Deck Sorcery

with **The Sunday Lunatics**

Project Report

Brought to you by:

THE SUNDAY LUNATICS

Tang Meng
Tan Jun Khiang
Tan Wei Lian, William
Leong Jun Hun, Darryl

Table of Contents

Executive Summary

Computer gaming is a billion dollar industry, characterized by video and mobile games which provide endless hours of entertainment to gamers. Nonlinear gamplays presents players with challenges that can be completed in a number of different sequences and could result in a variety of outcomes. Increasingly, game developers are leveraging on machine learning methods, such as supervised learning like support vector machines or neural networks to build the models of player experience to enhance gameplay.

Hearthstone, a online collectible card game (CCG) is a turn-based 1-vs-1 strategy game set in the Warcraft universe. In CCGs every player is asked to construct a specific deck of 30 cards before the actual match. With 9 different hero classes and over 2,000 possible cards to choose from, this is no doubt a daunting task. Each card include specific rules that affect the interaction between players, promoting a varied and dynamic game play. Players need to deal with hidden information and randomness, with the combination of states, rules and cards that may result in complex or unpredicted reactions, such as combos, combination of card. Hearthstone is a game where actions may have non-deterministic results, and the challenge for each user to maximize the possibility of winning using their constructed deck.

TSL has developed Deck Sorcery, a web based platform, which focuses on the automated deck construction and deck win rate optimization using information from historical dataset of decks. The actual gameplay mechanisms are not covered in this project but they can be considered in future enhancement of the intelligent system. Deck Sorcery is based on a Hybrid Reasoning System: Co-operating Experts architecture, considering the complex nature of deck construction and requirements for various techniques. Deck Sorcery leverages on a combination of machine reasoning techniques such as deductive and inductive reasoning, as well as planning & optimization in Synthetic Problem Solving.

The lack of any universal benchmarking methods for win rate prediction made assess the strength of the deck other than by human observation. Also, a multitude of factors such as the player’s intrinsic ability or the number of actions in the game are not considered in our analysis.

The technologies used in our POC solution are a combination of Flask, Python and external Python libraries such as Scikit Learn, DEAP and Apyori.

List of Abbreviations

Abbreviation	Abbreviation Description
CCG	Collectible Card Game
DEAP	Distributed Evolutionary Algorithms in Python
GUI	Graphical User Interface
MCTS	Monte-Carlo Tree Search
SVR	Support Vector Regression
TSL	The Sunday Lunatics
WR	Win Rate

Introduction

The use of Artificial Intelligence in Computer games is not new, with AI techniques being used in Computer Games dating back to the 1950s.[1] Recently, with advancements in computer hardware, more computing power has become available for executing AI algorithms in Games, giving rise to more advanced and sophisticated techniques, which can be executed within shorter response times.[2] This has allowed game developers to shift from simpler, Case Based Reasoning gameplays to a more dynamic gameplays which are more adaptive and appealing to the player.[3]

Hearthstone is a free-to-play online collectible card game (CCG) developed by Blizzard Entertainment. It is released in March 2014 and is available on Windows and OS X operating system as well as on iOS and Android mobile devices. It is a turn-based 1-vs-1 strategy game set in the Warcraft universe. There are 9 different hero classes[4] to choose from, with over 2,000 cards to build a deck of 30 cards upon. Players spend mana points to cast spells, weapons and summon minions to attack the opponent, with the goal to reduce the opponent's health to zero or below. [5]

While the cards and gameplay are most similar to a traditional physical card game, Hearthstone fully utilizes the capabilities of the digital environment. Thus, making it easy-to-learn for beginners while allowing a highly complex, dynamic and strategically competitive gameplay at the tournament level.

