL)	Method used to implement smart NPC with human-like behaviour. It generate positive reward for mimicking the expert actions rather than trial-and-error method followed by classic RL method	(Rahman,2021)
Proximal Policy Optimisation(PPO)	RL models have demonstrated unique gaming strategies that outperform humans. So, PPO, a RL class, is used to train an agent to learn the best strategy to defeat smart NPC. Its easy to implement and offer better performance than state-of-the-art methods.	(Rahman,2021; Schulman *et al.*,2017).

Figure 4. ML techniques for training and defeating smart NPCs.

4.2. Real-time Rendering

The real-time rendering for computer games has become increasingly challenging due to high-definition resolutions, refresh rates and photorealistic effects, leading to higher computational costs (Xiao *et al.*,2020). The modern graphics hardware resolves this problem by rendering the content in low-quality resolution in order to decrease the computational costs and then leverages ML techniques for high-quality upsampling in computer games (ibid).

Upsampling is a technique used to balance the class labels by creating artificial data points or replicating smaller sample(Kumar,2021). Neural super-sampling is a technique based on Convolutional Neural Network (CNN) used for high-quality upsampling of image content in real-time computer games(Xiao *et al.*,2020). This method is easy to integrate with the existing platforms without requiring any additional hardware or software and can accept low resolution content from modern game engines to produce high-quality content(ibid).

4.3. Game Victory Prediction

It is essential to evaluate probability of player winning a computer game in order to enhance the appeal of gameplay, especially in online games and the opponent is an AI bot (Ruta *et al.*,2021). However, the game victory prediction is often difficult as it depends upon numerous factors including the game's complexity, stage and play data such as player performance, emotional state etc. (ibid). ML can be used to predict the probability of winning game and the factors driving the outcome of game from huge volume of playing data (ibid). Gradient Boosting Machine(GBM) model is one of the techniques used to predict the game victory since it performs well for binary classification, in this case it is win or lose the game, from a large complex multidimensional including game and play datasets when fitted with robust feature engineering (ibid).

GBM is a kind of ensemble method that combine weak models to get best overall performance (Masui,2022). Extreme Gradient Boosting(XGBoost) and Light Gradient Boost Machine(LGBM) are examples of this model(ibid). GBM works well with input dataset having missing values, outliers, categorical values with high cardinality(ibid).

As mentioned earlier, robust feature engineering is key for the better accuracy of ML models. It is a technique that derives new variables from the data to simplify and accelerate transformations while promoting the model accuracy (Patel,2021). However, a bad feature can negatively impact the performance of a ML model(ibid).

4.4. Game Cheating Detection

Despite its popularity, the computer gaming industry is losing revenue and game developer reputation every year due to game cheating (Ackley,2022;Tao *et al.*,2020). Due to unfair advantages gained by fraud players through illegal bots and plugins, honest players lose interest later in the game as it adversely affects their gaming experience(ibid). For example, in games like MMORPG where millions of players share the dynamic virtual environment and events can impact the player's experience even after logging off, fraud players can easily gain superiority in the game using game bots, resulting in a massive imbalance in the gaming ecosystem (Rouse,2017;Tao *et al.*,2020). Similarly, the perspective plug-in is commonly used in FPS games by fraud players to gain unfair advantage over the honest players(Tao *et al.*,2020). Different ML techniques are used for computer game cheating detection and a few are depicted in Fig.4.

Detection Technique	Description
Portrait view	Leverages a time series technique with a MLPNN model to differentiate human players from game bots based on a set of portrait features including player and game features.
Behaviour view	Utilise a combination of supervised and unsupervised learning methods to identify game bots based on the user behaviour patterns.
Graph view	Constructs a social graph involving Real Money Transfer(RMT) while exchanging virtual goods and identifies the out-game transactions.
Image view	Deploys a automated real-time wallhack detection system using CNN to find vulnerabilities in the game.
Multi-view Attention Network (MVAN)	Employs a combination of portrait, behaviour and graph view methods to identify RMT in games.
Explainable Multi-view Game Cheating Detection Framework (EMGCD)	Although the above ML techniques are highly accurate in detecting cheating, they do not provide stakeholders such as game designers, scientists, operators, and customer service teams with an explanation for the decision. The EMGCD combines the features of portrait, behaviour, and image classifying techniques with the evidence obtained from the graph view to provide a clear explanation of the decision.

Figure 5.ML techniques for Game Cheating Detection(Tao et al.,2020)

5. Future Applications of Machine Learning in the Computer Gaming Industry

As AI advances rapidly, it can offer many more benefits and innovative solutions that could revolutionize the computer gaming industry(Darbinyan,2022). The following paragraphs will cover a few future ML applications in the computer gaming industry.

5.1. Game Content Generation

Gaming content generation is a highly time-consuming and resource intensive process (Darbinyan,2022). In order to gain a strong market presence and provide an exceptional gameplay experience, game content creators have to produce innovative content that satisfies a wide range of players. AI can be used to generate game content in large scale more quickly with less cost(ibid). Fig.6 illustrates some ML techniques that can be used for game content generation.

ML Technique	Application
Procedural Content Generation (PCG)	Algorithm to generate game content including challenges, quest, game characters etc.
Reinforcement Learning (RL)	Generate intelligent game characters and adaptive games based on play data. Also, to improve game content based on player preference and performance.
Generative Adversarial Network (GAN)	Rendering high-definition and photorealistic game scenes and characters.

Figure 6.ML techniques for Computer Game Content Generation(Pretty,Fayek and Zambetta,2023; Risi and Togelius,2020)

5.2. Personalised and Realistic Gaming Experience

AI is capable for enhancing the performance and the rendering of visuals in computer games, and making player experience more natural and realistic(Darbinyan,2022). Virtual augmented reality combine with AI can create even more immersive, interactive and personalised experience in video games(ibid). Currently, most games follow predefined script for dialogues and story line(Ambalina,2023). However, AI can be used to make game characters that deliver dynamic dialogue based on player input including text and speech, just like in the real-world(ibid). Fig.7 depicts some ML techniques for personalised and dynamic gaming experience.

ML Technique	Application
Natural Language Processing (NLP)	Generate dynamic conversation between game characters. Also, can be used to interpret the player input including text and speech and receive human-like response in real-time.
Reinforcement Learning (RL)	Generate personalised and adaptive virtual environment in computer games
Recurrent Neural Network (RNN)	Extracting information from the human language and generate appropriate response back to the player.
Deep Learning Super Sampling (DLSS)	Rendering ultra high-definition and photorealistic game scenes and characters. Also, offer outstanding performance.

Figure 7.ML techniques for Personalised and Dynamic Gaming Experience (Burnes,2023;Pretty,Fayek and Zambetta,2023;Zhu,2022)

6. Conclusion

AI has the potential to revolutionise the computer gaming industry and enhance the player experience drastically. Firstly, this report covered the key differences between

ML and traditional statistical approach. Then covered the relevance and the impact of ML in the current computer gaming industry and discussed some future applications of ML in this industry. Although AI offers several benefits, it is also critical to understand the challenges of implementing AI such as ethics, privacy and explanations for decisions, to achieve new heights in this industry.

7. References

- Ackley, M. (2022) *How online gaming fraud affects the industry*, *Kount*. Available at: https://kount.com/blog/effects-online-gaming-fraud-schemes/ (Accessed: March 21, 2023).
- Adair, C. (2022) *15 reasons people play video games*, *Game Quitters*. Available at: https://gamequitters.com/15-reasons-people-play-video-games/ (Accessed: March 18, 2023).
- Ambalina, L. (2023) *5 predictions for the future of AI in the gaming industry*, *AI Time Journal - Artificial Intelligence, Automation, Work and Business*. Available at: https://www.aitimejournal.com/5-predictions-for-the-future-of-ai-in-the-gaming-industry/23061/?utm_content=cmp-true (Accessed: March 21, 2023).
- Arai, K. (2022) *Proceedings of the future technologies conference (FTC) 2022, volume 1*, *SpringerLink*. Cham, Switzerland: Springer International Publishing AG. Available at: https://link.springer.com/chapter/10.1007/978-3-031-18461-1_11 (Accessed: March 11, 2023).
- Brownlee, J. (2019) *10 examples of linear algebra in machine learning*, *MachineLearningMastery.com*. Available at: https://machinelearningmastery.com/examples-of-linear-algebra-in-machine-learning/ (Accessed: March 9, 2023).
- Burnes, A. (2023) *Nvidia DLSS 3: Coming to diablo IV, Forza Horizon 5 & redfall, DLSS frame generation publicly available at GDC*, *NVIDIA*. Available at: https://www.nvidia.com/en-gb/geforce/news/march-2023-rtx-dlss-game-updates/ (Accessed: March 21, 2023).
- Darbinyan, R. (2022) *Council post: How artificial intelligence can empower the future of the gaming industry*, *Forbes*. Forbes Magazine. Available at: https://www.forbes.com/sites/forbestechcouncil/2022/07/13/how-artificial-

intelligence-can-empower-the-future-of-the-gaming-industry/ (Accessed: March 21, 2023).

