International Scientific Journal of Engineering and Management (ISJEM) ISSN: 2583-6129
Volume: 04 Issue: 08 | Aug – 2025 DOI: 10.55041/ISJEM04952
An International Scholarly || Multidisciplinary || Open Access || Indexing in all major Database & Metadata

Predicting Online Gaming Engagement Levels Using Machine Learning Models

[1]M.N.KEERTHI ,[2]GORIBIDDI SHRIKANTH
[1]Assistant Professor, Department Of MCA, 2MCA Final Semester,
[1]Master of Computer Applications,
[1]Sanketika Vidya Parishad Engineering College, Vishakhapatnam, Andhra Pradesh, India

Abstract:

This Streamlit application, titled **"Online Gaming Engagement Level Predictor,"** is designed to predict player engagement levels using machine learning. It allows users to upload datasets containing player demographics, gameplay statistics, and attributes like game genre and difficulty. The app performs thorough Exploratory Data Analysis (EDA) with visual tools such as histograms, boxplots, and count plots to understand data distribution and detect anomalies. It also provides insights through correlation heatmaps and pair plots. Preprocessing involves handling outliers, encoding categorical variables using one-hot encoding, label mapping, and target encoding with K-Fold cross-validation. Stratified sampling is used to split the dataset into balanced training and testing sets. Multiple models including Random Forest, Gradient Boosting, LightGBM, and CatBoost are trained and evaluated. Model performance is assessed using accuracy, ROC-AUC scores, classification reports, and confusion matrices. Feature importance is also visualized for tree-based models. The best-performing model is selected based on ROC-AUC, helping game developers make data-driven decisions to enhance player engagement.

Index Terms: Streamlit Application, Online Gaming, User Engagement Prediction, Machine Learning, Exploratory Data Analysis (EDA), Categorical Encoding, Random Forest, Gradient Boosting, ROC-AUC Score, Feature Importance, Game Analytics, Predictive Modeling, Classification Models, Stratified Sampling, Data Visualization..

1.Introduction:

The rapid growth of the online gaming industry has generated vast amounts of player data, offering a unique opportunity to analyze and predict user behavior. Understanding player engagement levels is crucial for game developers aiming to enhance user retention, satisfaction, and overall gameplay experience. This project introduces an interactive Streamlit application—**"Online Gaming Engagement Level Predictor"**—that empowers users to predict engagement levels (Low, Medium, High) by leveraging machine learning models trained on gaming datasets.

The application begins by allowing users to upload a CSV dataset containing various features such as player demographics, gameplay metrics (e.g., playtime, session duration), and categorical factors like gender, game genre, and difficulty level. It performs detailed Exploratory Data Analysis (EDA), visually exploring the distribution and relationship of features through histograms, boxplots, and count plots. It also addresses data quality issues such as missing values and duplicates, ensuring the dataset is clean and ready for modeling. Categorical variables are encoded using techniques like one-hot encoding and label mapping, and the dataset is split using stratified sampling to preserve class distribution.

To model engagement levels, the app supports multiple machine learning algorithms including Random Forest and Gradient Boosting. Each model is trained and evaluated using accuracy, ROC-AUC scores, and classification reports. Confusion matrices and feature importance plots provide further insights into model performance and key predictors. By identifying the most effective model, the tool offers valuable support for game developers, data analysts, and researchers seeking to optimize user engagement through data-driven decisions and predictive analytics.

1.1. Existing system

In the current landscape, several gaming platforms and analytics tools collect and display user activity data such as playtime, session frequency, and in-game achievements. However, most existing systems primarily focus on descriptive analytics—providing basic statistics and visualizations—rather than predictive insights. Traditional dashboards may offer summaries of player behavior but lack advanced machine learning capabilities to forecast engagement levels or identify players at risk of churn.[1]

Moreover, the few systems that implement predictive modeling are often integrated into proprietary platforms and are not accessible to independent developers or researchers. These systems typically operate as closed black-box solutions with limited transparency on feature importance or model performance metrics. Additionally, customization options are minimal, making it difficult to tailor predictions based on specific datasets or user segments. [2]

