CSE343/ECE343 Machine Learning Mid sem Report
Prediction of purchase decisions in F2P (Free To Play) Games.

Harsh Hingorani 2022197 | Harsh Nangia 2022199 | Idhant Arora 2022220 | Madhav Kansil 2022270

1. Abstract

In this report, we will demonstrate what are the purchasing decision patterns of an individual playing a Free to play game and what factors impact and drive a player to spend money for in-game purchases. We also investigate the importance of handling class imbalances and how different techniques result in different statistics. In our analysis we also employed Feature Engineering and Encoding for improving model accuracy.We also used GridSearchCV to find best parameters for our best performing model. **Github**

2. Introduction

Gaming has come a long way, from 2D-style games in the 1970s to 3D and virtual reality gaming in the new century's second decade. All of this is because of the advancements in game design technologies like ray tracing, easy-to-use game engines like UnReal Game Engine and Godot. But the most impact has been the introduction of multiplayer gaming like BGMI, Valorant etc.

The motivation for this project is that as more and more people indulge in gaming and increase in commercialisation of games primary source of revenue for these games comes from optional in game purchases done by the players. This makes companies wanting to know what all patterns and user behaviour influence these patterns. These questions can be answered using machine learning models to predict whether a player would make a purchase or not.

3. Literature Survey

3.1. In-game transactions in Free-to-play games: Player motivation to purchase in-game content (Link)

The research undertaken by Faculty of Department of Game Design of Uppsala universitet was aimed at classifying players based on their likelihood of making purchases and to predict the number of purchases they would make. The findings suggested that understanding player activity could significantly improve lifetime value (LTV) predictions and strengthen revenue streams for developers.

3.2. Predicting player disengagement and first purchase with event-frequency (Link)

The authors of this paper have tried to address the issues of predicting player disengagement, and moment of first purchase based on event-frequency data. The generalization was achieved by using frequency of the events inside the game prior to knowing anything about them. For the first purchase prediction the players' behavior was observed over the past two weeks. The results of the research have shown that frequency based data representation is significantly better than random guess.

3.3. Predicting Purchase Decisions in Mobile Free to Play Games (Link)

Mobile digital games are predominantly released under the freemium business model, yet only a small percentage of players make purchases. The ability to predict which players will spend money allows companies to optimize their marketing strategies and tailor customer relationship management to individual user profiles. This research addresses this challenge through two predictive models trained on a dataset of 100,000 players:

- A classification model aimed at predicting whether a purchase will occur or not.
- A regression model aimed at predicting the number of purchases a user will make.

4. Dataset

4.1. Data at a Glance

Feature	What it represents	Type
PlayerID	ID of the player playing	`int`
Gender	Gender of the player	`str`
Age	Age of the player	`int`
Location	Location of the player playing	`str`
GameGenre	Type of game	`str`
PlayTimeHours	Avg hours spent per session	`float`
GameDifficulty	Difficulty level of the game	`str`
SessionsPerWeek	Gaming sessions per week	`int`
AvgSessionDurationMinutes	Avg duration of each session in mins	`int`
PlayerLevel	Current Level of player	`int`
AchievementsUnlocked	Number of achievements unlocked	`int`
EngagementLevel	Engagement level reflecting player retention	`str`
InGamePurchases	Whether a purchase was made or not	`bool`

4.2. Description

The dataset was taken from kaggle. The given dataset has 13 features described in subsection 4.1. We

take the feature `InGamePurchase` as our target variable for analysis. During EDA our observations were as follows

4.3. Observations

- The dataset is highly imbalanced for the predictive feature. This causes a problem for a model to correctly classify the new test data points.
- The dataset has a few features irrelevant to the problem statement like PlayerID. We tackle these problems in PreProcessing part.
- Data doesn't have an interaction feature which can be used as a better feature to show how one variable correlates with another. Feature Engineering is used to solve this.