---

[1] "Artificial intelligence in video games - Wikipedia." https://en.wikipedia.org/wiki/Artificial_intelligence_in_video_games. Accessed 21 Apr. 2019.
[2] "AI Model for Computer games based on Case Based Reasoning and ...." 10 Sep. 2008, https://dl.acm.org/citation.cfm?id=1453242. Accessed 21 Apr. 2019.
[3] "Artificial Intelligence for Adaptive Computer Games - Semantic Scholar." https://www.semanticscholar.org/paper/Artificial-Intelligence-for-Adaptive-Computer-Games-Ram-Onta%C3%B1%C3%B3n/4460a2e7d93fb1a942aceec13c5b112d270c336f. Accessed 21 Apr. 2019.
[4] "Common deck types - Hearthstone Wiki." 15 Apr. 2019, https://hearthstone.gamepedia.com/Common_deck_types. Accessed 21 Apr. 2019.
[5] "Improving Hearthstone AI by Combining MCTS and Supervised ...." 14 Aug. 2018, https://arxiv.org/abs/1808.04794. Accessed 21 Apr. 2019.

Business Problem Background

One of the key features of CCGs is deck construction, which is personalizing a deck before any match. Deck construction is a challenge that involves a huge and irregular search space, with random and variable behaviour after any simple card changes.

Considering the large number of possible moves and the incorporated uncertainties in building game trees for these games, it becomes very difficult due to the exponential growth of the number of nodes at each level. Given the highly dynamic nature of the game environment, the challenge for the player is to construct a winning deck.

Project Objectives & Success Measurements

The Sunday Lunatics (TSL) targets to help automate the deck construction based on a specific hero class and a set of cards selected from the library by the users and optimize its deck win rate. The success measurements are:

- A Graphical User Interface (GUI) for players to generate a deck of cards, based on their card library of from all available cards in Hearthstone.
- A runnable standalone bespoke intelligent system that integrates
  - Deck Construction Rules *(Business rule reasoning techniques)*
  - Deck Construction optimization techniques *(Evolutionary computing through Genetic Algorithms)*
  - Deck Synergy Rules *(Association Rule Mining)*

Project Solution

Project Scope

TSL has developed the platform, Deck Sorcery which focuses on the automated deck construction and deck win rate optimization using information from historical dataset of decks. This is achieved via the use of a hybrid reasoning system which will be discussed further in the subsequent section.

The actual gameplay mechanisms are not covered in this project but they can be considered in future enhancement of the intelligent system. Potential future works will be discussed at the end of the report.