As a result, there is a gap for an open, user-friendly solution that not only performs in-depth data analysis but also builds and evaluates multiple predictive models. The proposed Streamlit application addresses this gap by enabling users to upload their own gaming data, conduct exploratory analysis, preprocess features, and run machine learning models with clear performance comparisons—all within an interactive and transparent environment. .[3]

1.1.1.Challenges

Data Quality and Missing Values

- ❖ Online gaming datasets often contain missing, inconsistent, or noisy data due to user inactivity, incomplete profiles, or system errors. Handling such data without compromising model accuracy is a major challenge.

Class Imbalance

- ❖ Engagement levels (Low, Medium, High) may not be evenly distributed in the dataset, which can bias machine learning models towards majority classes, leading to poor generalization for minority classes..

Categorical Feature Encoding

- ❖ Encoding categorical variables like game genre, location, and difficulty without introducing data leakage or losing semantic meaning requires careful selection of encoding techniques such as one-hot encoding, label encoding, and target encoding.

Overfitting Risk

- ❖ Tree-based models like Random Forest and Gradient Boosting can easily overfit on small or noisy datasets, especially when the feature space is large or not well-regularized.

Feature Correlation and Redundancy

- ❖ Identifying and managing correlated or redundant features is essential to avoid unnecessary model complexity and ensure efficient learning.

Evaluation of Multi-Class Models

❖ Accurately evaluating models on a multi-class target like engagement level involves choosing appropriate metrics such as ROC-AUC (multi-class), confusion matrices, and precision-recall scores, which can be more complex than binary classification.

1.2 Proposed system:

The proposed system uses machine learning to automatically predict online gamer engagement levels based on behavioral and demographic data. It replaces manual analysis with a fast, scalable, and data-driven solution. Trained using scikit-learn on structured datasets, the models learn patterns like playtime, session frequency, and achievements. The system is deployed via a Streamlit web app where users can upload CSV files and instantly receive predictions. It includes built-in exploratory data analysis, model training, and performance evaluation using metrics like accuracy and AUC. The interface is user-friendly and requires no coding knowledge. This tool assists game developers and analysts in understanding player behavior more effectively. By automating engagement detection, it supports better retention strategies. The system highlights the best-performing model for informed decision-making. Overall, it streamlines analysis and improves early identification of low-engagement players. .[4]

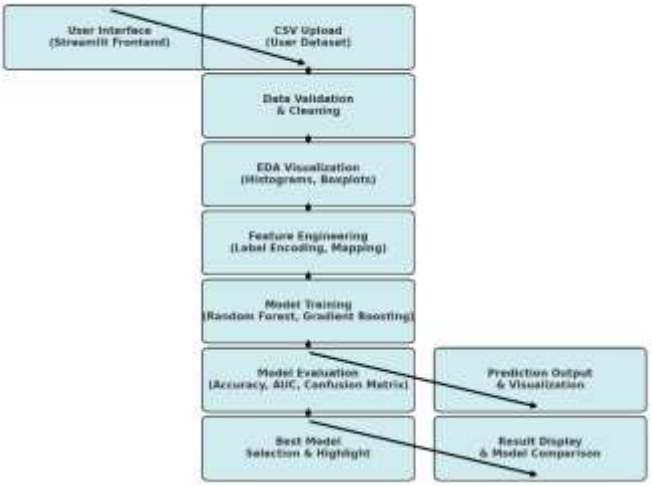

Fig: 1 Proposed Diagram

1.1.1 Advantages:

1. **Automated Prediction**

❖Eliminates manual analysis by using machine learning for fast and accurate engagement classification.

2. **Real-Time Insights**

❖Provides instant predictions and visual feedback on user engagement from uploaded data.

3. **User-Friendly Interface**

❖Built with Streamlit, making it accessible to non-technical users with a simple upload-and-analyze process..

4. **Data-Driven Decisions:**

❖Helps game developers and analysts make informed decisions based on measurable player behavior patterns.