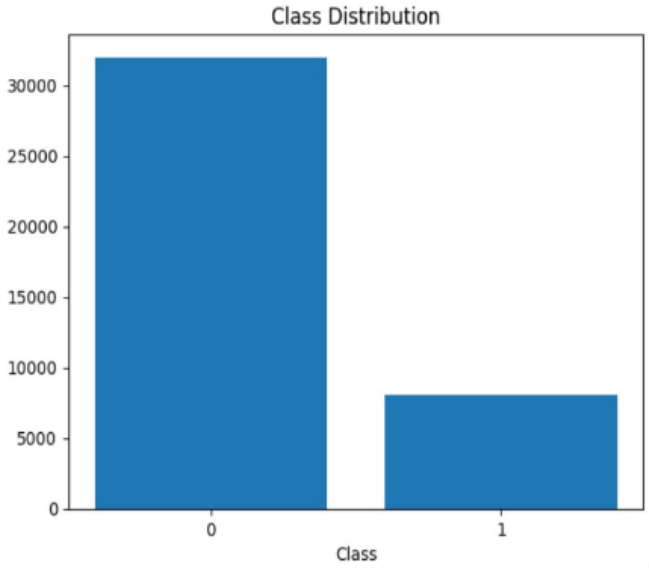

Figure 1. Imbalanced Dataset

4.4. Feature Engineering

For better explanation of the dataset we added another feature called `EstimatedAnnualIncome`. We added this, taking Country and Engagement level as reference. This assumption is reasonable considering that a person living in 1st world country having an engagement level high would be from a well off household as compared to someone who have a high engagement level in a 3rd world country. We assign random income between custom ranges. We chose against the categorical variable of High, Medium, Low due to dataset having low variation and already major variables were categorical. We added integer values to introduce a notion of variation for robust model estimation.

We take ranges from 0 to 150,000. With high income bins as (100,000 to 150,000). Medium bins as (50,000 to 100,000) and low as (0 to 50,000). Figures are in dollars taken as inspiration from a part of real world data given in [1] and [2].

5. Methodology

We classified that the user will be making an In-game purchase in a Free to Play game, using the player's behavioral attributes. Encoded the In-game Purchase in Free to Play games as 1 as Yes and 0 as No. For reducing the imbalance of data, we used SMOTE (Synthetic Minority Oversampling Technique) and Tomek Links and few more. For the predictive models we have used 4 models as mentioned in 5.2

5.1. Preprocessing

5.1.1 Correcting Imbalances

During EDA we observed that the dataset was imbalanced i.e number of samples classified as ***No (0)*** and ***Yes (1)*** were in the ratio 4 : 1, meaning out of 80% of the dataset are classified as negative and rest 20% as positive. To address this issue we employed resampling methods such as undersampling and oversampling.

After trying different method we concluded that we would employ `Tomek linking` for `undersampling` and ***SMOTE***, ***ADASYN*** as ***oversampling*** techniques and a combination of SMOTE and ENN(Edited Nearest Neighbours) called **SMOTEENN** for both under and over sampling. While splitting dataset for training and testing we also used Stratified Sampling for equal distribution. This way we can show the impact different sampling methods on our dataset.

5.1.2 Irrelevant Variables and Encoding

We employed one hot and numerical encoding to encode variables which were categorical in nature to convert them to numerical data so that our model could interpret their relevance easily. We used sklearn's `LabelEncoder` class. We also dropped irrelevant features like PlayerID as it provides no information towards the predictive nature of the model and EngagementLevel and Location as their interaction term is already included.

5.1.3 Scaling

We also used Min-Max scaling for each feature to make them consistent and in range between 0 and 1.

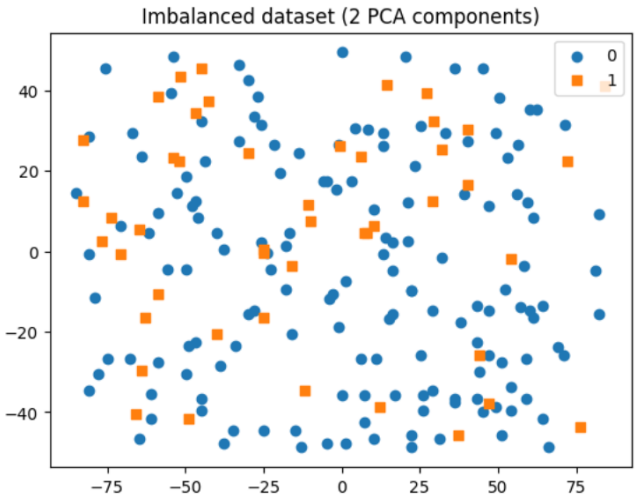

Figure 2. Imbalanced Dataset

5.1.4 Sampling

After employing SMOTE and Tomek Links, to get a robust estimate of the models we used a few probabilistic sampling methods like `Systematic Random` and `Stratified Random` sampling techniques to create a new and smaller dataset for a better representation for different classes in the dataset. We also did boostrap sampling for all the models and present the results.

