Summary

The health and longevity of a game is fully dependent on the game's ability to retain its players over time. Players must be engaged for them to want to keep spending money on subscriptions for the game. The aim of this project is to predict if a player will stop playing.

The available data is limited: the dataset contains only one feature per user per month. The goal is to feature engineer this one column into multiple defining attributes of a player. The aim of these features is to help answer the following questions:

- Will the player keep playing?
- What will the score of the player be next month?

The most helpful features for answering these questions are:

- What's the players programming skill level?
- Have their bases been destroyed or did their bot break?
- How long has the player been inactive?
- How long has the player been playing?

Turning these questions into features in the dataset has enabled the training of several models, from which the best model was chosen to predict if a player will leave. This model is the Decision tree.

The model will be used (hypothetically) to generate a file which contains the players which will stop playing the game in the current month, which Screeps will use to try to retain these players.

1. Business understanding

1. 1 Screeps?

Screeps can be defined as a 4X game. 4X is a subgenre of strategy-based computer and board games, and includes both turn-based and real-time strategy titles. The gameplay generally involves building an empire. Emphasis is placed upon economic and technological development, as well as a range of military and non-military routes to supremacy.

Players in Screeps get score when they exploit the energy mines in rooms under their control. The more rooms they control, the more score they get. All players are ranked on monthly leaderboards based on their score. This is the score which is the primary input for the models which will be discussed in the following chapters.

One unique attribute of the game Screeps is that the players do not have to “play” the game. The players code a “bot” to play the game for them. This means a player can have a (very high) score, even if they have not “played” during that month.

1.2 Revenue streams

Screeps has three primary revenue streams:

1. The upfront price a user pays to “buy” the game
2. One-month CPU unlocks (mandatory for certain endgame bots)
3. Permanent CPU unlocks (mandatory for certain endgame bots)

This means Screeps is incentivized to keep selling their game, and to keep players engaged. This practically means the following:

1. Players must be kept engaged by the game.
2. Engaged players will tell their friends about the game and leave good reviews.
3. Engaged players will buy CPU unlocks.
4. Good reviews will drive sales.

Therefore, the understanding of the likelihood of a player to stay engaged is of vital importance to the long-term financial success of a game like Screeps.

The business objective for this project is to provide an early warning that a player will quit playing the game. This allows Screeps to incentivize this player to stay (and keep spending money on the game).

2. Data understanding and preparation

The data understanding and preparation are considered two separate phases in the CRISP DM cycle. However, they've been combined into one phase due to the limited amount of data available at the start of the project. The understanding of the data was only possible by creating new features (conventionally done in the data preparation phase).

2.1 Available data

The raw data is sourced for the official Screeps leaderboard. This data is very limited in scope, it effectively contains nothing but the "Season", "user ID" and "Score". The data quality is extremely high: the data does not contain any NULLs or NaNs.

The following table is logged during the preprocessing of the raw data, and displays useful information with regard to the underlying data:

Description of raw data		
	score	rank
count	1.672.790	167.279
mean	22.362.920	948
std	71.502.860	580
min	1	0
25%	401.154	454
50%	2.391.110	930
75%	10.850.880	1.405
max	1.647.224.000	2.979

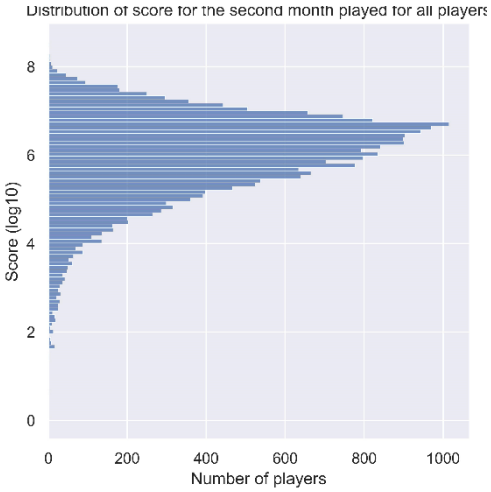
An important thing to note is that a player is only on the leaderboard if their score is greater than 0. This means that the a player is only actively playing if their score is greater than 0.