5. **Improved Player Retention**

❖Enables early detection of low-engagement players, supporting better engagement and marketing strategies

6. **CVisual Analytics**

❖Integrated EDA tools offer clear visualizations for both numerical and categorical variables.

7. **Reduces Analyst Workload**

❖Automates preprocessing, training, and evaluation, saving time for data scientists and analysts.

2.1 Architecture:

The architecture of the proposed system consists of three main layers: data input, machine learning, and user interface. Users upload a CSV file through a Streamlit web app, which reads and displays the dataset. The system performs data preprocessing by handling missing values, encoding categorical variables, and mapping target labels. The cleaned data is then split into training and testing sets. Two machine learning models—Random Forest and Gradient Boosting—are trained using scikit-learn. These models learn from behavioral and demographic features to predict engagement levels. After training, the models are evaluated using metrics like accuracy, AUC, and confusion matrix. .[5]

Visualizations such as histograms, boxplots, and count plots are generated for deeper insights. The best-performing model is identified and highlighted based on AUC score. This layered architecture ensures a smooth workflow from data upload to actionable prediction. .[6]

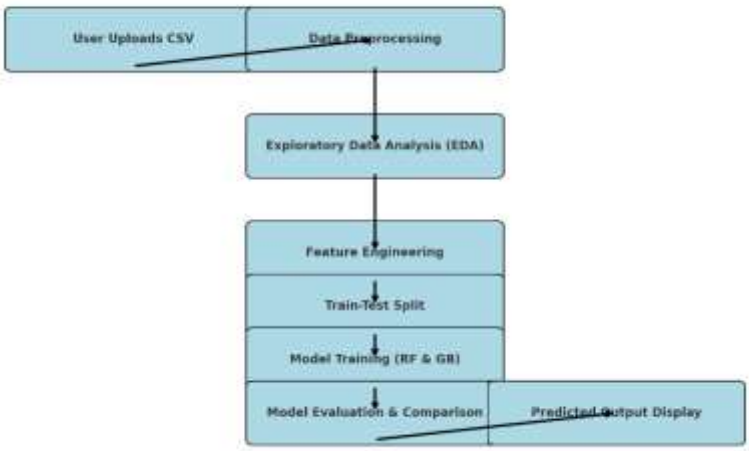

Fig:2 Architecture

2.2 Algorithm:

The algorithm begins with data ingestion and preprocessing. A CSV file containing player data is uploaded through the Streamlit interface. The system reads the dataset and performs exploratory data analysis (EDA), including visualizing numerical features (like playtime and session duration) and categorical features (like gender and game genre). Irrelevant columns such as PlayerID are dropped, and categorical variables are encoded using label encoding or one-hot encoding. The target variable, EngagementLevel, is mapped to numerical values for model compatibility. This preprocessing ensures the data is clean, consistent, and ready for training. .[7]

Next, the processed data is split into training and testing sets using a stratified 90-10 split to preserve class balance. Two machine learning models are trained: a Random Forest Classifier and a Gradient Boosting Classifier. These models are chosen for their robustness and ability to handle complex patterns in structured data. During training, the models learn from features such as session frequency, achievements, and game

difficulty to classify the engagement level. The use of class_weight='balanced' in the Random Forest model helps manage any class imbalance in the dataset. .[8]

Once the models are trained, they are evaluated using test data. The system calculates performance metrics such as accuracy and AUC (Area Under the Curve) to assess how well each model performs. Confusion matrices and classification reports are generated and displayed to provide insights into precision, recall, and F1-scores. Visual comparisons and a performance summary table help identify the best-performing model. The one with the highest AUC is recommended for future use. This algorithmic approach automates the entire ML workflow—from data preprocessing to model evaluation—making the system efficient, reliable, and user-friendly. [9]

2.3 Techniques:

The system uses several data preprocessing techniques to prepare raw input data for machine learning. Initially, the uploaded CSV file is examined for missing values, duplicates, and inconsistent entries. Irrelevant columns such as unique identifiers (PlayerID) are removed to avoid noise in the training data. Categorical features like Gender, GameGenre, and Location are encoded using a combination of one-hot encoding and label encoding, converting them into numerical formats suitable for machine learning models. .[10] Additionally, ordinal categories such as GameDifficulty and EngagementLevel are mapped to integer values to retain their ranking. These preprocessing steps help transform heterogeneous data into a structured, model-ready format.