- **`Random Sampling`** - To apply Random Sampling we took intersection of SMOTE and Tomek Links data points and randomly append them to make a new shorter dataset with consistent data points.
- **`Systematic Random Sampling`** - To apply Systematic Random Sampling take randomly shuffled data points and using a step size append them to make a new dataset. It is done to introduce a better notion of all classes being included.
- **`ADASYN`** - Adaptive Synthetic Sampling or ADASYN works by generating synthetic samples for minority classes based on the feature space of the original dataset. It calculates the density distribution of each minority class sample and generates synthetic samples according to the density distribution. This adaptive approach ensures that more synthetic samples are generated for minority class samples that are harder to learn, thus improving the classification performance of machine learning models [3].
- **`SMOTE-ENN`** - It uses SMOTE ability to generate synthetic examples for minority class and ENN ability to delete the observation and the majority class from the observation's K-nearest neighbor if both the classes are different [4].

5.2. Models

5.2.1 Logistic Regression

We employed a Logistic Regression from scikit learn library for predicting the InGamePurchases and checked robustness. We see an accuracy of 70% in Logistic Regression on SMOTE and 82% on Tomek Links. The best was from Tomek followed by random strategies which is 76%.

5.2.2 Decision Tree

Then we tried Decision Tree for classification taking implementation from scikit learn. We observed a accuracy same as that of logistic regression with a score of 69% on SMOTE and 80% using Tomek Links followed by random strategies which is 76%.

5.2.3 Random Forest

To tackle Decision Tree's overfitting problem we make a Random Forest. Applying this we achived the accuracy of 85% on SMOTE and 87% on Tomek Link methods followed by 83% on random strategies taking `n_estimators as 500`.

5.2.4 Naive Bayes

Naive Bayes is another classification model which uses a Naive interpretation of Bayes Theorem, which assumes that features are not correlated among themselves and are independent of each other. This model performs worst as compared to others achieving an accuracy of 69% in SMOTE and 79% using Tomek Links this saw no improvement with random strategies too.

6. Results and analysis

- Observing the results, we see a lower accuracy in logistic regression. This is because of the fact that data contains a high variation in samples and estimating a proper boundary for logistic becomes difficult due to high outliers and variation in data. Other statistics were ***F1 score*** as ***45%***, ***Recall*** as ***64%*** and ***Precision*** as ***35%*** with Tomek Links dataset and ***F1 score*** as ***30%***, ***Recall*** as ***20%*** and ***Precision*** as ***63%*** with SMOTE dataset. Tomek Links gives better metrics in terms of precision. But recall for SMOTE is better maybe due to more synthetic data since the new points generated are very near to each other. Random Strate-

gies give ***F1 score*** as ***45%***, ***Recall*** as ***49%*** and ***Precision*** as ***42%***

- The improvement of Metrics of Decision Tree as compared to Logistic Regression is because of the fact that Decision Trees split the node at highest information gain in contrast to Logistic Regression which only tries to get probabilities of the data point. But one need to be careful as an unpruned decision tree might overfit. The statistics were ***F1 score*** as ***45%***, ***Recall*** as ***50%*** and ***Precision*** as ***42%*** with Tomek Links dataset and

  F1 score as ***46%***, ***Recall*** as ***50%*** and ***Precision*** as ***42%*** with SMOTE dataset. Random Strategies give ***F1 score*** as ***46%***, ***Recall*** as ***53%*** and ***Precision*** as ***40%***