- Duetzmann, S. (2016) *Video game definition of the week: Sports games*, *RSS*. Available at: https://engagedfamilygaming.com/parent-resources/video-game-definition-week-sports-games/#:~:text=Sports%20games%20come%20in%20two,on%20realistic%20recreations%20of%20sport. (Accessed: March 20, 2023).
- Fröberg, J. and Håkansson, C. (2021) *Application of machine learning to construct advanced NPC behaviors in Unity 3D*. thesis. Available at: https://liu.diva-portal.org/smash/get/diva2:1604156/FULLTEXT01.pdf (Accessed: March 18, 2023).
- Emmanuel, Z. (2022) "Video Games and Consoles - UK - 2022." Available at: https://reports.mintel.com/display/1170557/?fromSearch=%3Ffreetext%3DVideo%2520games%26resultPosition%3D1 (Accessed: March 16, 2023).
- Eshuis, S. *et al.* (2023) "Player experience and enjoyment: A preliminary examination of differences in video game genre," *Simulation & Gaming*, pp. 1–2. Available at: https://doi.org/10.1177/10468781231158818.
- Hassibi, K. (2016) *Machine learning vs. traditional statistics: Different philosophies, different approaches*, *Data Science Central*. Available at: https://www.datasciencecentral.com/machine-learning-vs-traditional-statistics-different-philosophi-1/ (Accessed: March 10, 2023).
- Jung, A. (2022) *Machine learning the basics*, *Springer*. Singapore, Singapore: Springer. Available at: https://link.springer.com/book/10.1007/978-981-16-8193-6 (Accessed: March 8, 2023).
- Kumar, S. (2021) *5 techniques to work with imbalanced data in Machine Learning*, *Medium*. Towards Data Science. Available at: https://towardsdatascience.com/5-techniques-to-work-with-imbalanced-data-in-machine-learning-80836d45d30c#:~:text=Upsampling%20or%20Oversampling%20refers%20to,to%20create%20artificial%20data%20points. (Accessed: March 20, 2023).
- MasterClass (2021) *Learn about platform game: 7 examples of Platform Games - 2023*, *MasterClass*. Available at:

https://www.masterclass.com/articles/platform-game-explained (Accessed: March 20, 2023).

- Masui, T. (2022) *All you need to know about gradient boosting algorithm − Part 1. regression*, *Medium*. Towards Data Science. Available at: https://towardsdatascience.com/all-you-need-to-know-about-gradient-boosting-algorithm-part-1-regression-2520a34a502 (Accessed: March 20, 2023).
- Media, S. (2019) *Top Five reasons behind video games' soaring popularity*, *ShinyShiny*. Available at: https://www.shinyshiny.tv/2019/10/top-five-reasons-behind-video-games-soaring-popularity.html (Accessed: March 18, 2023).
- Murdoch, W.J. *et al.* (2019) "Definitions, methods, and applications in interpretable machine learning," *Proceedings of the National Academy of Sciences*, 116(44), pp. 22071–22080. Available at: https://doi.org/10.1073/pnas.1900654116.
- Nabi, J. (2019) *Machine learning -fundamentals*, *Medium*. Towards Data Science. Available at: https://towardsdatascience.com/machine-learning-basics-part-1-a36d38c7916 (Accessed: March 9, 2023).
- Patel, H. (2021) *What is feature engineering - importance, tools and techniques for machine learning*, *Medium*. Towards Data Science. Available at: https://towardsdatascience.com/what-is-feature-engineering-importance-tools-and-techniques-for-machine-learning-2080b0269f10 (Accessed: March 20, 2023).
- Pavlovic, D. (2020) *Video game genres everything you need to know: HP® Tech takes*, *Video Game Genres Everything You Need To Know | HP® Tech Takes*. Available at: https://www.hp.com/us-en/shop/tech-takes/video-game-genres (Accessed: March 20, 2023).
- Plarium (2022) *What is the difference between MMO & MMORPG? - plarium*, *plarium.com*. Available at: https://plarium.com/en/blog/difference-between-mmo-and-mmorpgs/ (Accessed: March 20, 2023).
- Pretty, E.J., Fayek, H.M. and Zambetta, F. (2023) "A case for personalized non-player character companion design," *International Journal of Human–Computer Interaction*, pp. 1–20. Available at: https://doi.org/10.1080/10447318.2023.2181125.

- Rajula, H.S. *et al.* (2020) “Comparison of conventional statistical methods with machine learning in medicine: Diagnosis, drug development, and treatment,” *Medicina*, 56(9), pp. 1–10. Available at: https://doi.org/10.3390/medicina56090455
- Rahman, S. (2021) *Intelligent NPCs with Unity's ML Agents Toolkit*, *Intelligent NPCs with Unity's ML Agents Toolkit - Graphics, Gaming, and VR blog - Arm Community blogs - Arm Community*. Arm. Available at: https://community.arm.com/arm-community-blogs/b/graphics-gaming-and-vr-blog/posts/intelligent-npcs-with-machine-learning (Accessed: March 18, 2023).
- Riedl, M. and Bulitko, V. (2021) “Interactive narrative: A novel application of artificial intelligence for computer games,” *Proceedings of the AAAI Conference on Artificial Intelligence*, 26(1), pp. 2160–2165. Available at: https://doi.org/10.1609/aaai.v26i1.8447.
- Risi, S. and Togelius, J. (2020) *Increasing generality in machine learning through procedural content generation*, *Nature News*. Nature Publishing Group. Available at: https://www.nature.com/articles/s42256-020-0208-z#:~:text=Procedural%20content%20generation%20(PCG)%20refers,PCG%20methods%20have%20been%20devised. (Accessed: March 21, 2023).
- Rouse, M. (2017) *Massively multiplayer online role-playing game*, *Techopedia*. Available at: https://www.techopedia.com/definition/1919/massively-multiplayer-online-role-playing-game-mmorpg#:~:text=A%20massively%20multiplayer%20online%20role,takes%20place%20is%20never%20static. (Accessed: March 21, 2023).
- Ruta, D. *et al.* (2021) “Automated feature engineering for prediction of victories in Online Computer Games,” *2021 IEEE International Conference on Big Data (Big Data)* [Preprint]. Available at: https://doi.org/10.1109/bigdata52589.2021.9671345.
- Saltzman, M. (2022) *More adults play video games than kids – and more surprising stats*, *USA Today*. Gannett Satellite Information Network. Available at: https://eu.usatoday.com/story/tech/gaming/2022/06/11/gaming-study-finds-adults-play-more-than-

kids/7581101001/#:~:text=The%20popular%20reasons%20are%20that,%2C%E2%80%9D%20Pierre%2DLouis%20says. (Accessed: March 18, 2023).

- Schulman, J. *et al.* (2017) *Proximal policy optimization*, *Proximal Policy Optimization*. Open AI. Available at: https://openai.com/research/openai-baselines-ppo (Accessed: March 18, 2023).
- Smith, N. (2023) *The best MMORPG and top mmos you should play 2023*, *PCGamesN*. Available at: https://www.pcgamesn.com/10-best-pc-mmos (Accessed: March 20, 2023).
- Stephenson, J. (2019) *6 ways machine learning will be used in game development*, *Logikk*. Available at: https://www.logikk.com/articles/machine-learning-in-game-development/ (Accessed: March 18, 2023).
- Stewart, P.D.M. (2020) *The actual difference between statistics and machine learning*, *Medium*. Towards Data Science. Available at: https://towardsdatascience.com/the-actual-difference-between-statistics-and-machine-learning-64b49f07ea3 (Accessed: March 9, 2023).
- Sun, S. *et al.* (2019) "A Survey of Optimization Methods from a Machine Learning Perspective," *Arxiv*, p. 1. Available at: https://arxiv.org/pdf/1906.06821.pdf (Accessed: March 9, 2023).
- Takano, S. (2021) *Thinking machines machine learning and its hardware implementation*, *ScienceDirect*. London, United Kingdom: Academic Press. Available at: https://doi.org/10.1016/C2018-0-02470-6 (Accessed: March 8, 2023).
- Tao, J. *et al.* (2020) "Xai-driven explainable Multi-view game cheating detection," *2020 IEEE Conference on Games (CoG)*, pp. 144–151. Available at: https://doi.org/10.1109/cog47356.2020.9231843 (Accessed: March 20, 2023).
- Thompson, N.C. *et al.* (2022) *The Computational Limits Of Deep Learning*, p. 1. Available at: https://arxiv.org/pdf/2007.05558.pdf (Accessed: March 10, 2023).
- Wald, H. (2022) *The 25 best party games to play at home*, *gamesradar*. GamesRadar+. Available at: https://www.gamesradar.com/best-party-games/ (Accessed: March 20, 2023).

- Xiao, L. *et al.* (2020) “Neural supersampling for real-time rendering,” *ACM Transactions on Graphics*, 39(4). Available at: https://doi.org/10.1145/3386569.3392376.
- Yannakakis, G.N. and Togelius, J. (2018) *Artificial Intelligence and games*. Cham, Switzerland: Springer, pp. 15-20, 25–26. Available at: https://link.springer.com/content/pdf/bfm:978-3-319-63519-4/1 (Accessed: March 19, 2023).
- Zhu, X. (2022) *RNN Language Processing Model-driven spoken dialogue system modeling method*, *Computational intelligence and neuroscience*. U.S. National Library of Medicine. Available at: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8898104/ (Accessed: March 21, 2023).