System Design

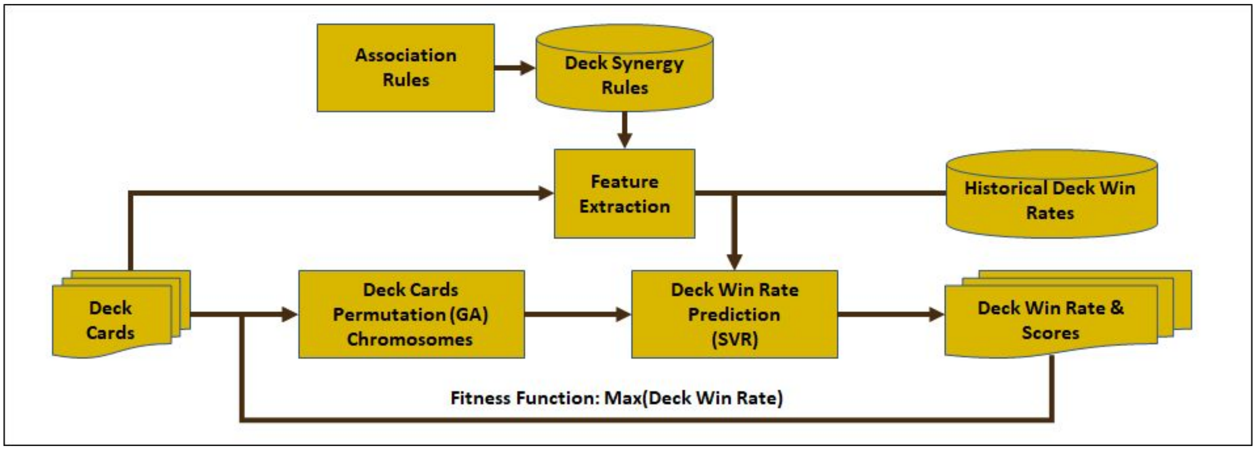
Hybrid Reasoning System

A Hybrid Reasoning System was developed to perform Automated Deck Construction. Deck Sorcery is based on a Co-operating Experts architecture. A Co-operating Expert architecture was chosen due to the complex nature of deck construction - different techniques are employed and no single techniques is sufficient. Deck Sorcery leverages on a combination of machine reasoning techniques:

- Deductive Reasoning: Rule Systems - for Deck Construction Rules
- Inductive Reasoning used in Data Mining & Machine Reasoning: Association Rule for Deck Synergy rule generation
- Planning & Optimization in Synthetic Problem Solving: Genetic Algorithms for Deck Building

The objective of the Hybrid Reasoning system is to build a deck with the highest possible win rate. The key components of Deck Sorcery system are:

- Rule Based System: Deck Construction Rules
- Association Rules: Data Mining to develop Deck Synergy Rules
- Genetic Algorithm: Genetic Algorithms usually work well with combinatorial problems, and solutions are usually a vector of elements, such as cards.



Deck Construction Rules

Each game match should be a 1-vs-1 battle between two decks and each player should choose only 1 hero before any match. Each deck should comply with the following rules:

- Exactly 30 cards
- Up to 2 of any non-legendary card
- Only 1 of any legendary card
- Class-specific cards only be playable by a hero of the same class.

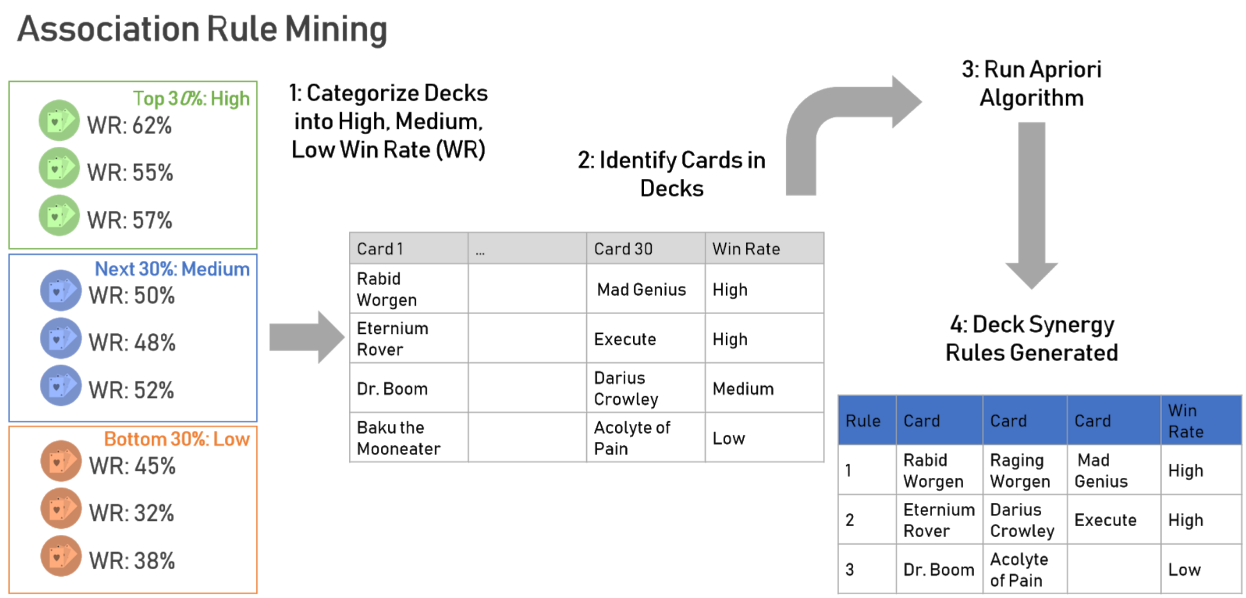
Association Rules - Deck Synergy

Association rule mining was performed to examine the evidence of deck synergy - specific combinations of cards which contributed to high win rates for decks. Historical Deck Win Rates was mined for patterns or co-occurrence of cards. The Apriori algorithm was to used to perform Association Rule mining and establish these Deck Synergy Rules.This was done through the usage of the Apyori package in Python.