2.2 How to compare players to other players

The quality of the player's bot improves every month the player is actively playing the game. This means each active month of playing results in a greater score. This means the score of an individual should not be compared to players who've played the game for a much longer time. The quality of the bot of someone who has been playing for a long time is expected to be higher.

The creation of a new attribute "Months played" was required to enable fair comparisons between players who started in 2017 and players who started in 2023. This attribute is used as the X axis in time series comparisons instead of the "Season".

Bots gain exponentially more score the better they become. Turning score into a logarithmic (log10) showed score is effectively normally distributed (although it would be more correct to describe it as having an alpha distribution). Either way, this means tools which work on a normal distribution will also work on the logarithmic score of a player.



2.3 What does playing mean?

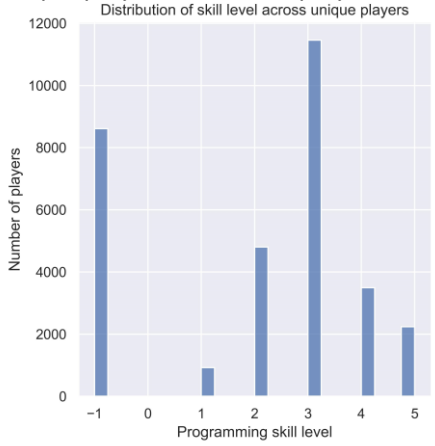
Playing the game Screeps is coding. Players play the game in a code-editor of their choice, where they write code. This code is used to run their bot after they deploy this code to the official Screeps server.

Coding is a very complicated and hard to learn skill. It can take years to become a competent programmer. This means there is a vast gap between a new programmer and a veteran programmer. There is a very real correlation between programming skill and Screeps score. This was visualized "2.2 How to compare players to other players".

This distribution of score allowed for the creation of a new feature called "skill level" based on the score a player has achieved in their second month. "Skill level" is rated on a scale of 1 to 5, in which 1 is someone who cannot even create a functional bot and 5 is someone who is able to code an amazing bot in two months. This was done by utilizing the measures for a boxplot:

- Level one is hardcoded to be a score of less than 10.000 as the lower fence is below zero: (-6169210). This means their bot has never worked properly after two months of playtime.
- level 2 between is awarded for a score between 10.000 and the Q1 (273093)
- level 3 is awarded for a score between Q1 (273093) and Q3 (4567963)
- level 4 is awarded for a score between Q3 (4567963) and the UF (11010266)
- level 5 is awarded for a score greater than the upper fence (11010266).

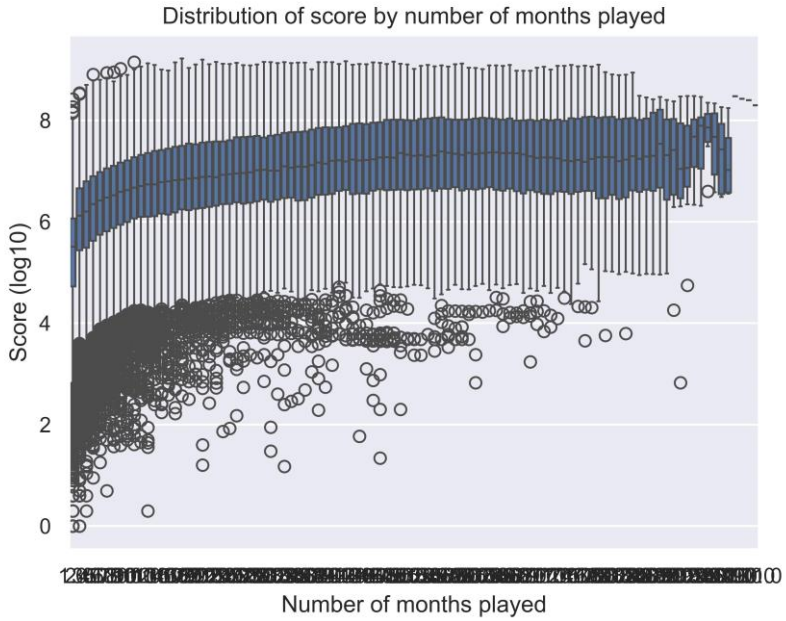
On top of this there is also a skill level of -1, which are all of the players who have not played for more than 1 month. These are players who are effectively out of scope for this project, as Screeps was simply not the game for them. This leads to the following distribution of skill for the 32.000 unique players who have played Screeps:



This feature will be used in many of the coming visualizations, as it ended up proving to be extremely powerful.

2.4 Score development over time

The score of the player base sharply increases in the first months of playtime, but it seems to stagnate after 12-24 months of playing the game. The distribution of score by month is displayed in the (hard to read) figure below. The X axis is illegible. The X axis starts at 1 and ends at 100. The alternate version of this figure is a heatmap, but time ran out and therefore it was not included in this version.



The stagnation is a potential feature with a lot of meaning. It might mean many things, like:

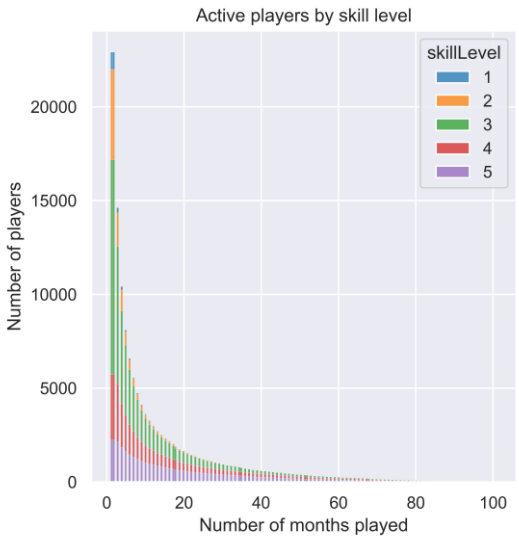
- The player has reached the maximum number of rooms they can control, and needs to improve their infrastructure.
- The player focusses on combat, the stock market, the power mechanic, expanding territory, base building, or other aspects of the game.
- The player is forced to optimize their CPU.

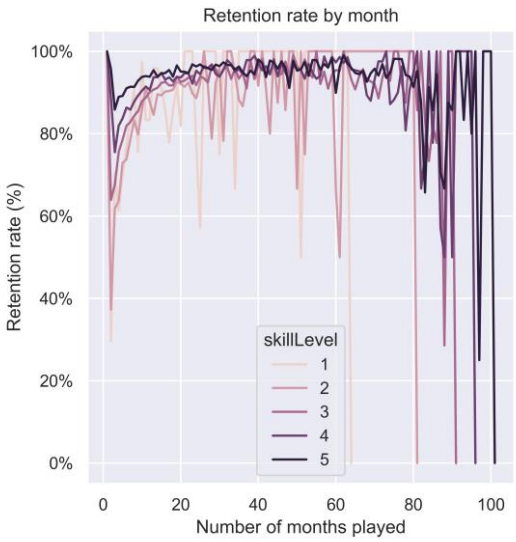
However, all actions should eventually lead to an increase in score. Therefore, the following assumption can be made with regards to score: “If someone’s score is not increasing, then a player is not playing (AFK)”.

This led to the creation of a new feature “months AFK”. This measure is the count of months a player has had a score below their highest score. The hypothesis is that this should have a high correlation with the likelihood of a player quitting the game (simply because their bot will die). The veracity of this claim will be discussed in chapter 3.