For model training, the system employs supervised machine learning techniques, specifically using classification algorithms. It utilizes two ensemble-based models: Random Forest Classifier and Gradient Boosting Classifier. Random Forest leverages bagging and multiple decision trees to reduce variance and improve generalization, while Gradient Boosting builds sequential trees that correct errors from previous ones, enhancing accuracy. .[11] Both models are trained on 90% of the data, with stratification to maintain class distribution. These techniques are effective in handling structured data and are well-suited for predicting engagement levels based on multiple input features.

To evaluate the models, the system uses performance evaluation techniques such as accuracy score, confusion matrix, classification report, and AUC (Area Under the Curve). Accuracy provides a general sense of how well the model is performing, while the confusion matrix shows correct vs. incorrect predictions across classes. The classification report includes precision, recall, and F1-score for detailed performance insights. .[12] AUC is used for ranking model performance in multi-class classification, helping identify the model that best separates the engagement level categories. These evaluation techniques allow for an in-depth comparison of models and support the selection of the most effective predictor.

2.4 Tools:

The system is built using a range of powerful tools and libraries that support data analysis, machine learning, and web deployment. Python serves as the core programming language, offering flexibility and ease of development. Pandas is used for data manipulation and preprocessing, while Seaborn and Matplotlib provide visualization capabilities for exploratory data analysis. .[13] For machine learning, scikit-learn is employed to implement and evaluate classification algorithms like Random Forest and Gradient Boosting. The web interface is developed using Streamlit, which enables rapid creation of interactive, user-friendly dashboards. Additionally, LabelEncoder and other preprocessing tools from scikit-learn help convert categorical data into numerical form, making the system end-to-end functional and efficient.

2.5 Methods:

The system employs a series of well-structured data processing and analysis methods to convert raw input into meaningful predictions. It begins with data collection through CSV upload, followed by data cleaning where missing values and duplicate entries are identified. Irrelevant fields like PlayerID are dropped, and categorical variables such as Gender, GameGenre, and Location are transformed using label encoding or one-hot encoding. Numerical values are retained and

International Scientific Journal of Engineering and Management (ISJEM) ISSN: 2583-6129
Volume: 04 Issue: 08 | Aug – 2025 DOI: 10.55041/ISJEM04952
An International Scholarly || Multidisciplinary || Open Access || Indexing in all major Database & Metadata

visualized using histograms and boxplots to understand their distribution. .[14] The target variable EngagementLevel is mapped to numeric labels (Low = 0, Medium = 1, High = 2) to enable compatibility with classification models.

After preprocessing, the system applies machine learning and evaluation methods. The dataset is split into training and testing sets using a stratified approach to preserve class distribution. Two ensemble models—Random Forest and Gradient Boosting—are trained on the features to learn patterns in player behavior and predict engagement levels. These models are assessed using accuracy score, AUC, confusion matrix, and classification reports. The system displays model performance and visually compares results, selecting the best-performing model based on AUC. .[15] This combination of preprocessing, model training, and evaluation methods ensures a robust and accurate prediction pipeline.