The antecedent conditions for our Association Rule mining were the presence of a certain card, or combination of cards. The consequent was the categorical win rate - Low, Medium or High. For the purpose of our analysis, we considered rules where the consequent fell into all 3 categories to enable our model to penalize decks which could lead to low win rates.

For the Apriori algorithm to ingest the data required for association rule mining, data preprocessing was required. This involved categorizing the continuous variable of Deck Win Rate, this categorization was done by means of percentile - where the top 33.3% of Decks based on Win Rates would fall into the High Win Rate category and the next 33.3% into the Medium category. The decks were then transformed into their feature array of Cards, with their corresponding Win Rate. The Apriori algorithm was run against this to generate the Deck Synergy Rules.

The following diagram depicts the overview of the Deck Synergy Rules generation process for a single Hero type.



The following parameters are used in in extracting the association rules.

Minimum Support	0.25
Minimum Confidence	0.1
Minimum Lift	1

One consideration for the Deck Synergy rules is that the rules for each Deck type would differ because of different cards. Hence, separate deck synergy rules were developed.

The table below depicts a subset of Deck Synergy Rules. The complete Deck Synergy Rules for all Hero Types is provided in the data folder.

Rule	Antecedent Conditions: Cards in Deck			Consequent: Categorical Win Rate
Druid Rule #142	Wispering Woods	Archmage Vargoth	Savage Roar	HIGH
Warrior Rule #27	Omega Assembly	Dyn-o-matic		HIGH
Rogue Rule #136	Edwin VanCleef	Backstab	Eviscerate	MEDIUM

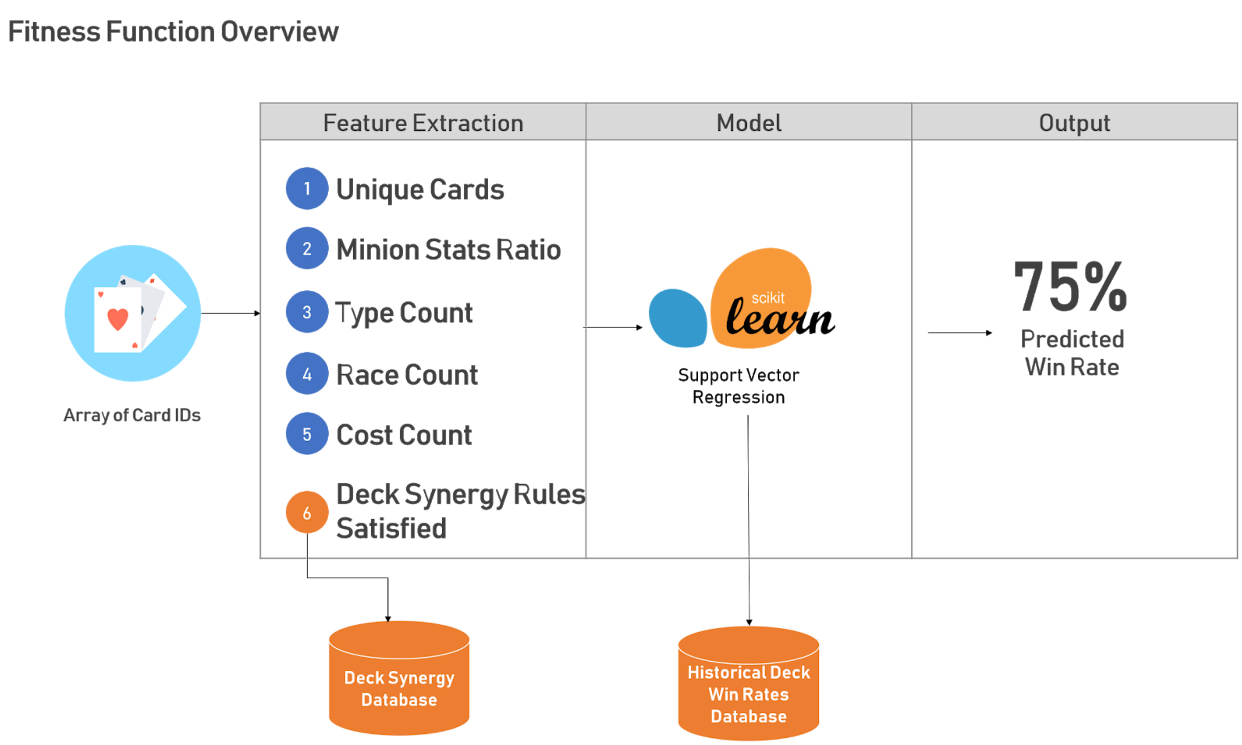
Priest Rule #8	Psychic Scream	Shadow Visions		MEDIUM