B. Part-2: Build a Data Science Prediction Model

1. Import Packages

```
# import required libraries
import numpy as np
import matplotlib.pyplot as pyplt
import pandas as pd
import seaborn as sns
from sklearn.preprocessing import RobustScaler
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
from sklearn.metrics import accuracy_score, f1_score
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.svm import SVC
```

2. Read Data

The data is loaded from the csv file to a data frame, data_df.

```
## Step-1 : Load data ##
data_df=pds.read_csv("data.csv")
# show the dataset
print(data_df)
# print a return space
print('\n')
# show the keys
print(data_df.keys())
```

```
      Gender Home Location Level of Education  Age(Years)  Number of Subjects  \
0       Male          Urban     Under Graduate          18                  11
1       Male          Urban     Under Graduate          19                   7
2       Male          Rural     Under Graduate          18                   5
3       Male          Urban     Under Graduate          18                   5
4       Male          Rural     Under Graduate          18                   5
...      ...            ...                ...         ...                 ...
1028  Female          Rural     Under Graduate          19                   3
1029  Female          Rural     Under Graduate          20                   6
1030  Female          Rural     Under Graduate          20                   3
1031  Female          Rural     Under Graduate          19                   6
1032  Female          Urban     Under Graduate          20                   6

     Device type used to attend classes Economic status  Family size  \
0                                Laptop    Middle Class            4
1                                Laptop    Middle Class            4
2                                Laptop    Middle Class            5
3                                Laptop    Middle Class            4
4                                Laptop    Middle Class            4
...                                 ...             ...          ...
1028                             Mobile    Middle Class            6
1029                            Desktop    Middle Class            3
1030                             Mobile    Middle Class            3
1031                             Mobile    Middle Class            2
1032                             Laptop    Middle Class            4
```

```
      Internet facility in your locality Are you involved in any sports?  ...  \
0                                      5                              No  ...
1                                      1                             Yes  ...
2                                      2                              No  ...
3                                      4                             Yes  ...
4                                      3                              No  ...
...                                  ...                             ...  ...
1028                                   5                             Yes  ...
1029                                   1                              No  ...
1030                                   2                              No  ...
1031                                   3                             Yes  ...
1032                                   3                              No  ...

     Time spent on social media (Hours)  Interested in Gaming?  \
0                                     1                    No
1                                     1                   Yes
2                                     1                    No
3                                     2                    No
4                                     2                   Yes
...                                 ...                   ...
1028                                  4                   Yes
1029                                  2                   Yes
1030                                  3                    No
1031                                  2                   Yes
1032                                  1                   Yes

      Have separate room for studying?  Engaged in group studies?  \
0                                   No                         No
1                                  Yes                         No
2                                  Yes                         No
3                                   No                        yes
4                                  Yes                        yes
...                                ...                        ...
1028                               Yes                        yes

     Average marks scored before pandemic in traditional classroom  \
0                                                 91-100
1                                                 91-100
2                                                  71-80
3                                                 91-100
4                                                  81-90
...                                                  ...
1028                                              91-100
1029                                               71-80
1030                                               71-80
1031                                               61-70
1032                                               81-90

     Your interaction in online mode  \
0                                  1
1                                  1
2                                  1
3                                  1
4                                  3
...                              ...
1028                               4
1029                               5
1030                               2
1031                               3
1032                               5

     Clearing doubts with faculties in online mode Interested in?  \
0                                                1      Practical
1                                                1         Theory
2                                                1           Both
3                                                2         Theory
4                                                3           Both
...                                            ...            ...
1028                                             4         Theory
```

```
      Performance in online  Your level of satisfaction in Online Education
0                         6                                        Average
1                         3                                            Bad
2                         6                                            Bad
3                         4                                            Bad
4                         6                                        Average
...                     ...                                            ...
1028                      8                                        Average
1029                     10                                        Average
1030                      6                                        Average
1031                      6                                           Good
1032                     10                                        Average

[1033 rows x 23 columns]
```

```
Index(['Gender', 'Home Location', 'Level of Education', 'Age(Years)',
       'Number of Subjects', 'Device type used to attend classes',
       'Economic status', 'Family size', 'Internet facility in your locality',
       'Are you involved in any sports?', 'Do elderly people monitor you?',
       'Study time (Hours)', 'Sleep time (Hours)',
       'Time spent on social media (Hours)', 'Interested in Gaming?',
       'Have separate room for studying?', 'Engaged in group studies?',
       'Average marks scored before pandemic in traditional classroom',
       'Your interaction in online mode',
       'Clearing doubts with faculties in online mode', 'Interested in?',
       'Performance in online',
       'Your level of satisfaction in Online Education'],
      dtype='object')
```

3. Data Pre-processing

As part of data pre-processing, data is cleaned and transformed using the below code. The column names are lengthy and have space in between. So, the column names are renamed using rename function.

```
## Step-2 : Data cleaning and transformations ##
# renaming the column names since the column names are lengthy and delimited with space.
# referred from 'https://stackoverflow.com/questions/11346283/renaming-column-names-in-pandas'
data_df.rename(columns={'Gender': 'gender', 'Home Location': 'home_loc','Level of Education':'edu_level','Age(Years)':'age',
                        'Number of Subjects':'subject_count','Device type used to attend classes':'device_type',
                        'Economic status':'economic_status', 'Family size':'family_size',
                        'Internet facility in your locality':'internet_facility',
                        'Are you involved in any sports?':'active_in_sports',
                        'Do elderly people monitor you?':'monitered_by_elderly',
                        'Study time (Hours)':'study_time_hrs', 'Sleep time (Hours)':'sleep_time_hrs',
                        'Time spent on social media (Hours)':'social_media_hrs',
                        'Interested in Gaming?':'interested_in_gaming',
                        'Have separate room for studying?':'study_room',
                        'Engaged in group studies?':'group_studies',
                        'Average marks scored before pandemic in traditional classroom':'classroom_avg_score',
                        'Your interaction in online mode':'online_interaction',
                        'Clearing doubts with faculties in online mode':'online_doubt_clarification',
                        'Interested in?':'interested_in','Performance in online':'performance_online',
                        'Your level of satisfaction in Online Education':'satisfaction_level'}, inplace=True)
# show the keys
print(data_df.keys())
```

```
Index(['gender', 'home_loc', 'edu_level', 'age', 'subject_count',
       'device_type', 'economic_status', 'family_size', 'internet_facility',
       'active_in_sports', 'monitered_by_elderly', 'study_time_hrs',
       'sleep_time_hrs', 'social_media_hrs', 'interested_in_gaming',
       'study_room', 'group_studies', 'classroom_avg_score',
       'online_interaction', 'online_doubt_clarification', 'interested_in',
       'performance_online', 'satisfaction_level'],
      dtype='object')
```

In the next step, the values of the columns are converted into numerical values.

```
 # reassigned gender with numerical values.
 gender = {'Male': 1,'Female': 0}
 data_df.gender = [gender[item] for item in data_df.gender]

 # reassigned home location with numerical values.
 home_loc = {'Urban': 1,'Rural': 0}
 data_df.home_loc = [home_loc[item] for item in data_df.home_loc]

 # reassigned level of education with numerical values.
 edu_level = {'School': 0,'Under Graduate':1,'Post Graduate':2}
 data_df.edu_level = [edu_level[item] for item in data_df.edu_level]


# reassigned device type with numerical values.
# created new dataframe with device type category columns - Desktop, Laptop, Mobile.
device_df=pd.get_dummies(data_df, columns=["device_type"], prefix=["device_type"])
print(device_df)
# megred newly created dataframe to the main dataframe.
data_df=data_df.merge(device_df)
# dropped the column for device type since new columns are already added to the dataframe.
data_df=data_df.drop(columns=['device_type'])

# reassigned economic status with numerical values.
eco_level = {'Poor': 0,'Middle Class':1,'Rich':2}
data_df.economic_status = [eco_level[item] for item in data_df.economic_status]

# dictionary for Yes or No values.
yes_no_dict = {'Yes': 1,'yes':1,'No': 0}

cols={'active_in_sports','monitered_by_elderly','interested_in_gaming','study_room','group_studies'}
for c in cols:
  data_df[c]=[yes_no_dict[item] for item in data_df[c]]
  print(c)
```

```
      gender  home_loc  edu_level  age  subject_count economic_status  \
0          1         1          1   18             11    Middle Class
1          1         1          1   19              7    Middle Class
2          1         0          1   18              5    Middle Class
3          1         1          1   18              5    Middle Class
4          1         0          1   18              5    Middle Class
...      ...       ...        ...  ...            ...             ...
1028       0         0          1   19              3    Middle Class
1029       0         0          1   20              6    Middle Class
1030       0         0          1   20              3    Middle Class
1031       0         0          1   19              6    Middle Class
1032       0         1          1   20              6    Middle Class
```

```
      family_size  internet_facility active_in_sports monitered_by_elderly  \
0               4                  5              No                  Yes
1               4                  1             Yes                  Yes
2               5                  2              No                  Yes
3               4                  4             Yes                  Yes
4               4                  3              No                   No
...           ...                ...             ...                  ...
1028            6                  5             Yes                  Yes
1029            3                  1              No                   No
1030            3                  2              No                   No
1031            2                  3             Yes                   No
1032            4                  3              No                   No

      ...  group_studies  classroom_avg_score  online_interaction  \
0     ...             No               91-100                   1
1     ...             No               91-100                   1
2     ...             No                71-80                   1
3     ...            yes               91-100                   1
4     ...            yes                81-90                   3
...   ...            ...                  ...                 ...
1028  ...            yes               91-100                   4
1029  ...             No                71-80                   5
1030  ...            yes                71-80                   2
1031  ...             No                61-70                   3
1032  ...            yes                81-90                   5

     online_doubt_clarification interested_in performance_online  \
0                             1     Practical                   6
1                             1        Theory                   3
2                             1          Both                   6
3                             2        Theory                   4
4                             3          Both                   6
...                         ...           ...                 ...
1028                          4        Theory                   8
     satisfaction_level  device_type_Desktop  device_type_Laptop  \
0               Average                    0                   1
1                   Bad                    0                   1
2                   Bad                    0                   1
3                   Bad                    0                   1
4               Average                    0                   1
...                 ...                  ...                 ...
1028            Average                    0                   0
1029            Average                    1                   0
1030            Average                    0                   0
1031               Good                    0                   0
1032            Average                    0                   1

       device_type_Mobile
  0                       0
  1                       0
  2                       0
  3                       0
  4                       0
  ...                   ...
  1028                    1
  1029                    0
  1030                    1
  1031                    1
  1032                    0

  [1033 rows x 25 columns]
  interested_in_gaming
  group_studies
  active_in_sports
  monitered_by_elderly
  study_room
```

The input dataset contains both categorical and ordinal data. The columns with ordinal data is converted to numerical values based on the order of values. For example, level of education (values: School, Under Graduate, Post Graduate) is encoded as follows: School=0, Under Graduate=1, Post Graduate=2.