2.5 Bad developers quit early

One of the main conclusions in 2.4 is that the score seems to stagnate after 12-24 months of playtime. This is a case of "survivorship bias". The first meaningful conclusion drawn from the created features is as follows: "The higher your skill level, the more likely you are to keep playing". This is displayed in the figures below:





This is an exceptionally meaningful conclusion, as this means that players of a high skill level are more likely to keep playing the game (and therefore spend more money on the game). This means the preferred customers of Screeps are skilled programmers.

2.6 Investment should result in increased retention rate

Human brains are more likely to like something in which they have personally invested a lot of time, effort or money into. This is associated to the phenomenon called cognitive resonance/dissonance. This is relevant to the retention of players, because it implies players who have invested a lot of energy in Screeps should be more likely to be more positive about the game.

This positivity should manifest itself in two ways:

1. The player should be less likely to quit playing.
2. If the player quits playing, they should be more likely to return.

The aim is to quantify this investment by tracking the following measures for players:

1. The highest score they have achieved in their lifetime.
2. The total cumulative score they have achieved in their lifetime.

The correlation between these features and the retention rate of these players will be discussed in chapter 3.

2.7 Quitting by dying

A player's bot can die. If the bot dies, the player stops getting score for that month. There are 2 main reasons for a bot dying:

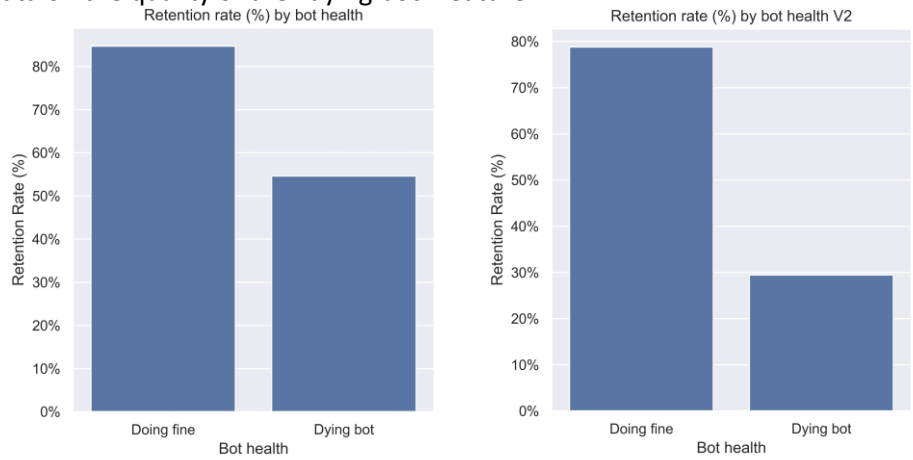
1. Another bot kills it.
2. A bug in the player's code breaks the bot.

The death of a bot tends to be a slow process for bots with more than one room. Good bots are resilient: they must be able to recover from catastrophic damage without direct player input.

A feature was derived from the score column in order to quantify these phenomena: “Bot health”. This feature is binary. Bots are either fine, or they are dying. A bot is dying if they have lost at least 50% of their score in comparison to the previous month. This is explicitly not using the logarithmic score used to determine skill level, as that would represent a significantly larger loss of score.

Graphing this column shows that 55% of players who’s bot has lost more than half of their score do not quit the game. This was significantly higher than expected. An investigation of the data quickly revealed that a significant portion of players who have stopped playing the game returned to play the game after 2-70 months. This happened in 13.650 out of 44.000 cases in which a player quit the game (this number is logged every time the preprocessing module in main.py is executed).

This lead to a second version of the same graph, in which a player can also temporarily quit the game. These graphs are showed side by side below, to display the impact of the exploration of the data on the quality of the “dying bot” feature:



It’s clear from the figures above that the “bot health” feature is an excellent indicator of “if a player is going to keep playing”. This means it has potential to be used in achieving the aim of this project: to identify if a player is going to leave, so Screeps can incentivize them to stay.