Fitness Function

The Fitness Function developed leverages regression analysis, which predicts the value of target. We utilize Support Vector Regression (SVR), which outputs the predicted Win Rate instead of Support Vector Machine. In SVR, we try to fit the error within a certain threshold, unlike simple regression, where the objective is to minimise the error rate.[6]

The rationale behind the use of SVR as part of the Fitness Function is the high level of uncertainty with regards to factors which contribute to the win rates of the deck. While some of the factors have been previously researched, TSL believes that these factors could vary significantly, depending on Hero Type. The use of SVR seeks to account for differences between the Hero Types. Decks generated by Deck Sorcery are transformed into features - regression analysis is performed using the features of historical decks to predict the win rate of the Deck constructed.

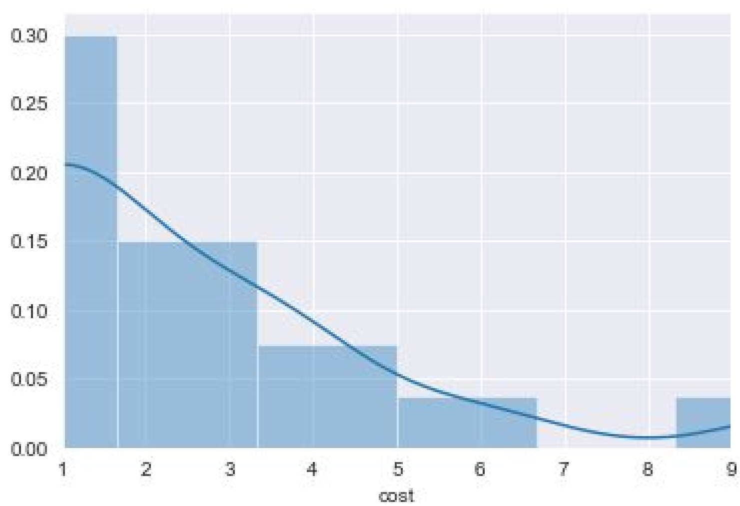
The overview of the Fitness Function is provided below:



The 6 features identified are:

[6] "Support Vector Regression Or SVR – Coinmonks – Medium." 28 Jun. 2018, https://medium.com/coinmonks/support-vector-regression-or-svr-8eb3acf6d0ff. Accessed 20 Apr. 2019.