The columns gender and device_type are examples of categorical data. When converting categorical data into numerical values, one hot encoding technique is usually used because categorical data have no order. One hot encoding will increase dimensions and degrade the model training performance. Therefore, columns with two category values are encoded as 1 and 0, while columns with more than two category values are encoded with one hot encoding. The above code converts the values of gender (values: female, male) column into 1 and 0 and replace device_type (values: Desktop, Mobile, Laptop) column with three columns - device_type_Mobile, device_type_Desktop, and device_type_Laptop.

It was found that one category is listed as ‘Nov-20’ instead of ’11-20’ in the classroom_avg_score column and this issues is resolved by assigning it with correct order value.

```
print('Avg Score:')
print(data_df['classroom_avg_score'].value_counts())
```

Python

```
Avg Score:
81-90     343
71-80     313
91-100    158
61-70     118
51-60      52
41-50      24
31-40      11
21-30       7
Nov-20      6
0-10        1
Name: classroom_avg_score, dtype: int64
```

```
# reassigned classroom_avg_score with numerical values.
avg_score_classes = {'0-10': 0,'Nov-20':1,'21-30':2,'31-40':3,'41-50':4,'51-60':5,'61-70':6,'71-80':7,'81-90':8,'91-100':9}
data_df.classroom_avg_score = [avg_score_classes[item] for item in data_df.classroom_avg_score]
```

```
# reassigned interested_in with numerical values.
# created new dataframe with interested_in category columns - Theory, Practical, Both
interested_in_df=pd.get_dummies(data_df, columns=["interested_in"], prefix=["interested_in"])
print(interested_in_df)
# megred newly created dataframe to the main dataframe.
data_df=data_df.merge(interested_in_df)
# dropped the column for interested_in since new columns are already added to the dataframe.
data_df=data_df.drop(columns=['interested_in'])

# reassigned satisfaction_level with numerical values.
sat_level = {'Bad':0,'Average':1,'Good':2}
data_df.satisfaction_level = [sat_level[item] for item in data_df.satisfaction_level]
print(data_df.satisfaction_level.value_counts())

# print dataframe
print(data_df)
```

```
      gender  home_loc  edu_level  age  subject_count  economic_status  \
0          1         1          1   18             11                1
1          1         1          1   19              7                1
2          1         0          1   18              5                1
3          1         1          1   18              5                1
4          1         0          1   18              5                1
...      ...       ...        ...  ...            ...              ...
1028       0         0          1   19              3                1
1029       0         0          1   20              6                1
1030       0         0          1   20              3                1
1031       0         0          1   19              6                1
1032       0         1          1   20              6                1

      family_size  internet_facility  active_in_sports  monitered_by_elderly  \
0               4                  5                 0                     1
1               4                  1                 1                     1
2               5                  2                 0                     1
3               4                  4                 1                     1
4               4                  3                 0                     0
...           ...                ...               ...                   ...
1028            6                  5                 1                     1
1029            3                  1                 0                     0
1030            3                  2                 0                     0
1031            2                  3                 1                     0
1032            4                  3                 0                     0

      ...  online_interaction  online_doubt_clarification  performance_online  \
0     ...                   1                           1                   6
1     ...                   1                           1                   3
2     ...                   1                           1                   6
3     ...                   1                           2                   4
4     ...                   3                           3                   6
...   ...                 ...                         ...                 ...
1028  ...                   4                           4                   8
1029                        5                           5                  10

     satisfaction_level  device_type_Desktop  device_type_Laptop  \
0               Average                    0                   1
1                   Bad                    0                   1
2                   Bad                    0                   1
3                   Bad                    0                   1
4               Average                    0                   1
...                 ...                  ...                 ...
1028            Average                    0                   0
1029            Average                    1                   0
1030            Average                    0                   0
1031               Good                    0                   0
1032            Average                    0                   1

     device_type_Mobile  interested_in_Both  interested_in_Practical  \
0                     0                   0                        1
1                     0                   0                        0
2                     0                   1                        0
3                     0                   0                        0
4                     0                   1                        0
...                 ...                 ...                      ...
1028                  1                   0                        0
1029                  0                   0                        0
1030                  1                   0                        0
1031                  1                   1                        0
1032                  0                   0                        1

      interested_in_Theory
0                        0
1                        1
2                        0
3                        1
4                        0
...                    ...
1028                     1
1029                     1
1030                     1
1031                     0
1032                     0

[1033 rows x 27 columns]
1    541
2    251
0    241
Name: satisfaction_level, dtype: int64
```

```
      gender  home_loc  edu_level  age  subject_count  economic_status  \
0          1         1          1   18             11                1
1          1         1          1   19              7                1
2          1         0          1   18              5                1
3          1         1          1   18              5                1
4          1         0          1   18              5                1
...      ...       ...        ...  ...            ...              ...
1028       0         0          1   19              3                1
1029       0         0          1   20              6                1
1030       0         0          1   20              3                1
1031       0         0          1   19              6                1
1032       0         1          1   20              6                1

      family_size  internet_facility  active_in_sports  monitered_by_elderly  \
0               4                  5                 0                     1
1               4                  1                 1                     1
2               5                  2                 0                     1
3               4                  4                 1                     1
4               4                  3                 0                     0
...           ...                ...               ...                   ...
1028            6                  5                 1                     1
1029            3                  1                 0                     0
1030            3                  2                 0                     0
1031            2                  3                 1                     0
1032            4                  3                 0                     0

      ...  online_interaction  online_doubt_clarification  performance_online  \
0     ...                   1                           1                   6
1     ...                   1                           1                   3
2     ...                   1                           1                   6
3     ...                   1                           2                   4
4     ...                   3                           3                   6
...   ...                 ...                         ...                 ...
1028  ...                   4                           4                   8
1029  ...                   5                           5                  10
1030  ...                   2                           2                   6
1031  ...                   3                           3                   6
1032  ...                   5                           4                  10

      satisfaction_level  device_type_Desktop  device_type_Laptop  \
0                      1                    0                   1
1                      0                    0                   1
2                      0                    0                   1
3                      0                    0                   1
4                      1                    0                   1
...                  ...                  ...                 ...
1028                   1                    0                   0
1029                   1                    1                   0
1030                   1                    0                   0
1031                   2                    0                   0
1032                   1                    0                   1

      device_type_Mobile  interested_in_Both  interested_in_Practical  \
0                      0                   0                        1
1                      0                   0                        0
2                      0                   1                        0
3                      0                   0                        0
4                      0                   1                        0
...                  ...                 ...                      ...
1028                   1                   0                        0
1029                   0                   0                        0
1030                   1                   0                        0
1031                   1                   1                        0
1032                   0                   0                        1


      interested_in_Theory
0                        0
1                        1
2                        0
3                        1
4                        0
...                    ...
1028                     1
1029                     1
1030                     1
1031                     0
1032                     0

[1033 rows x 27 columns]
```

```
# check for null values
print(data_df.isnull().sum())
```

```
gender                          0
home_loc                        0
edu_level                       0
age                             0
subject_count                   0
economic_status                 0
family_size                     0
internet_facility               0
active_in_sports                0
monitered_by_elderly            0
study_time_hrs                  0
sleep_time_hrs                  0
social_media_hrs                0
interested_in_gaming            0
study_room                      0
group_studies                   0
classroom_avg_score             0
online_interaction              0
online_doubt_clarification      0
performance_online              0
satisfaction_level              0
device_type_Desktop             0
device_type_Laptop              0
device_type_Mobile              0
interested_in_Both              0
interested_in_Practical         0
interested_in_Theory            0
dtype: int64
```

4. Feature Engineering

```
## Step-3 : Feature Engineering ##
# highly interactive online students may have good satisfaction level.
data_df['high_interactive_stud_online']=np.where((data_df['online_doubt_clarification'] > 3)&(data_df['online_interaction'] > 3) &
                                                 (data_df['performance_online'] > 5)&(data_df['study_time_hrs']>4),1,0)
# small family with internet
# As family size increases the wifi speed may decrease since more people are simultaneously using it, resulting in poor satisfaction level.
data_df['small_family_with_internet']=np.where((data_df['family_size']<5)&(data_df['internet_facility']==1),1,0)
# students with laptop and internet facility
data_df['laptop_internet_user']=np.where((data_df['device_type_Laptop'] == 1) & (data_df['internet_facility'] == 1),1,0)
# students with desktop and internet facility
data_df['desktop_internet_user']=np.where((data_df['device_type_Desktop'] == 1) & (data_df['internet_facility'] == 1),1,0)
# students with mobile and internet facility
data_df['mobile_internet_user']=np.where((data_df['device_type_Mobile'] == 1) & (data_df['internet_facility'] == 1),1,0)
# postgraduate students
data_df['pg_stud']=np.where(data_df['edu_level']==2,1,0)
```

In this step, a few features are derived from the existing columns. For example, high_interactive_stud_online is the column value indicates the student is highly interactive in online sessions which might lead to higher satisfaction level. Based on the values of the columns – online_doubt_clarification, online_interaction, performance_online and study_time_hrs, the value is set to the derived column,

high_interactive_stud_online. If the student performs above average then the student is highly interactive in online sessions.

```
[7]  print(data_df['online_doubt_clarification'].value_counts())
```

```
3    375
2    213
4    189
1    165
5     91
Name: online_doubt_clarification, dtype: int64
```

```
[8]  print(data_df['online_interaction'].value_counts())
```

```
3    433
4    189
2    189
1    129
5     93
Name: online_interaction, dtype: int64
```

```
print(data_df['performance_online'].value_counts())
```

```
6     244
8     218
7     181
5      92
10     87
4      78
9      73
2      37
3      23
Name: performance_online, dtype: int64
```

```
print(data_df['study_time_hrs'].value_counts())
```

```
4     213
3     181
5     150
2     144
6     109
1      73
8      68
7      58
10     25
9      12
Name: study_time_hrs, dtype: int64
```

Similarly, another feature, small_family_with_internet is derived from the family_size and internet_facility columns. If the family size is big and have internet facility, then there are chances of getting poor network bandwidth. So, the student belonging to a small family with internet facility may have better bandwidth and may show better satisfaction level in online classes.