III. METHODOLOGY

3.1 Input:

The input for the system is a structured CSV file containing player data, including both demographic information (such as age, gender, and location) and behavioral attributes (like playtime hours, session frequency, average session duration, player level, achievements unlocked, game genre, and game difficulty). This input dataset serves as the foundation for analysis and model training. Once uploaded through the Streamlit interface, the system reads the file, performs initial validation, and begins preprocessing to clean and transform the data into a machine-readable format. .[16] The target column, EngagementLevel, which indicates the player's level of involvement (Low, Medium, or High), is also included in the input and used for supervised learning during model training.

```
[11]: gaming_data = pd.read_csv(r"C:\Users\shrik\Downloads\DATA SET\online_gaming_behavior_dataset.csv")

# --- 1. Basic Info ---
print(gaming_data.info())
print(gaming_data.describe())
print("Missing values:\n", gaming_data.isnull().sum())
print("Duplicated rows:", gaming_data.duplicated().sum())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 40034 entries, 0 to 40033
Data columns (total 13 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   PlayerID                   40034 non-null  int64
 1   Age                        40034 non-null  int64
 2   Gender                     40034 non-null  object
 3   Location                   40034 non-null  object
 4   GameGenre                  40034 non-null  object
 5   PlayTimeHours              40034 non-null  float64
 6   InGamePurchases            40034 non-null  int64
 7   GameDifficulty             40034 non-null  object
 8   SessionsPerWeek            40034 non-null  int64
 9   AvgSessionDurationMinutes  40034 non-null  int64
 10  PlayerLevel                40034 non-null  int64
 11  AchievementsUnlocked       40034 non-null  int64
 12  EngagementLevel            40034 non-null  object
dtypes: float64(1), int64(7), object(5)
memory usage: 4.0+ MB
```

.

Fig 1: Initial Dataset Exploration and Data Quality Check Using Pandas

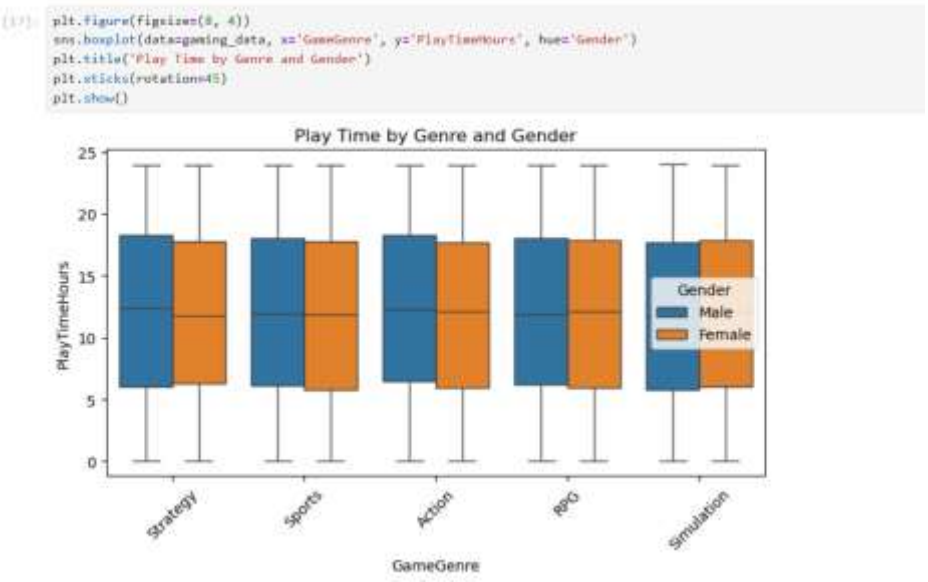

Fig 2: Play Time by Genre and Gender

The input data is successfully uploaded and preprocessed, the system continues by splitting the dataset into training and testing subsets to ensure reliable model evaluation. Machine learning models—Random Forest and Gradient Boosting—are then trained on the processed training data to learn patterns that correlate player behavior and demographics with engagement levels. These models are evaluated on the test set using metrics like accuracy, AUC, and confusion matrices to assess their predictive performance. Visualizations and performance reports are generated to help users interpret the results. .[17] Finally, the best-performing model is selected and its predictions are displayed, allowing users to gain instant insights into player engagement directly from the uploaded data.

