Timothy Barao

Jonathan Clow

Andrew Sosa

CIS5930, Data Mining

Dr. Shayok Chakraborty

Proposal: Deck Recommendation Algorithm for Hearthstone

Research Question

As artificial intelligence continues to seek new and interesting problems to challenge existing systems and algorithm design, so has the collectible card game Hearthstone become a test bed for AI algorithms and machine learning techniques. Prior literature has investigated using machine learning to determine the contents of an opponent’s deck. However, these techniques require a preconstructed deck composition to effectively compete with players. Constructing decks that remain competitive is a unique challenge in and of itself, involving several factors such as card synergy and opportunistic effects that exist beyond the player’s control, whereas the collection of cards the player has collected can change over time. In this case, several known repositories of player constructed decks exist that provide competitive compositions for high level play. Using these sources as well as collected data of what cards exist in a player’s own collection, we can devise a recommendation system to suggest not one or two recommended outcomes for a player, but a full constructed deck of thirty cards using common compositions to enable the player to compete at a high level with what they have.

Literature Survey

Algorithm

Collecting information on deck compositions is a simple as copying a “deck code” from popular deck building websites such as HearthPwn (http://www.hearthpwn.com/). These deck codes can then be parsed to list a deck composition for a given deck, which fills a record in our data set. These deck codes will also give us the associated hero class of the deck, filling one half of our deck classification strategy.

To classify decks, we use two attributes: the associated hero of the deck, as mentioned above a (limited to Mage, Hunter, Priest, Paladin, Rogue, Warlock, Druid, Warrior, and Shaman), and the relevant meta type (i.e., Pirate, Secret, Quest, Aggro). Deck codes come as a string of characters that can be decomposed into a deck list, comprised of between fifteen and thirty individual cards with quantities between one and two each due to deck restrictions. Note, however, that decks will be limited to decks that are valid in the current “Standard” playset. As this is the set of cards available for legal tournament play, these cards will be considered for construction of the highest win rate decks available to play of the highest caliber, and will represent a set of cards factors less than the full set of cards available to players within Hearthstone.

Once the decks have been decomposed from their deck codes, we can insert each as a record in a table. These records can then be used to build classification models for both decks for individual heroes and meta types within those hero subsets. Being that each deck may contain different cards, if a deck record is added that includes a card not prior found, we will add a column and mark values for all other decks as ‘0’, whereas presence of a card will be counted by the quantity of the card in the deck. Using these card counts, we can build a model for each hero and meta type within that hero’s set of decks based off presence of cards to find the similarities between decks. These models can then be used to classify new decks if information is missing. A model will also be generated from all decks included in the classification set to determine common elements among all decks. While this model is anticipated to include no cards specific to any hero archetype, it is anticipated to contain a number of valuable “neutral” cards that are used commonly.

Using the classification models for heroes and deck meta types, we can then make recommendations for decks a user can create using the user’s collection. This can be accomplished similarly to other recommendation algorithms, whereas this experiment will consider presence of values within a user’s collection to make recommendations, especially given that a user’s collection will include vastly more cards than any deck. However, using the models built prior, we can make recommendations for decks that the user is able to build or most closely build from archetypes classified prior. This can take place through a two-pass process: in the first pass, the user’s collection is analyzed to determine the deck archetype they are most likely to be able to build from the cards they possess, and if their collection is no sufficient to build a full deck of thirty cards, a second pass will identify common elements of all decks that they possess to fill out the remaining thirty cards. Collections will be sampled from those collected by the researchers and their cohort, both at current time and limited to the cards available to new users, with the collections themselves collected via the use of a third-party API if possible and manual entry if not. The decks generated will then be played in the Heathstone’s “Standard Ranked Play” mode, as this will capture the highest level of play for the decks, and win rates will be tabulated and compared to the deck archetype model that the deck was constructed from to evaluate success.

Experiments

Expected experiments include:

* Construction of deck archetype models, by computation of similarities among decks within archetypes and within all decks to generate a general model
* Classification of new unclassified decks against these generated models
* Recommendation of new decks through collection availability
* Playtesting of recommendation generated decks and comparison against archetype win rates

Datasets include:

* Deck compositions of high win rate decks within current Standard card set, as gathered via deck code from popular deck building sites (i.e. HearthPwn)
* Player collections as provided by reseachers, cohort members

Timeline

* October 10
  + Proposal submission
  + Deck code collection to compose into training and test sets
  + Deck code decomposition script translation from Python to MatLab begins
  + Dataset population begins
  + Recommendation algorithm research begins for adaptation to experiment
* October 17
  + Deck code decomposition script translation from Python to MatLab ends
  + Deck similarity model generation begins
  + Collection dataset users contacted for approval
* October 31
  + Deck similarity model generation ends
  + Deck classification algorithm completion and testing against test set
  + Collection dataset generation begins from prior contacted users
  + Collection similarity and deck recommendation algorithm implementation begins
* November 14
  + Collection similarity and deck recommendation algorithm ends
  + Deck archetype recommendation system tests begin
  + Recommended deck playtesting begins
* December 3
  + Experiments conclude
  + Construct report and findings
  + Code set finalization and preparation for submission begins
* December 10
  + Final report due
  + Code submission finalized and submitte d