2.8 Returning players

The investigation into bot health in 2.7 has led to the discovery of another feature of key importance. A binary feature called “Returned”. This feature is True if a player who has quit the game ended up returning. If this project could identify players who are more likely to return then this information could be used by Screeps to focus on these players with advertisement campaigns, in order to get them to play the game again.

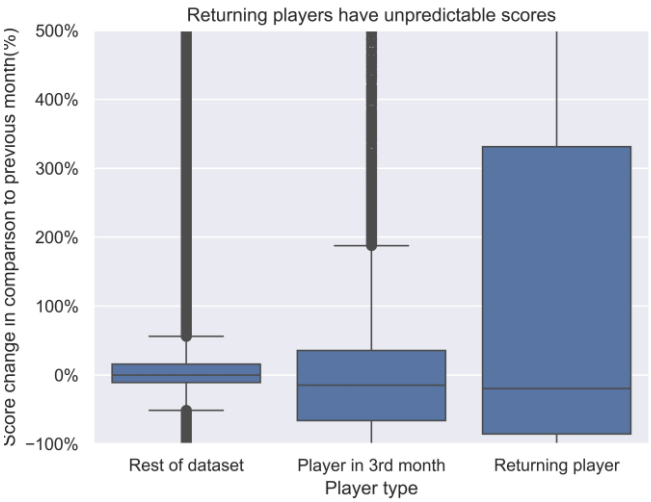
This means there are now three features for retention in the dataset:

1. Retention rate: “Is the player still playing today?” (yes = 1, no = 0)
2. Temporary stop: “Did the player stop playing for a few months” (yes = 1, no = 0)
3. Returning: “Did the player come back from having stopped for a few months”

The quality of these features and their correlation to the other engineered features will be discussed in chapter three.

One of the initial aims of the project was to predict the score for a given user for a given month. This ended up not being part of the final scope for the project, due to it having a lower priority. The foundation for the prediction of score was made, even though this prediction was not included in the final version of this project.

One clear problem was that returning players lack all of their existing infrastructure. This means their score would be significantly lower than usual. This would have ruined the accuracy of the hypothetical model. This is displayed in the graph below:



The graph below shows the score of a player can either significantly increase in comparison to their previous month, but it can also decrease. It shows there is a lot more variability in in the potential score of a player. The potential score increases are very large, even when compared to the third month of playtime in which the player is still building their bot.