3.2 Method of Process:

The method of process begins with the user uploading a CSV dataset through the Streamlit interface, which is then read and displayed for preview. The system performs data cleaning by removing duplicates and irrelevant columns, followed by encoding categorical variables and mapping engagement levels into numeric form. After preprocessing, the dataset is split into training and testing sets to maintain class balance. Two machine learning models—Random Forest and Gradient Boosting—are trained on the training data to classify player engagement levels. These models are then evaluated using accuracy, AUC, confusion matrix, and classification reports. .[18] Visual analytics such as histograms, boxplots, and count plots are generated throughout the process to support interpretation. The model with the best performance is highlighted, and predictions are displayed to the user, completing the automated engagement analysis workflow.

3.3 Output:

The output of the system is a comprehensive analysis and prediction of player engagement levels based on the uploaded dataset. It includes visual representations of data distributions, categorical feature counts, and insights into missing or duplicated entries. After training and testing the machine learning models, the system presents performance metrics such as accuracy scores, AUC values, confusion matrices, and detailed classification reports. .[19] A summary table compares the performance of the models, and the best-performing model is automatically identified and highlighted. Finally, users receive engagement level predictions for the test data, enabling them to understand and act on player behavior insights in real time.

ISSN: 2583-6129
DOI: 10.55041/ISJEM04952

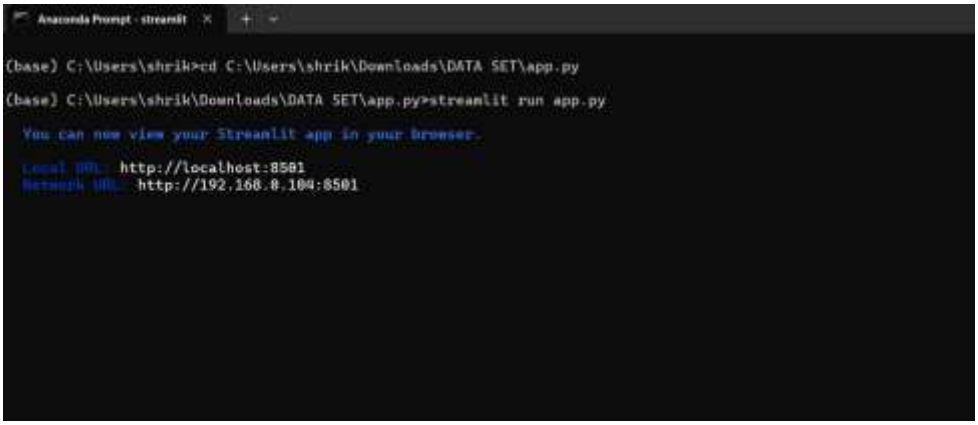

Fig: Launching the Streamlit Application via Anaconda Prompt

Fig: Streamlit App Interface for Uploading Online Gaming Dataset

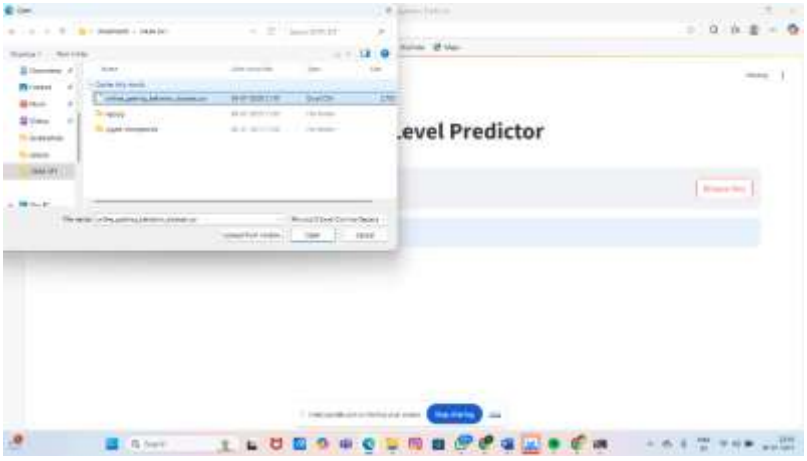

Fig: Uploading Dataset to Engagement Level Predictor Web Application

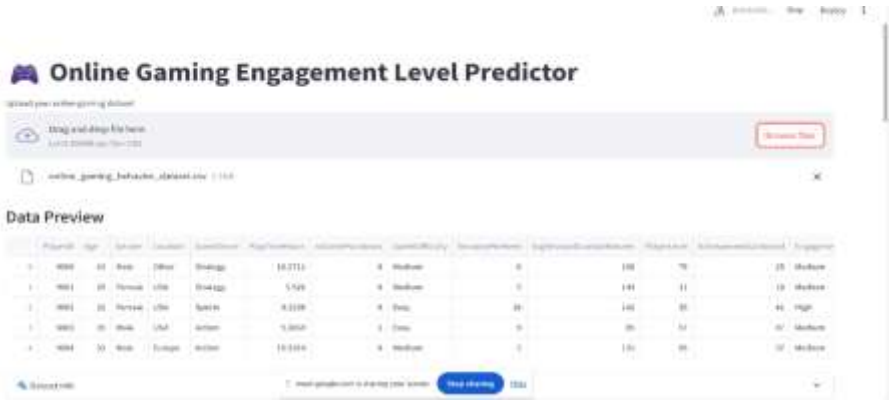