1. **Unique Cards:** The number of unique cards in the deck. Previous research has evaluated that the number of repeated cards in the deck can influence the Win Rate[7]
2. **Minion Stats Ratio:** The ratio of the Sum of the Health and Attack score of all Minion cards in the Deck to the Cost of all the Minion Cards in the Deck. It is computed as following:
   SUM(Health + Attack) for all Minion Cards in the Deck/ SUM(Cost) for all Minion Cards in the Deck
3. **Type Count:** The number of card types present in the deck. Card Types of interest include are Minion, Spell, Weapon and Hero.
4. **Race Count:** The number of card races present in the deck. Card Races are Beast, Dragon, Demon, Pirate, Totem, Murloc,Elemental and Mechanical.
5. **Cost Count:** The number of cards based on their respective Costs. E.g. the number of Cost 1 cards, Cost 2 Cards. This feature seeks to represent the Mana Curve, which has been previously identified as a feature for consideration. A mana curve is a histogram plot representing a deck by counting the number of cards per cost. An ideal Mana curve should be skewed towards the left.[8]



6. **Deck Synergy Rules Satisfied:**The number of deck synergy rules, based on the win rate category (High, Medium, Low) contained within the deck.

Genetic Algorithm

Genetic algorithm is applied to the deck construction process, with the evaluation delegated to an engine that tests every potential deck against a range of human-made decks, provided by the sample historical data. Genetic Algorithms are deemed to work well with combinatorial