Internet facility is one of the key factors that helps the students to have better experience in online sessions. Therefore, based on the device type and internet

facility, new features are derived. Furthermore, a few more attributes are considered in the feature list.

```
features_list=['study_time_hrs','pg_stud','small_family_with_internet','laptop_internet_user',
               'desktop_internet_user','mobile_internet_user','online_interaction','online_doubt_clarification',
               'performance_online','high_interactive_stud_online','internet_facility','monitered_by_elderly',
               'interested_in_Theory','age','home_loc','group_studies',]
```

4.1. Data Normalisation

As discussed earlier, the values of the different columns are in different scale. It is good to normalise the data for better model accuracy and performance. The following code used RobustScaler for normalisation because it performs well with data with outliers.

```
## data normalisation ##
scaler = RobustScaler()
# get the column names
col_names = data_df.columns
df = scaler.fit_transform(data_df)
result_df = pd.DataFrame(df, columns=col_names)
print(result_df)
```

```
      gender  home_loc  edu_level  age  subject_count  economic_status  \
0        0.0       0.0        0.0 -0.5            2.0              0.0
1        0.0       0.0        0.0  0.0            0.0              0.0
2        0.0      -1.0        0.0 -0.5           -1.0              0.0
3        0.0       0.0        0.0 -0.5           -1.0              0.0
4        0.0      -1.0        0.0 -0.5           -1.0              0.0
...      ...       ...        ...  ...            ...              ...
1028    -1.0      -1.0        0.0  0.0           -2.0              0.0
1029    -1.0      -1.0        0.0  0.5           -0.5              0.0
1030    -1.0      -1.0        0.0  0.5           -2.0              0.0
1031    -1.0      -1.0        0.0  0.0           -0.5              0.0
1032    -1.0       0.0        0.0  0.5           -0.5              0.0

      family_size  internet_facility  active_in_sports  monitered_by_elderly  \
0             0.0                1.0               0.0                   0.0
1             0.0               -3.0               1.0                   0.0
2             1.0               -2.0               0.0                   0.0
3             0.0                0.0               1.0                   0.0
4             0.0               -1.0               0.0                  -1.0
...           ...                ...               ...                   ...
1028          2.0                1.0               1.0                   0.0
1029         -1.0               -3.0               0.0                  -1.0
1030         -1.0               -2.0               0.0                  -1.0
1031         -2.0               -1.0               1.0                  -1.0
1032          0.0               -1.0               0.0                  -1.0

      ...  device_type_Mobile  interested_in_Both  interested_in_Practical  \
0     ...                0.0                 0.0                      1.0
1     ...                0.0                 0.0                      0.0
2     ...                0.0                 1.0                      0.0
3     ...                0.0                 0.0                      0.0
4     ...                0.0                 1.0                      0.0
```

```
      interested_in_Theory  high_interactive_stud_online  \
0                      0.0                           0.0
1                      1.0                           0.0
2                      0.0                           0.0
3                      1.0                           0.0
4                      0.0                           0.0
...                    ...                           ...
1028                   1.0                           0.0
1029                   1.0                           0.0
1030                   1.0                           0.0
1031                   0.0                           0.0
1032                   0.0                           0.0

      small_family_with_internet  laptop_internet_user  desktop_internet_user  \
0                            0.0                   0.0                    0.0
1                            1.0                   1.0                    0.0
2                            0.0                   0.0                    0.0
3                            0.0                   0.0                    0.0
4                            0.0                   0.0                    0.0
...                          ...                   ...                    ...
1028                         0.0                   0.0                    0.0
1029                         1.0                   0.0                    1.0
1030                         0.0                   0.0                    0.0
1031                         0.0                   0.0                    0.0
1032                         0.0                   0.0                    0.0

      mobile_internet_user  pg_stud
0                      0.0      0.0
1                      0.0      0.0
2                      0.0      0.0
3                      0.0      0.0
4                      0.0      0.0
...                    ...      ...
1028                   0.0      0.0
```

4.2. Separate the Target (y) from Features (X)

Firstly, define X with the identified features, features_list and then define y with the target variable, satisfaction_level.

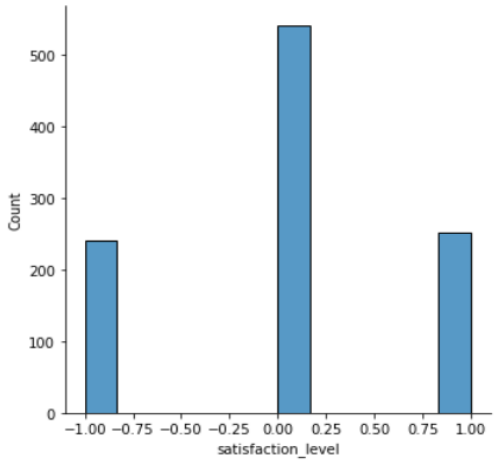
```python
# define X
# satisfaction_level is the target variable, hence removed from X.
X=result_df[features_list]
print(X.dtypes)
```

```
study_time_hrs                float64
pg_stud                       float64
small_family_with_internet    float64
laptop_internet_user          float64
desktop_internet_user         float64
mobile_internet_user          float64
online_interaction            float64
online_doubt_clarification    float64
performance_online            float64
high_interactive_stud_online  float64
internet_facility             float64
monitered_by_elderly          float64
interested_in_Theory          float64
age                           float64
home_loc                      float64
group_studies                 float64
dtype: object
```

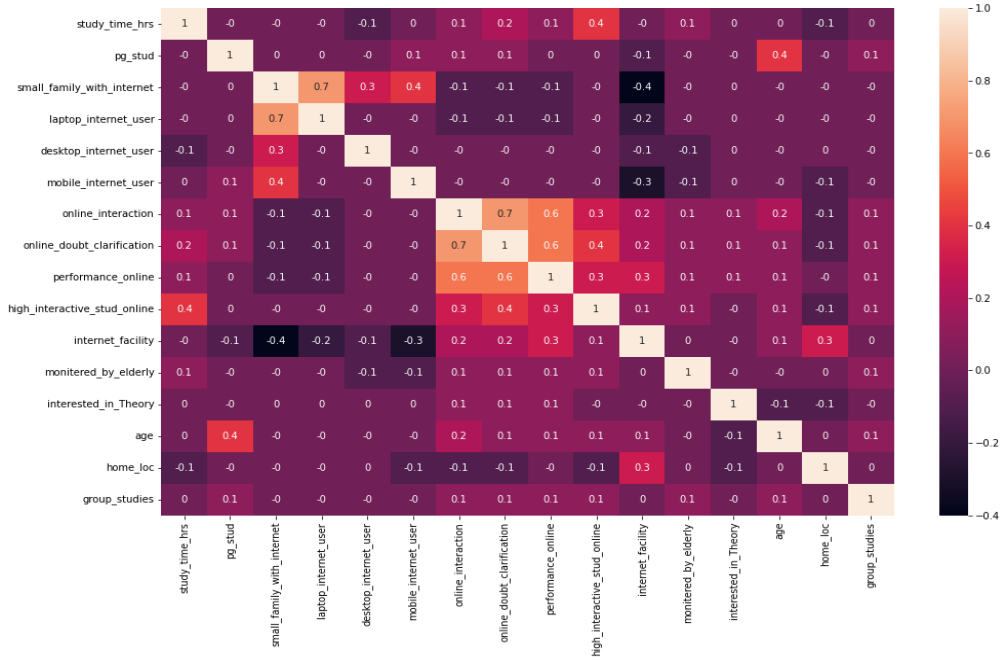
```
# define y
# satisfaction_level is the target variable, hence added to y.
y=result_df['satisfaction_level']
print(y.dtypes)
```

