Hearthstone Arena Card Classifier

Garret Lim
Computer Science
University of California, Riverside
Riverside,CA
glim005@ucr.edu

Alex Te
Computer Science
University of California, Riverside
Riverside,CA
ate001@ucr.edu

Ran Liu
Computer Science
University of California, Riverside
Riverside,CA
rliu@engr.ucr.edu

Introduction

Hearthstone is an online card game that was created by Blizzard Entertainment. In the game players can choose a class and build a 30 card deck consisting of spells, minions and weapons. In order to win, the player must reduce his/her opponent's health to 0. One of the game modes is Arena which allows players to draft their cards instead of building a deck from their collection. In the drafting phase, a player is given a choice of 3 classes and must pick one class. They are then given a choice between 3 cards and must choose one of those cards to put into their deck. The card selection repeats until the player has 30 cards in their deck.

Every couple of months, new cards are released into the game causing the first few weeks of gameplay following the release to be experimental until a tier list of is released showing which cards have been performing well. We are designing a solution to predict which new cards will be good picks in arena without having to wait for a tier list to come out which allows players to skip the experimental phase.

In this solution, we are taking data from the arena tier list of existing cards and training that data to predict the quality of cards from the new expansion.

Related Work

Defcon 22: Hacking Hearthstone

One of the tools developed was a way to find undervalued cards for Hearthstone. In order to do this, a card pricing model was formulated based on a card's statistics and text. Assumptions made were that "mana cost is proportional to its power", "power of cards increases roughly linearly", "card effects have a constant price", "A card has an intrinsic value", and "the cost of the card is the sum of its attributes." (Bursztein) This is similar to the approach that we used to form our features as we also used card statistics and text to determine if a card was good or not in arena.

Another part of their tool was a way of predicting an opponent's deck based on the cards they play. It uses n-grams to evaluate which cards go well together to model card synergies that are evident in deck building for the competitive ladder. In our case, this can be a useful tool to fine tune our arena card classifier as we are able to find cards that have synergies with each other and take that into account when classifying a card's usefulness for arena.

Helping AI to Play Hearthstone Challenge using Neural Networks

This project was a submission to the AAIA '17 Data Mining Challenge. The goal of this project was to predict the chance of winning a game for the first player given the game state and that it is his turn. This project also uses features of cards for its Data however, instead of evaluating cards, it is evaluating the player.

Initially the original data mining solution used, logistic regression, support vector machine and random forest achieving an accuracy of 79.32%, 78.83% and 78.4% respectively. Upon closer analysis it was concluded "that the problem is highly non-linear and more complex models should perform better" because "an SVM trained under a very small data sample achieved only a slightly worse result." (Grad) To deal with this problem, neural networks were used. Multiple models with different number of hidden layers were trained and in turn, each of these models were retrained multiple times. After that the best networks were selected. After that, ensembles of the best predictors were created. The final evaluation won with an accuracy of 80.18%.

While this project is for constructed decks, it serves as a possibility to extend to arena decks. Given the state of the arena game mode, this may be a more complex problem as

constructed decks tend to follow the same groups of cards whereas arena has a larger variety.

Proposed Method

The three methods we chose to classify the cards are K nearest neighbors (KNN), naive bayes, and decision tree implemented by Alex Te, Garret Lim, and Ran Liu respectively.

With KNN we hoped to classify a card's merit based of off their intrinsic distance from another card that was trained for our classifier to identify. The features we proposed to use were: mana, mechanic text, card type, mechanic type, bold word, attack, and health. Mechanic type, mechanic text, and bold words were interpreted in a special way because for KNN, we needed the data to be numerical (in order to calculate the distance). For example, a card can have a bold word (which is a special effect that the card possesses) but that needs to be mapped to a number, which we used https://hearthstone.gamepedia.com/Ability to help with. This goes the same for mechanic type and mechanic text, which were both difficult to get into numbers because they had to be interpreted by players, which can leave a lot of bias and in turn, skew the data. After getting the data in the correct format, we trained our KNN classifier with a small training set, and passed in a testing set to check the accuracy. This yielded a 52% accuracy which is not very good if we want to correctly classify cards. The next approach was to use feature selection with these seven features. The results showed that four features were better, namely mana, mechanic text, card type, and attack. However, this only improved the classifier by 3.19%. What we concluded after using feature selection was that KNN is not a good classifier to use when determining the merit of a card. This maybe because there is more to a card than what can be observed and measured. For example, a card may appear as bad, but when combined with another card in a specific set, then it may be the best card in the game. Our calculations did not account for that which may explain the low success rate.