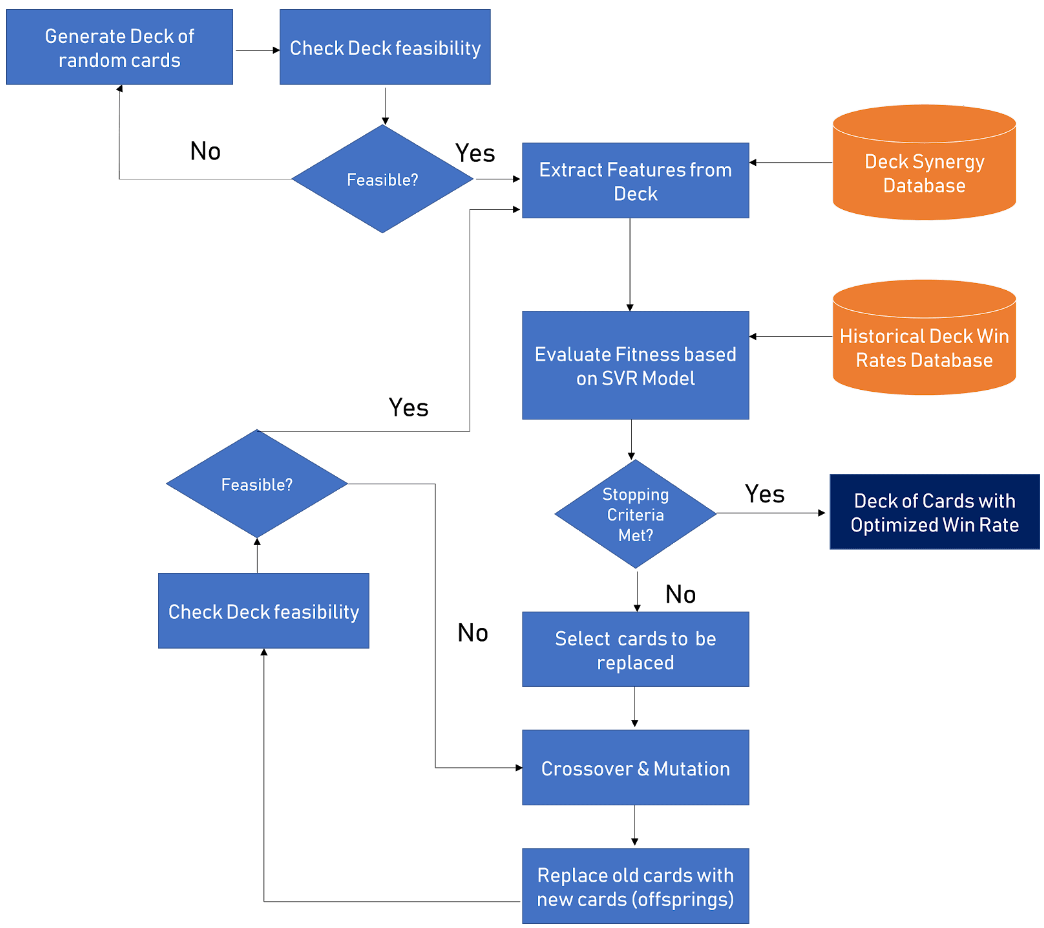
---

[7] "(PDF) Automated Playtesting in Collectible Card ... - ResearchGate." https://www.researchgate.net/publication/324767888_Automated_Playtesting_in_Collectible_Card_Games_using_Evolutionary_Algorithms_a_Case_Study_in_HearthStone. Accessed 19 Apr. 2019.
[8] "Automated Playtesting in Collectible Card Games using Evolutionary ...." https://www.researchgate.net/publication/324767888_Automated_Playtesting_in_Collectible_Card_Games_using_Evolutionary_Algorithms_a_Case_Study_in_HearthStone. Accessed 19 Apr. 2019.

problems, and solutions are usually a vector of elements. In Deck Sorcery, the elements represents cards and the vectors are Decks.[9]

Deck Sorcery leverages on the Distributed Evolutionary Algorithms in Python (DEAP), a computation framework. DEAP implements its evolutionary algorithm using pluggable genetic and selection operators.[10]

The stopping criteria for the running of the Genetic Algorithm is after 1000 generations. The following diagram details a high level overview of the implementation of the Genetic Algorithm:



The GA execution process is outlined below:

1. A random deck of 30 cards is generated
2. The deck is evaluated against Deck Rules (defined in the Deck Rules section) to validate that the combination of cards is possible and not deck rules are violated.

---

[9] "Creating Hearthstone decks by using Genetic Algorithms | GeNeura ...." 27 Apr. 2018, https://geneura.wordpress.com/2018/04/27/creating-hearthstone-decks-by-using-genetic-algorithms/. Accessed 21 Apr. 2019.

[10] "DEAP (Distributed Evolutionary Algorithm in Python ... - Springer Link." https://link.springer.com/article/10.1007/s10710-018-9341-4. Accessed 20 Apr. 2019.

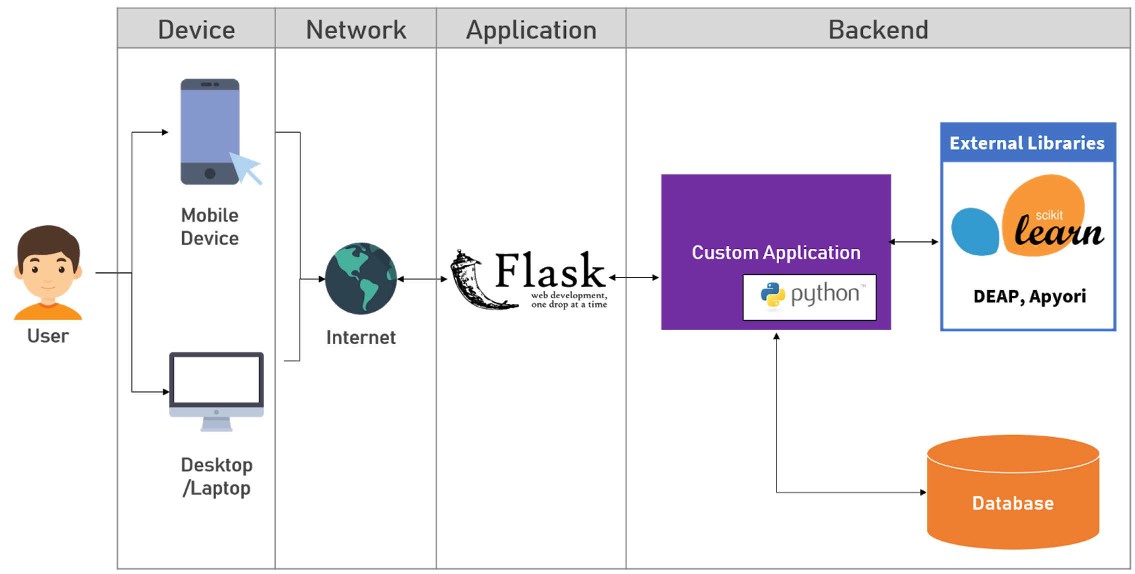
    a. If the deck violates any deck rule, return to step 1
    b. If the deck does not violate any deck rule, proceed to extract the features from the deck (Step 3)
3. Extract features from the cards. These features are detailed in the Fitness Function section of this document.
4. The extracted numerical features are then passed to the SVR model, which contains historical deck win rate. Against the model, a win rate is predicted.
5. Check for the number of generations
    a. If the number of generations is 1000, output the deck with the highest win rate
    b. If the number of generations is less than 1000, proceed to step 6
6. Select Cards to be replaced
7. Perform Crossover and Mutation
8. Replace old cards with new cards
9. The deck is evaluated against Deck Rules (defined in the Deck Rules section) to validate that the combination of cards is possible and not deck rules are violated.
    a. If the deck violates any deck rule, return to step 7
    b. If the deck does not violate any deck rule, proceed to extract the features from the deck (Step 3)