4.3. Plot the features using bar chart and correlation matrix

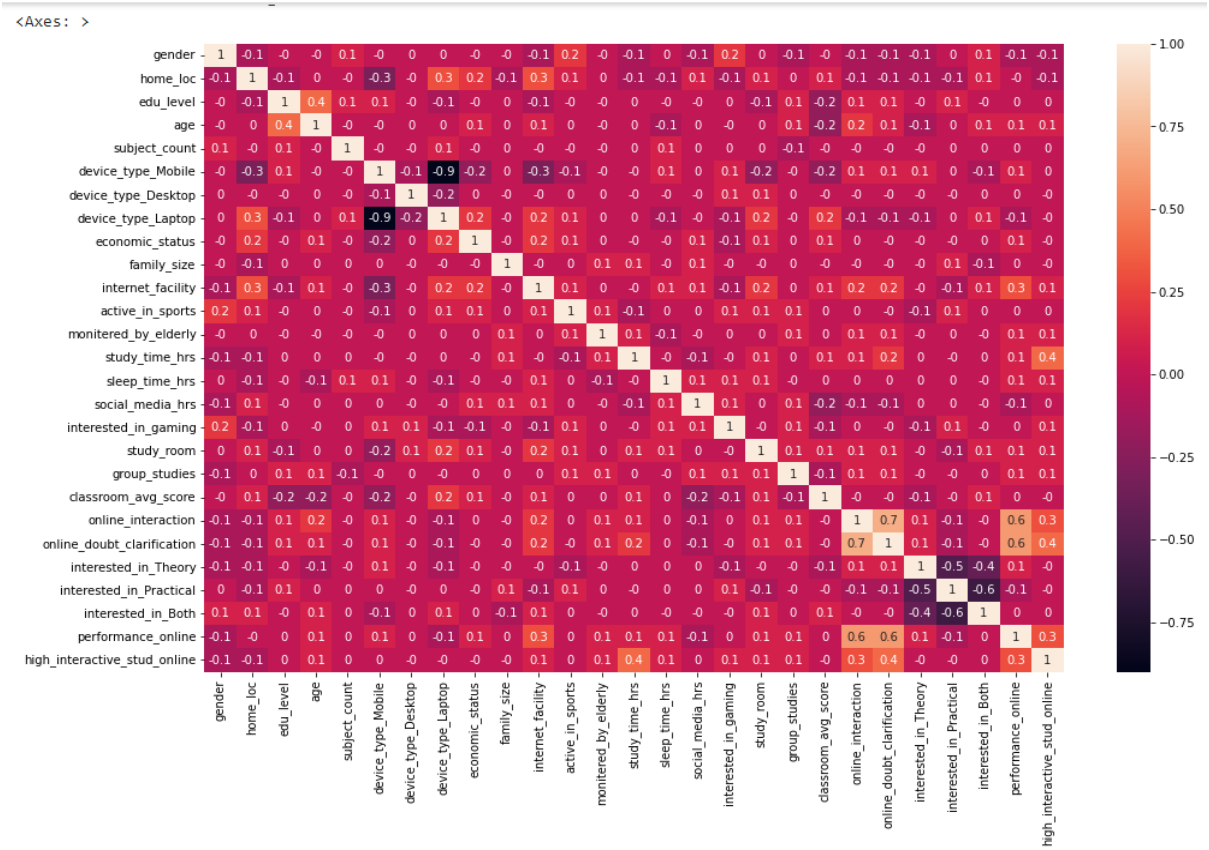
```
## plot correlation matrix and bar chart ##
ax = sns.displot(y)
# print bar chart
pyplt.show()
# increase the size of the map for better readability.
# referred from "https://datascience.stackexchange.com/questions/17540/make-seaborn-heatmap-bigger"
pyplt.figure(figsize=(20,15))
# create a correlation matrix rounding to one decimal point
correlation_matrix = result_df[features_list].corr().round(1)
# print a correlation heat map
sns.heatmap(data=correlation_matrix,annot=True)
```



<Axes: >



In the above correlation matrix, the features considered are showing very little correlation to the target value, satisfaction_level. So, another correlation matrix using all the attributes is constructed below. It indicates that most attributes have very less correlation to the target variable. So, it is better to keep all attributes rather than selecting a few attributes for better model accuracy. Furthermore, the input dataset is very small and doesn't have much variety of data.



The code changes made in the previous steps to construct the above correlation matrix with all attributes is shown below:

```
## Step-4 : Separate the target value ##
# define X
# satisfaction_level is the target variable, hence removed from X.
#X=result_df.drop(columns=['satisfaction_level'],axis=1)
selected_features=['gender', 'home_loc', 'edu_level', 'age', 'subject_count',
       'device_type_Mobile','device_type_Desktop','device_type_Laptop',
       'economic_status', 'family_size', 'internet_facility',
       'active_in_sports', 'monitered_by_elderly', 'study_time_hrs',
       'sleep_time_hrs', 'social_media_hrs', 'interested_in_gaming',
       'study_room', 'group_studies', 'classroom_avg_score',
       'online_interaction', 'online_doubt_clarification',
       'interested_in_Theory','interested_in_Practical','interested_in_Both',
       'performance_online','high_interactive_stud_online']
X=result_df[selected_features]
print(X.dtypes)
```

```
gender                          float64
home_loc                        float64
edu_level                       float64
age                             float64
subject_count                   float64
device_type_Mobile              float64
device_type_Desktop             float64
device_type_Laptop              float64
economic_status                 float64
family_size                     float64
internet_facility               float64
active_in_sports                float64
monitered_by_elderly            float64
study_time_hrs                  float64
sleep_time_hrs                  float64
social_media_hrs                float64
interested_in_gaming            float64
study_room                      float64
group_studies                   float64
classroom_avg_score             float64
online_interaction              float64
online_doubt_clarification      float64
interested_in_Theory            float64
interested_in_Practical         float64
interested_in_Both              float64
performance_online              float64
high_interactive_stud_online    float64
dtype: object
```

```
## plot correlation matrix and bar chart ##
ax = sns.displot(y)
# print bar chart
pyplt.show()
# increase the size of the map for better readability.
# referred from "https://datascience.stackexchange.com/questions/17540/make-seaborn-heatmap-bigger"
pyplt.figure(figsize=(16,10))
# create a correlation matrix rounding to one decimal point
correlation_matrix = result_df[selected_features].corr().round(1)
# print a correlation heat map
sns.heatmap(data=correlation_matrix,annot=True)
```

5. Split data into training and test datasets

In this step, the input dataset is split into two sets for model training and testing purposes. As per Pareto principle,80 percent of input data is allocated for training while 20 percent is allocated for testing purpose. Hence, test_size parameter is set to 0.2, indicates 80:20 ratio.

```
## Step-5 : Split the data into training and test sets ##
# split data into training and test
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.2)
# print the shapes to check everything is OK
print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
```

In the above code, X_train and y_train variables indicate training dataset whereas X_test and y_test variables are the testing dataset. The shape of the variables is

checked to confirm the number of input values and number of features are properly allotted.

```
(826, 27)
(207, 27)
(826,)
(207,)
```

6. Select the ML Algorithms

This prediction problem is a multi-class classification problem since the target value, satisfaction_level can have three values – Poor, Bad or Good. Furthermore, it is important to understand the features influencing the satisfaction_level of students to take appropriate actions to enhance the student experience in online sessions. So, black-box algorithms such as Deep Learning techniques is not suitable for this problem because it doesn't give explanation for the final decision.

As discussed in the previous section, the correlation matrix is showing similar weightage for most features in the data. Therefore, the ML algorithms that can provide the measure of feature importance with better accuracy and performance should be considered in this case. The ML algorithms that are under consideration are listed below:

- **Random Forest Classifier (RFC)**: an ensemble learning method that combines multiple decision trees to enhance the prediction accuracy of model (Shafi,2018). It is suitable for multi-classification problems as well as binary classification problems, and it performs well with categorical and numerical data(ibid). Moreover, it measures features based on their importance and is resistant to multicollinearity and overfitting(ibid). In addition to being easy to use, it requires little data pre-processing, and it works faster because it trains the model in parallel(ibid).
- **Gradient Boosting Classifier (GBC)**: another ensemble learning technique that combine multiple weaker models such as decision trees to enhance the prediction accuracy of models(Aliyev,2020). During training, each predictor attempts to decrease residual errors of the previous predictor in order to improve the result produced by the previous predictor (ibid). Like Random Forest Classifier, it measures the importance of features, is resistant to

multicollinearity and overfitting, and is best suited to multiclass classification problems.

- **Support Vector Machine Classifier (SVC)**: an ML model that classify data points by identifying an optimal hyperplane in multi-dimensional space that separates these points(Gandhi,2018). It works well with small datasets and capable of capturing complex relationships between variables.

7. Hyper Parameter Optimisation

Hyper parameter optimisation is important to obtain optimal results from ML models. The F1 score is used to evaluate the accuracy of the ML model since the given problem is a classification problem. The hyper parameter optimisation for chosen three ML algorithms will be discussed in the following sections.

7.1. Hyper Parameter Optimisation for RFC

```
## Step-7 : Hyper parameter optimisation ##
tuned_parameters = [{'n_estimators': [10, 20, 30],
                     'criterion': ['gini', 'entropy'],
                     'max_depth': [3, 5, 7, 10],
                     'max_features': ['sqrt', 'log2', None]
                     }]

scores = ['f1_macro']

for score in scores:
    print("# Tuning hyperparameters for %s" % score)
    print("\n")
    rfc = GridSearchCV(RandomForestClassifier(), tuned_parameters, cv=5,
                       scoring=score)
    rfc.fit(X_train, y_train)
    print("Best parameters set found on the training set:")
    print(rfc.best_params_)
    print("\n")
```

```
# Tuning hyperparameters for f1_macro


Best parameters set found on the training set:
{'criterion': 'gini', 'max_depth': 10, 'max_features': 'log2', 'n_estimators': 30}
```

7.2. Hyper Parameter Optimisation for GBC

```
## Step-7 : Hyper parameter optimisation ##
## referred from https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html
tuned_parameters = [{'n_estimators' :[100,110],
                     'criterion': ['friedman_mse', 'squared_error'],
                     'max_depth': [5, 7, 10],
                     'min_samples_split':[7,10],
                     'max_features': ['sqrt', 'log2', None]
                     }]

scores = ['f1_macro']

for score in scores:
    print("# Tuning hyperparameters for %s" % score)
    print("\n")
    gb = GridSearchCV(GradientBoostingClassifier(), tuned_parameters, cv=5,
                      scoring=score)
    gb.fit(X_train, y_train)
    print("Best parameters set found on the training set:")
    print(gb.best_params_)
    print("\n")
```

```
# Tuning hyperparameters for f1_macro


Best parameters set found on the training set:
{'criterion': 'friedman_mse', 'max_depth': 10, 'max_features': 'log2', 'min_samples_split': 10, 'n_estimators': 110}
```

7.3. Hyper Parameter Optimisation for SVC

```
## Step-7 : Hyper parameter optimisation ##
from sklearn.svm import SVC
## referred from https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html
tuned_parameters = [{'C' :[0.1, 1, 10, 100],
                     'kernel': ['linear', 'poly','rbf'],
                     'gamma': ['scale','auto']
                     }]

scores = ['f1_macro']

for score in scores:
    print("# Tuning hyperparameters for %s" % score)
    print("\n")
    svc = GridSearchCV(SVC(), tuned_parameters, cv=5,
                       scoring=score)
    svc.fit(X_train, y_train)
    print("Best parameters set found on the training set:")
    print(svc.best_params_)
    print("\n")
```

```
# Tuning hyperparameters for f1_macro


Best parameters set found on the training set:
{'C': 0.1, 'gamma': 'scale', 'kernel': 'linear'}
```

8. Build ML model

Having identified the optimal values for the hyperparameters using GridSearchCV, the next step is to build the ML models using the selected algorithms.