In the naive Bayes approach, we wanted to rate a card's value with the assumption that all features of a card are independent from one another. As stated in the KNN explanation, we chose to use mana, mechanic text, card type, mechanic type, bold word, attack, and health for our features. mechanic text, mechanic type and bold words start out as text and to generalize that for the sake of this implementation, we decided to use the website linked in the previous explanation. As many cards in hearthstone are unique, we may find that some features, specifically the ones that started off as text, have a chance of 0 probability. An example is the bold word, "twinspell", which occurs in our testing set which is from the new expansion of hearthstone cards. This bold word is not found in any of the previous sets. To counteract this, we implemented laplace smoothing into our classifier. Initially, naive Bayes had an accuracy of 58%. In order to improve this accuracy we used the features generated from feature selection as a baseline and were able to improve the classifier to 62%. After a little more fine tuning, we ended up using mechanic text, mechanic type and bold words for our features which improved the classifier further to 67.4%. This low percentage may be attributed to the issues discussed above where some features had to be interpreted by players. Finding a better way to generalize cards may increase the accuracy of this classifier.

In the decision tree approach, we needed to build a decision tree by choosing a variable at each step that best splits the set of data. In our case, the measurement of "best" we choose is gini impurity, which gave us the likelihood of an incorrect classification of a new instance (choosing randomly from the dataset) of a random variable. For decision tree, the features that we used were: mana, mechanic text, card type, mechanic type, bold word, attack, and health. Mechanic text, mechanic type, bold word were were interpreted as numbers that represent real type for decision tree. There are so many card that have similar mechanic or other abilities and the number will help the tree to make a leaf more reasonable. After data cleaning, the first try of decision tree gave us 57% accuracy. We started to improve the algorithm by feature selection. It turned out to be different from KNN and bayes, the results shows an accuracy of 64%. But we didn't satisfied with the accuracy we got. We kept on improving decision tree by changing the max depth and min size of the tree to generate different trees. After plotting the graph of the connection of max depth and accuracy we choose 8 to be the max_depth as it improve our classifier to 67%. We did the same thing among min size, then the best accuracy we could get is 72.59%. As the accuracy was still not good enough for real gameplay assistance, there must be some reason.

Experimental Evaluation

The three classifiers that we have used (KNN, Naive Bayes, and Decision Tree) all varied in results, all of them did not have a good classification accuracy, however the best of the three was the Decision Tree. It was able to classify a card correctly 72.59% of the time. In a real world situation this would not be good enough because most people strive for at least 95% accuracy. With this in mind, we came to the conclusion that classifying a card, with only what the card contains, independently from any other cards is very difficult because the face value of a card is not only what is used to determine the merit of a card. Some cards, especially those of legendary rarity have unique effects that cannot be generalized. These effects can be powerful but because no other card has that effect it may not train well and thus could get a wrong evaluation. In addition, a lot of cards synergize together and if there is no synergizing, most of the time the card(s) by themselves are complete useless. This is exactly what happened when we were classifying the cards. We were taking the card's face value and not including other combinations of other cards. This would be a difficult problem because we would then need to connect every card to every card in Hearthstone.

Discussion and Conclusion

Overall, it is very difficult to use a program to classify card since there is more to the merit of a card than what can be seen and measured. The best classifier to do this in our test was using decision trees, which had a 72.59% success rate. We believe that this may have the highest success rate because it is not calculating based off of distance, but rather checking to see if the components of a card are actually good based off each split.

Resources such as Icy-Veins lets players know how good a card is by having profession players label the card as good or bad. This is more effective and reliable because programs do not know the current state of game and what cards synergize with others. Knowing how cards react to others is very important because then the card's value is not what is only seen when looking at a picture. With the player's knowledge and experience of the game, they can infer how cards will perform in a game and combine that knowledge with statistics such as mana cost, attack, and mechanics to give the best possible merit for any given card.

REFERENCES

Code

https://elie.net/blog/hearthstone/how-to-appraise-hearthstone-card-values/

https://elie.net/blog/hearthstone/predicting-hearthstone-opponent-deck-using-machine-learning/

https://annals-csis.org/Volume_11/drp/pdf/561.pdf

https://hearthstone.gamepedia.com/Ability