Key parameters for GA execution:

Population Size	300
No. of Generations	1000
Crossover Rate	0.5
Mutation Rate	0.2

Project Implementation

System Technical Architecture



The key technical components of Deck Sorcery are:

- **Python Application:** custom Python application, which leverages on external libraries such as scikit-learn for Support Vector Regression[11] analysis, Distributed Evolutionary Algorithms in Python (DEAP)[12] an evolutionary computation framework in the implementation of our Genetic Algorithms and Apyori[13] for Association Rule Mining for the development of Deck Synergy rules.
- **Flask:** Frontend web application framework with material design.
- **Database**: contains historical deck win rate data and Deck Synergy rules

Project Validation

Please refer to Project User Guide for step-by-step guide and validated functions of the Deck Sorcery platform.

---

[11] "1.4. Support Vector Machines — scikit-learn 0.20.3 documentation." http://scikit-learn.org/stable/modules/svm.html. Accessed 20 Apr. 2019.

[12] "DEAP documentation — DEAP 1.2.2 documentation." https://deap.readthedocs.io/. Accessed 20 Apr. 2019.

[13] "apyori · PyPI." 11 Apr. 2016, https://pypi.org/project/apyori/. Accessed 20 Apr. 2019.

Project Conclusion

Findings & Recommendations

Deck Sorcery is a simplified initial implementation of a Contemporary Reasoning System. The implementation of the Deck Building in web application seeks to encapsulate the complexity the Hybrid Reasoning System in a simple, intuitive user interface.

Associative Rule Mining was used identify synergies between cards in winning decks was used to develop Deck Synergy rules. The number of rules fulfilled by the generated deck, was considered alongside a variety of features. Some of these features, such as the number of unique cards in the deck and distribution of cost of the cards have been found to contribute to higher deck win rates. Deck Sorcery introduces new features such as Minion Stats ratio, Race Count and Type Count. The use of Support Vector Regression seeks to understand what are the attributes of a winning deck, through these features.

The implementation of Deck Sorcery is by no means a flawless one, numerous challenges were faced. There is a lack of any universal benchmarking methods for win rate prediction, making it difficult to assess the strength of the deck other than by human observation. Also, a multitude of factors such as the player’s intrinsic ability or the number of actions in the game are not considered in our analysis.

Future improvements or areas of research could include the use of Text Mining on Card descriptions, to identify commonality between decks. Monte-Carlo Tree Search (MCTS) algorithms, a heuristic search algorithm which are frequently employed in game play could also be explored.