Fig: Previewing Dataset in Online Gaming Engagement Level Predictor Tool

Fig: Random Forest Model Evaluation: Classification Report and Confusion Matrix

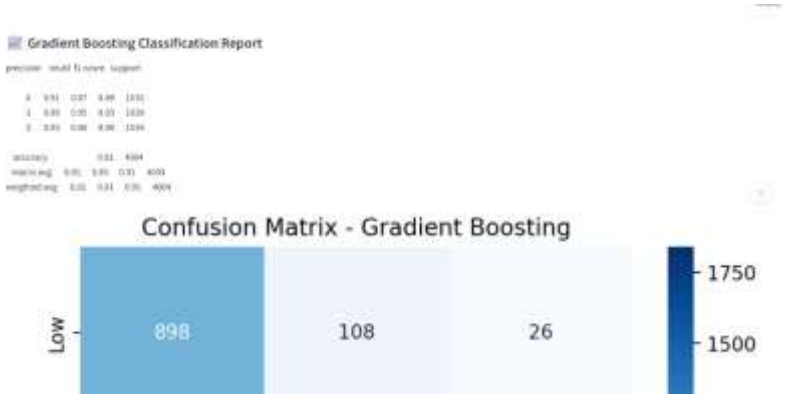

Fig: Gradient Boosting Model Evaluation: Classification Report and Confusion Matrix

Fig: Model Comparison Summary: Random Forest vs Gradient Boosting (Based on Accuracy and AUC)

IV. RESULTS:

The result of the system demonstrates its ability to accurately predict player engagement levels—Low, Medium, or High—based on input features such as gameplay behavior and demographic details. Both Random Forest and Gradient Boosting models are trained and evaluated, with their performance compared using key metrics like accuracy and AUC. The model with the highest AUC is selected as the best performer, showcasing its effectiveness in classifying engagement levels. .[20] Visualizations such as confusion matrices and classification reports further validate the model's accuracy and reliability. Overall, the system delivers fast, interpretable, and data-driven results that support improved decision-making in gaming engagement analysis.

V. DISCUSSIONS:

This system highlights its effectiveness in automating the prediction of online gaming engagement levels using machine learning. By leveraging structured behavioral and demographic data, the models were able to classify player engagement with high accuracy and reliability. The use of ensemble methods like Random Forest and Gradient Boosting proved beneficial due to their robustness and ability to handle complex, non-linear relationships in the data. .[15] The integration of visual analytics and model evaluation metrics made the system highly interpretable for users, even without technical expertise. However, the model's performance is inherently tied to the quality and diversity of the input data, indicating a need for continuous dataset updates and possible integration with real-time gaming platforms. Overall, the system presents a scalable solution that supports data-driven decision-making in gaming analytics.