8.1. RFC Model

```
## Step-8 : Build model ##
# define the RFC model
rfc_model = RandomForestClassifier(criterion='gini', max_depth=10, max_features='log2', n_estimators=30)
# fit the RFC model
rfc_model_fit = rfc_model.fit(X_train, y_train)
```

8.2. GBC Model

```
## Step-8 : Build model ##
# define the GB model
gb_model = GradientBoostingClassifier(criterion='friedman_mse', max_depth=10, max_features='log2',
                                      min_samples_split=10, n_estimators=110)
# fit the GB model
gb_model_fit = gb_model.fit(X_train, y_train)
```

8.3. SVC Model

```
## Step-8 : Build model ##
# define the SVC model
svc_model = SVC(C= 0.1,gamma='scale',kernel='linear')
# fit the SVC model
svc_model_fit = svc_model.fit(X_train, y_train)
```

9. Data Prediction

The next step is to predict the target variable of the test data using the models built.

9.1. Prediction using RFC model

```
## Step-9 : Predict the test data ##
# predict the data
rfc_predict = rfc_model_fit.predict(X_test)
```

9.2. Prediction using GBC model

```
## Step-9 : Predict the test data ##
gb_predict = gb_model_fit.predict(X_test)
```

9.3. Prediction using SVC model

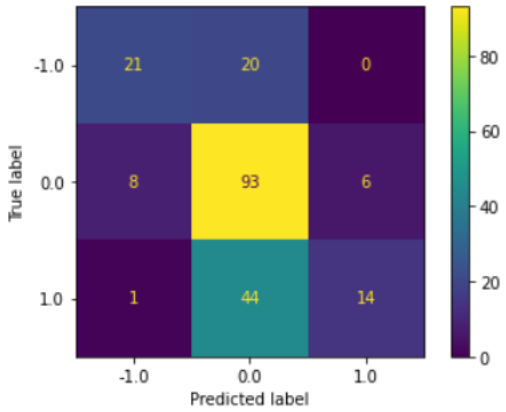
```
## Step-9 : Predict the test data ##
svc_predict = svc_model_fit.predict(X_test)
```

10. Model Evaluation

In this step, the accuracy of the three models is evaluated using F1 score.

10.1. RFC Model Evaluation

```
## Step-10 :Evaluate performance ##
# build and plot a confusion matrix
cm = confusion_matrix(y_test, rfc_predict, labels=rfc_model.classes_)
cm_disp = ConfusionMatrixDisplay(confusion_matrix=cm,
                                 display_labels=rfc_model.classes_)
cm_disp.plot()
pyplt.show()
```




```
# calculate metrics
acc = accuracy_score(y_test, rfc_predict)
f1_mac = f1_score(y_test, rfc_predict, average='macro')

# print results
print(f'Accuracy = {round(acc,2)}%')
print(f'F1 score = {round(f1_mac,2)}')
```

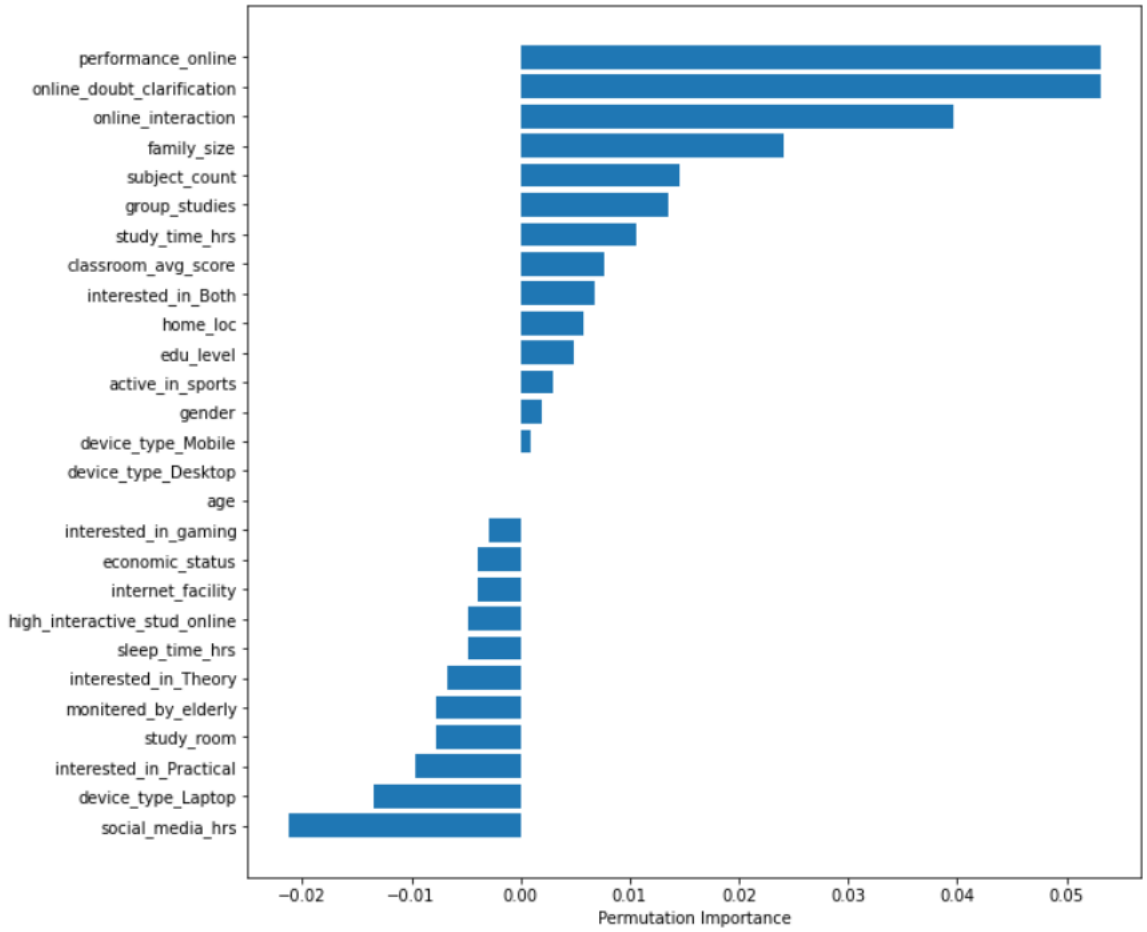
```
Accuracy = 0.61%
F1 score = 0.56
```

The feature importance metrics is calculated using permutation-based feature importance and the same is plotted using bar chart. Permutation-based feature importance method is not biased towards high-cardinality features and make good predictions than mean decrease in impurity (MDI) method.

```
# referred from https://mljar.com/blog/feature-importance-in-random-forest/
from sklearn.inspection import permutation_importance
print(rfc_model_fit.feature_importances_)
pyplt.figure(figsize=(10,10))
perm_importance = permutation_importance(rfc_model_fit, X_test, y_test)
sorted_idx = perm_importance.importances_mean.argsort()
pyplt.barh(X_train.columns[sorted_idx], perm_importance.importances_mean[sorted_idx])
pyplt.xlabel("Permutation Importance")
```

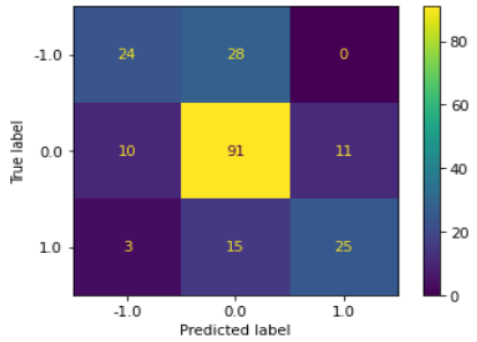
```
[0.02083975 0.01480973 0.02309937 0.04933686 0.0602089  0.01408498
 0.00194763 0.01899505 0.01073321 0.04310986 0.05152692 0.01693858
 0.01705616 0.05437071 0.04907886 0.05006043 0.01789835 0.01568661
 0.01919389 0.06470059 0.10073691 0.11369958 0.02237907 0.0139381
 0.01108158 0.11872016 0.00576817]
```

The below feature importance chart illustrates that performance_online, online_doubt_clarification, and online_interaction are the top three most important features, compared to other features in the dataset.



10.2. GBC Model Evaluation

```
## Step-10 :Evaluate performance ##
# build and plot a confusion matrix
cm = confusion_matrix(y_test, gb_predict, labels=gb_model.classes_)
cm_disp = ConfusionMatrixDisplay(confusion_matrix=cm,
                                 display_labels=gb_model.classes_)
cm_disp.plot()
pyplt.show()
```