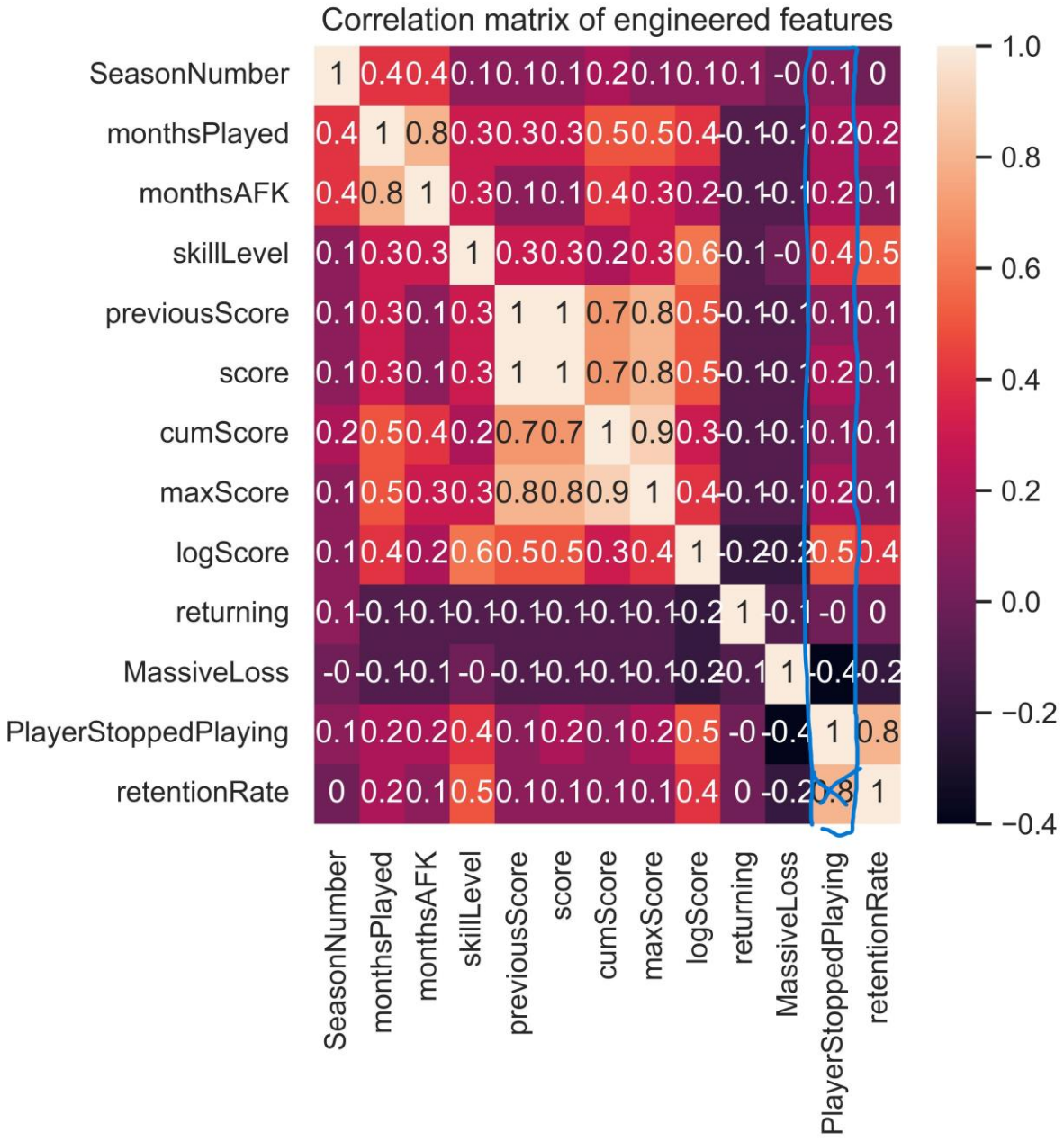
This means that all months in which the player returned from having previously quit the game, are considered noise with regards to predicting score. These should be filtered out of the dataset when trying to predict score.

3. Modelling and evaluation

The aim is to predict if a player is going to quit playing the game. This is a binary classification problem, which is to be solved with the data available. The model is selected by comparing a set of models to one-another and then picking the one which performs best.

3.1 Features for predicting if a player will keep playing

The initial aim was to use gridsearch to identify the best features for the model, however it became apparent that the dataset was far to large for this solution. This means that it was necessary to use other tools to determine the best features. The chosen tool (besides the data exploration described in chapter 2) was a correlation matrix:



The features which are going to be used for the model are the features “skill level”, “log score” and “Massive Loss (also known as bot health)”.

3.2 Things to note about the correlation matrix

The following things stood out when assessing the correlation matrix:

The months played feature seems to have no impact on the "retention rate" or "player stopped playing" features, even though a correlation was clearly visualized in the previous chapter:
This is most likely because the player is not on the leaderboard if they are not playing. If they are not on the leaderboard, then they are not in the dataset. And if they are not in the dataset they cause survivorship bias. This survivorship bias is why the correlation between these features appears low.

Months AFK has no impact on the retention of players:
The data seems to be telling us that the endgame of Screeps is not about getting more score, but rather about improving your bot to attain many of the other goals in the game.