- For Random Forest, the stats are ***F1 score*** as ***55%***, ***Recall*** as ***40%*** and ***Precision*** as ***87%*** with Tomek Links dataset and ***F1 score*** as ***54%***, ***Recall*** as ***43%*** and ***Precision*** as ***71%*** with SMOTE dataset. Random Strategies give ***F1 score*** as ***54%***, ***Recall*** as ***43%*** and ***Precision*** as ***73%***

- Poor performance of Naive Bayes is because of the fact that we engineered a correlated feature `EstimatedAnnualIncome` and also other features are correlated as evidenced by a correlation Heatmap. The statistics obtained with it was ***F1 score*** as ***23%***, ***Recall*** as ***16%*** and ***Precision*** as ***42%*** with Tomek Links dataset and ***F1 score*** as ***40%***, ***Recall*** as ***51%*** and ***Precision*** as ***42%*** with SMOTE dataset. Random Strategies give ***F1 score*** as ***45%***, ***Recall*** as ***53%*** and ***Precision*** as ***40%***

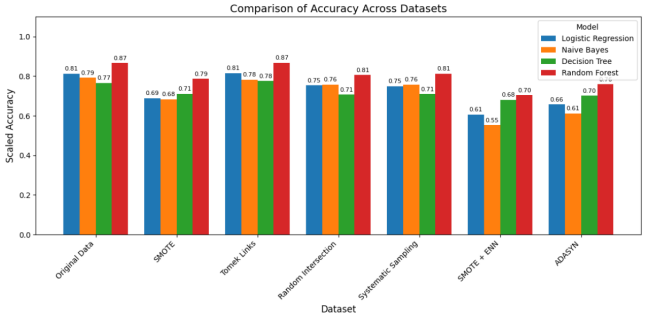

Figure 3. Accuracy Stack Graph

7. Conclusion

In our study of the given dataset, we employed various sampling methods and compared them to the original dataset. During this process we got to know how

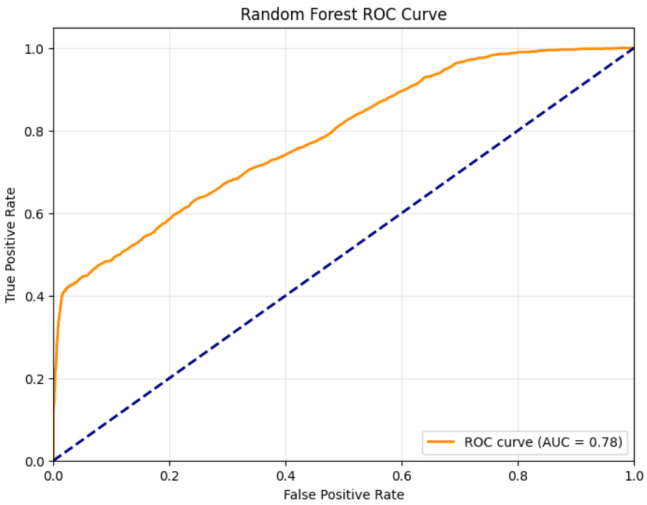

Figure 4. ROC Curve Tomek Links

to deal with data imbalance since it was present in the dataset. The techniques and Models implemented by us shows that TOMEK Links provides improved metrics for models this is because of the fact that it aims to balance class distribution by decreasing all Tomek links i.e. data points of different classes that are the closest neighbors of each other.

But we see from our correlation heatmaps that `InGamePurchases` might not be dependent upon variables like `AvgSessionDurationMinutes`. These can be due to the fact that certain features like `FirstInstallTime`, `IngameFriends` ,`AdsWatched` ,`AdsClicked` and `PlayerGameType` (single player or multiplayer) etc are not present in our current dataset.

For the models implemented, we observed random forest with TOMEK links sampling method to give best accuracy and better performance metrics than other models implemented. Then we implemented gridsearchCV to find the best hyperparameters for Random Forest, when used with TOMEK links sampled data.

8. References

[1] World Data Info for Average income around the world **(Link)**

[2] Investopedia Article on Average American net worth by Erin Gobler **(Link)**.

[3] ActiveLoop article on Adaptive Synthetic Sampling (ADASYN) **(Link)**.

[4] SMOTEENN documentation - Sklearn **(Link)**.