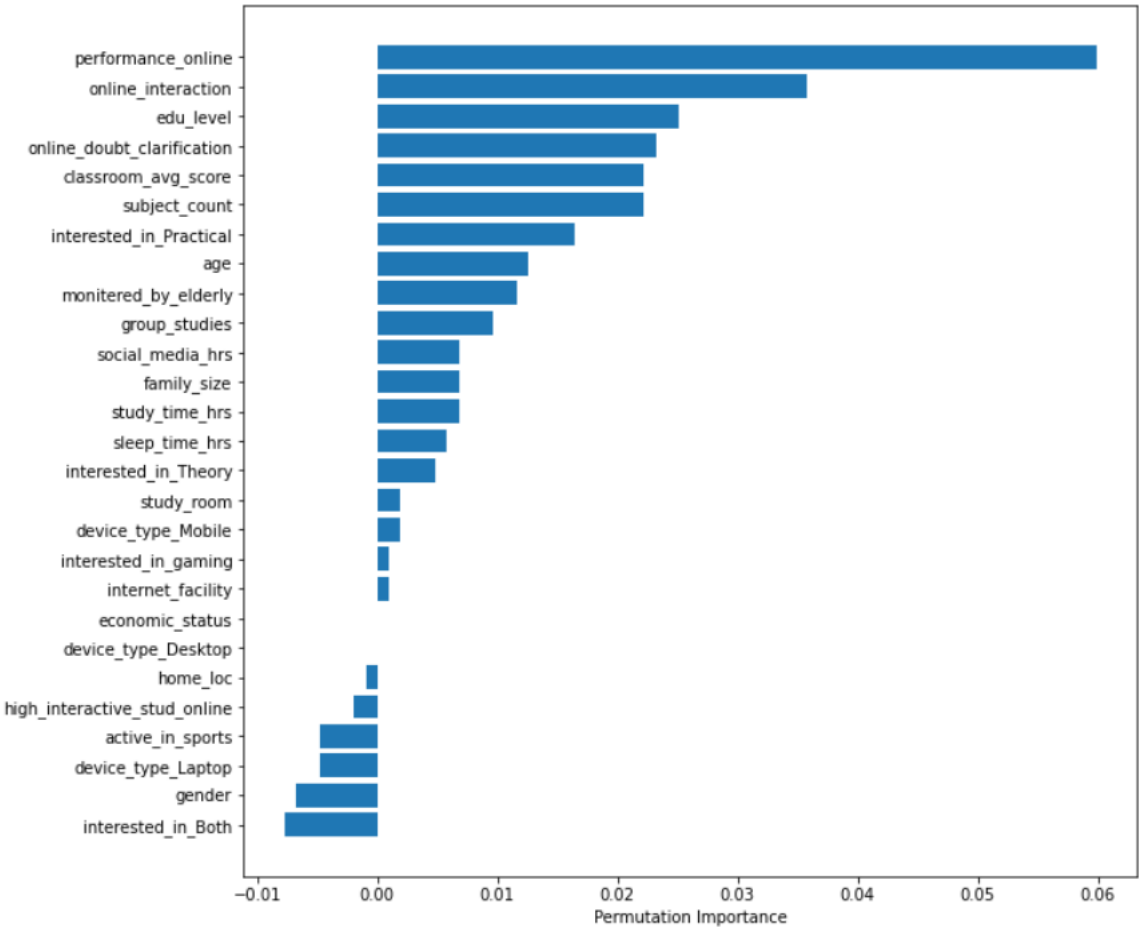

```
# calculate metrics
acc = accuracy_score(y_test, gb_predict)
f1_mac = f1_score(y_test, gb_predict, average='macro')

# print results
print(f'Accuracy = {round(acc,2)}%')
print(f'F1 score = {round(f1_mac,2)}')
```

```
Accuracy = 0.68%
F1 score = 0.64
```

```
# referred from https://mljar.com/blog/feature-importance-in-random-forest/
from sklearn.inspection import permutation_importance
print(gb_model_fit.feature_importances_)
pyplt.figure(figsize=(10,10))
perm_importance = permutation_importance(gb_model_fit, X_test, y_test)
sorted_idx = perm_importance.importances_mean.argsort()
pyplt.barh(X_train.columns[sorted_idx], perm_importance.importances_mean[sorted_idx])
pyplt.xlabel("Permutation Importance")
```

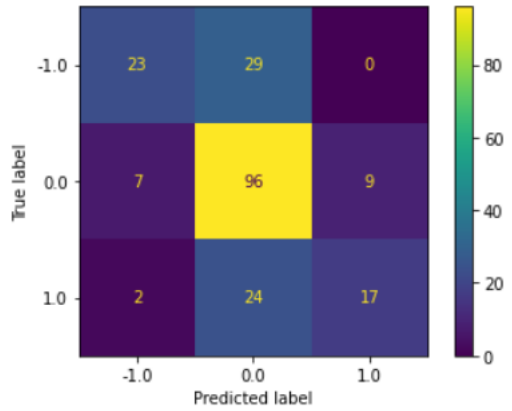
```
[0.01987143 0.01516846 0.02488624 0.05171038 0.06544868 0.0161524
 0.00532239 0.01652177 0.0156073  0.03972455 0.04639274 0.01435516
 0.01218236 0.05923122 0.05316638 0.04840739 0.01524295 0.01581853
 0.01427663 0.05989659 0.12076228 0.0864048  0.01756278 0.01446755
 0.01143166 0.13347    0.00651738]
Text(0.5, 0, 'Permutation Importance')
```

The below feature importance chart illustrates that performance_online, online_interaction and edu_level are the top three most important features, compared to other features in the dataset.



10.3. SVC Model Evaluation

```
## Step-10 :Evaluate performance ##
# build and plot a confusion matrix
cm = confusion_matrix(y_test, svc_predict, labels=svc_model.classes_)
cm_disp = ConfusionMatrixDisplay(confusion_matrix=cm,
                                 display_labels=svc_model.classes_)
cm_disp.plot()
pyplt.show()
```



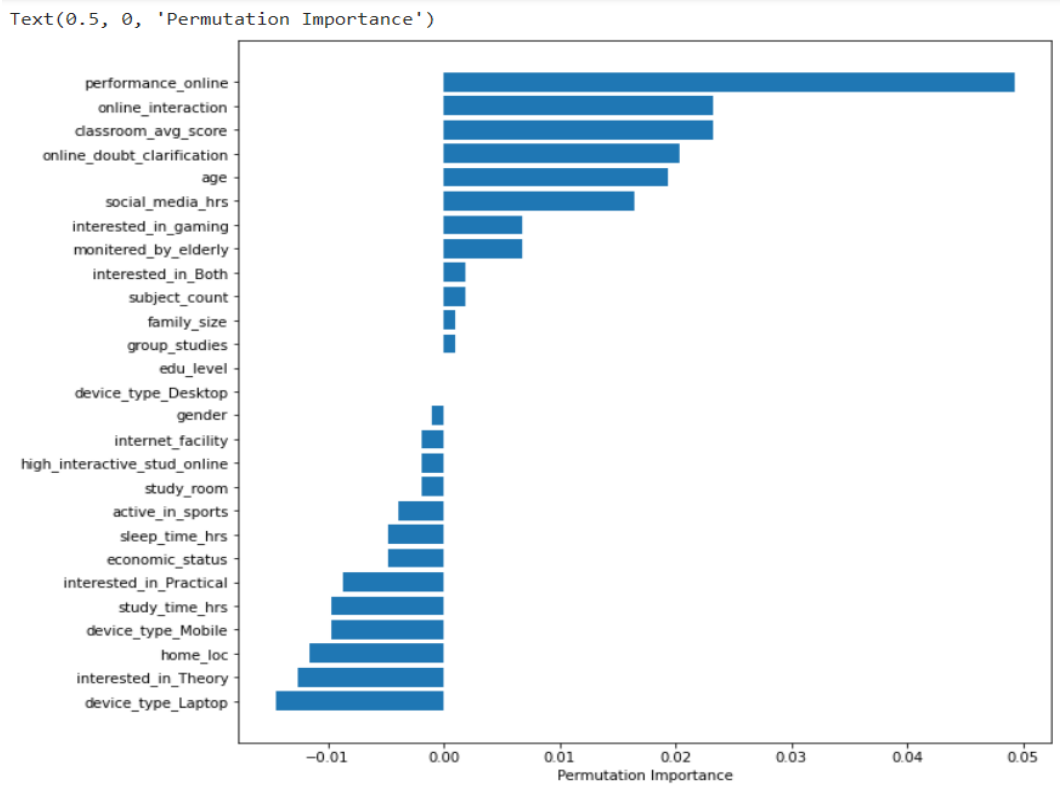
```python
# calculate metrics
acc = accuracy_score(y_test, svc_predict)
f1_mac = f1_score(y_test, svc_predict, average='macro')

# print results
print(f'Accuracy = {round(acc,2)}%')
print(f'F1 score = {round(f1_mac,2)}')
```

```
Accuracy = 0.66%
F1 score = 0.59
```

```python
# referred from https://mljar.com/blog/feature-importance-in-random-forest/
from sklearn.inspection import permutation_importance
perm_importance = permutation_importance(svc_model_fit, X_test, y_test)
feature_names = X_test.columns
features = np.array(feature_names)
sorted_idx = perm_importance.importances_mean.argsort()
pyplt.figure(figsize=(10,10))
pyplt.barh(features[sorted_idx], perm_importance.importances_mean[sorted_idx])
pyplt.xlabel("Permutation Importance")
```

The below feature importance chart illustrates that performance_online, online_interaction and classroom_avg_score are the top three most important features, compared to other features in the dataset.

```
Text(0.5, 0, 'Permutation Importance')
```



11. Best Model Selection

As shown in the below table, both Gradient Boosting Classifier model achieved the highest level of accuracy (F1 score) while Random Forest Classifier and Support Vector Machine Classifier models demonstrated lower accuracy in data prediction. Hence, Gradient Boosting Classifier model is selected as the best suited ML model for this classification problem.

ML Model	F1 Score
RFC	0.55
GBC	**0.64**
SVC	0.59

12. References

- Aliyev, V. (2020) *Gradient boosting classification explained through python*, *Medium*. Towards Data Science. Available at: https://towardsdatascience.com/gradient-boosting-classification-explained-through-python-60cc980eeb3d (Accessed: March 25, 2023).
- Gandhi, R. (2018) *Support Vector Machine - introduction to machine learning algorithms*, *Medium*. Towards Data Science. Available at: https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47 (Accessed: March 25, 2023).
- Navas, J. (2022) *What is hyperparameter tuning?*, *Anyscale*. Available at: https://www.anyscale.com/blog/what-is-hyperparameter-tuning (Accessed: March 25, 2023).
- Shafi, A. (2018) *Random Forest classification with Scikit-Learn*, *DataCamp*. DataCamp. Available at: https://www.datacamp.com/tutorial/random-forests-classifier-python (Accessed: March 25, 2023).