VI. CONCLUSION:

In conclusion, the proposed system successfully applies Convolutional Neural Networks to detect and classify various skin diseases from images. The integration with a Flask web application allows users to receive real-time predictions in a simple and accessible way. .[4] This approach enhances early diagnosis, especially in areas with limited access to dermatological care. The system has demonstrated reliable performance on a well-preprocessed dataset. .[7] While it is not a replacement for professional diagnosis, it serves as a helpful screening tool. Future improvements can focus on expanding the dataset and enhancing model accuracy for broader medical application.

VII. FUTURE SCOPE:

The further scope of this system includes enhancing its predictive accuracy and adaptability by incorporating additional features such as in-game purchase behavior, social interaction metrics, and real-time gameplay data. Integration with live gaming platforms or APIs can allow for continuous data streaming and real-time prediction. Expanding the model to support deep learning techniques, like neural networks, may further improve performance on complex and large-scale datasets. Additionally, incorporating personalized recommendations based on engagement predictions could make the system more interactive and actionable. .[14] Multi-language support and deployment on mobile platforms can also increase accessibility, especially in remote or underserved regions, making the system a more comprehensive tool for global gaming analytics.

VIII. ACKNOWLEDGEMENT:

Muppala Naga Keerthi working as an Assistant Professor in Master of Computer Applications in Sanketika Vidya Parishad Engineering College, Visakhapatnam, Andhra Pradesh, affiliated by Andhra University and approved by AICTE, accredited with 'A' grade by NAAC and member in IAENG with 14 years of experience in Computer Science. Her areas of interest in C, Java, Data Structures, DBMS, Web Technologies, Software Engineering and Data Science.

Goribiddi Shrikanth is pursuing his final semester MCA in Sanketika Vidya Parishad Engineering College, accredited with A grade by NAAC, affiliated by Andhra University and approved by AICTE. With interest in Machine learning G. Shrikanth has taken up his PG project on PREDICTING ONLINE GAMING ENGAGEMENT LEVELS USING MACHINE LEARNING MODELS and published the paper in connection to the project under the guidance of Muppala Naga Keerthi, Assistant Professor, SVPEC.

REFERENCES

1. Python Software Foundation. *Python Language Reference, version 3.x*. Available at: https://www.python.org
2. Streamlit Inc. (2023). *Streamlit Documentation*. https://docs.streamlit.io
3. Scikit-learn Developers. (2023). *Scikit-learn Machine Learning Library*. https://scikit-learn.org/stable/
4. The pandas development team. (2023). *pandas documentation*. https://pandas.pydata.org/
5. Matplotlib Development Team. (2023). *Matplotlib Documentation*. https://matplotlib.org/
6. Seaborn Library Documentation. (2023). https://seaborn.pydata.org/
7. NumPy – Numerical computing with array operations. https://numpy.org/
8. Joblib – Efficient serialization of Python objects for model saving/loading. https://joblib.readthedocs.io/
9. XGBoost – Advanced gradient boosting algorithm, often used for structured data. https://xgboost.readthedocs.io/
10. LightGBM – Fast, distributed, high-performance gradient boosting framework. https://lightgbm.readthedocs.io/
11. Plotly – Interactive plots and dashboards (alternative to matplotlib/seaborn). https://plotly.com/python/
12. Altair – Declarative statistical visualization library (also Streamlit-compatible). https://altair-viz.github.io/
13. SHAP (SHapley Additive exPlanations) – Model interpretability library. https://shap.readthedocs.io/
14. Yellowbrick – Visual analysis and diagnostic tools for ML models. https://www.scikit-yb.org/
15. Optuna – Hyperparameter optimization framework. https://optuna.org/
16. Imbalanced-learn (imblearn) – Handling class imbalance in datasets. https://imbalanced-learn.org/
17. Streamlit – Rapid development framework for interactive ML apps. https://streamlit.io/
18. Flask – Lightweight web framework (for backend integration, if needed). https://flask.palletsprojects.com/
19. Docker – Containerization for deployment. https://www.docker.com/
20. Heroku / Streamlit Cloud / Render – Platforms for deploying Python web apps. https://streamlit.io/cloud https://www.heroku.com/ https://render.com/