Investment (cumulative score and maximum score) has no impact on the retention of players, nor does it correlate with the returning player column:
The hypothesis that there is a relationship between "investment" and the likelihood of staying/returning has been completely disproven.

It seems impossible to predict if a player will return with the current dataset:
The feature "returning" indicates if a player will return after having quit the game. All features in the dataset do not correlate with the "returning" feature. This means that it is not possible to predict if a player will return with the current dataset.

3.3 Model selection for binary classification

The aim is to pick the best model for the identification of "If a player is going to stop playing". This will be achieved by picking all lightweight classification models, and comparing them against each other. The models included in this section are as follows:

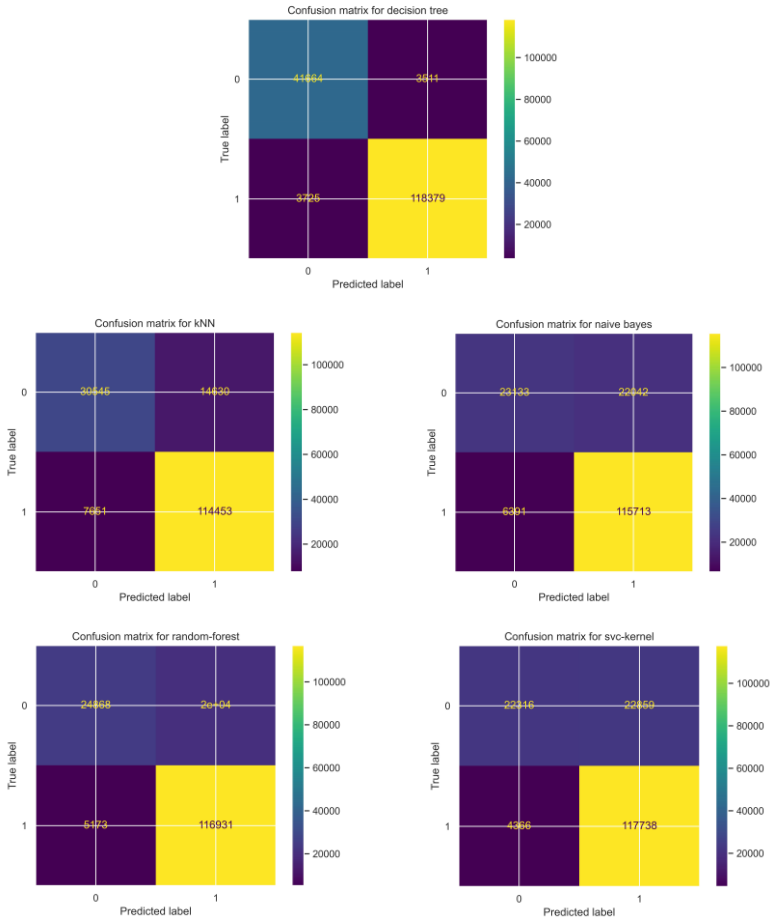
- Support vector machine
- Random forest (max depth = 7)
- Naïve bayes
- K nearest neighbors (neighbors = 5)
- Decision tree

The quality of these models is graded on two aspects: the F1 score and the confusion matrix.

The Random forest performs best based on F1 score:

Model performance	
Model name	F1 score
Support vector machine	0,837
Random forest	0,847
Naïve bayes	0,830
K nearest neighbors	0,832
Decision tree	0,787

However, the Decision tree outperforms the Random forest in the confusion matrix. This model performs almost twice as good as the other models on false positives and true negatives (see next page).



The decision tree will be used to classify if a player is going to leave.

4. Deployment plan

This section is purely hypothetical.

4.1 Output is CSV file

The end-product is going to be a CSV file generated by main.py in the output folder. This file will contain two columns: Player ID and skill level. This will contain all players who are going to quit playing the game in the month the file is downloaded.