Proceedings of the Federated Conference on Computer Science and Information Systems pp. 197–200
DOI: 10.15439/2018F363
ISSN 2300-5963 ACSIS, Vol. 15

Predicting Win-rates of Hearthstone Decks: Models and Features that Won AAIA'2018 Data Mining Challenge

Quang Hieu Vu
Data Science Group,
Zalora, Singapore
quanghieu.vu@zalora.com

Dymitr Ruta
EBTIC, Khalifa University
Abu Dhabi, UAE
dymitr.ruta@kustar.ac.ae

Andrzej Ruta
ING Bank Slaski,
Katowice, Poland
andrzej.ruta@ingbank.pl

Ling Cen
EBTIC, Khalifa University
Abu Dhabi, UAE
cen.ling@kustar.ac.ae

***Abstract*—Success of many computer games depends on designing a robust and adaptable AI opponent that would ensure the games continue to challenge, immerse and excite the players at any stage. The outcomes of card based games like "Heartstone: Heros of Warcraft", aside the player skills heavily depend on the initial composition of player card decks. To evaluate this impact we have developed an ensemble prediction model that tries to predict the average win-rates of the specific combination of bot-player and card decks. Our ensemble model consists of three sub-models: two Logistic Regression models and one Deep Learning model. The models are trained with both provided data and additional data about the cards, their health, attack power and cost. To avoid overfitting, we employ a trick to generate predictions for all possible combinations of opponent players and decks and obtain the result as the average of all these predictions.**

I. Introduction

Success of many computer games depends on designing a robust and adaptable AI opponent that would ensure the games continue to challenge, immerse and excite the players at any stage. In the history of game development, much efforts have been dedicated to the design and implementation of such a bot player. There are some very powerful examples of AI robots already, such as the famous "Deep Blue" that has defeated the world chess human champion. "Heartstone: Heros of Warcraft" is a turn-based card games between two players who select their heroes with a unique power and construct a deck of thirty cards that represent various spells, weapons, and minions, and can be summoned in order to attack the opponent with the goal of reducing the opponents health to zero and win. The outcomes of this game, aside the player skills, heavily depend on the initial composition of player card decks.

With the purpose of designing such robust and adaptable AI opponent for Heartstone, an important task is to correctly predict the marginal winning probability of AI players, given unseen decks. This very task has been defined as the target of the AAIA'18 data mining challenge. Specifically, the objective was to construct a prediction model that can learn win chances of the specific AI bots assigned to specific new card decks, based on the historic evidence of same AI bots playing with similar decks [1] against all kinds of players and decks extracted from hundreds of thousands of automated games. In this competition setup the training stage involved observing four AI players assigned to one of 400 available decks of 30 cards, battling each other in over 300000 automated games. The deviced prediction models were expected to use this data to learn how particular cards are played by the bots and evaluate their contribution or impact on the final win-rate estimate of specific decks-players. The competitive models are evaluated on the games with the configurations of the same 4 bot-players and 200 new card decks.

To solve the challenge we followed a pragmatic sequence of steps that could be in fact considered generic best practices for any competition involving predictive model build from data: understand the data, perform exploratory data analysis, conduct feature engineering and select features for the model, construct models, evaluate them and optimize the complete pipeline to maximally improve the predictive performance.

Since our solution turned out to be the best and won the 1^{st} place in AAIA'2018 data mining competition, we provide detailed information of how we follow these steps to design the top model. In addition, we also introduce a key technique that we have employed to avoid overfitting, which we believe was instrumental in winning the challenge. In general, our paper offers the following two major contributions:

- We provide a clear demonstration of how basic steps in designing a machine learning model should be executed: from data understanding and feature engineering to model design, parameter tunning and model's improvement.
- We present a critical method to avoid the overfitting in reconstructing regression based win-rate from large number of simple classification models and experimentally verify how it results in significant gains in testing set performance.

In rest of the paper is organized as follows. In Section II, we introduce related work. In Section III, we describe the problem, available data and features that we generated from the dataset to train our models. In Section IV, we present both our single models and how to combine them to form an ensemble model. We discuss how we avoid overfitting to improve model's accuracy in Section V. Finally, we conclude the paper in Section VII.

II. Related Work

In this section, we briefly introduce the machine learning techniques used in our model: Deep Learning, Logistic Re-

gression and Ensemble model.

A. Deep Learning

Deep Learning (DL) refers to a class of machine learning tech- niques and architectures, where many layers of non-linear information processing stages in hierarchical architectures are exploited for representation learning [2]. In particular, a DL network represents a multi-layer neural network with the deeper structures compared to the shallow models like Support Vector Machines and a specific method where the data is processed at and in between layers. Even though the concept of DL was introduced long time ago, it has only recently gained enormous popularity due the lower cost of computing hardware, the increased speed of chip processing, and recent advances in DL algorithms. DL has been successfully employed for computer vision, optimization, pattern recognition, signal processing, and natural language processing [3].

B. Logistic Regression

Logistic regression, developed by David Cox in 1958 [4], is a statistical method for regression analysis to describe the relationship between one dichotomous dependent variable (outcome) and one or more independent variables (predictors or features). Binary logistic model can be used for estimating the probability of a binary response based on predictors and gain insights on the presence of which factors increase the probability of a given outcome by a specific percentage.

Logistic regression has been widely used in medicine, e.g. to assess injury mortality or severity for patients [5], [6], [7], [8], [9], or help to diagnose some diseases like diabetes and coronary heart disease based on characteristics and physiological data of patients such like age, sex, body mass index, blood test results, etc. [10]. It has also been successfully applied in various areas, e.g. predicting votes of American voters based on their characteristics like age, income, sex, race, state of residence, previous votes, etc. [11], estimating probability of failure in various processes, systems or products [12], [13], predicting customers' propensity to purchase a product or cease a subscription in marketing applications [14]. Conditional random fields, the extended, sequential version of logistic regression for labeling or parsing sequential data, have been commonly used in natural language processing, biological sequences prediction, computer vision, etc. [15], [16], [17].

C. Ensemble method

Ensemble methods are learning algorithms that construct a set of classifiers and then classify new data points by taking a (weighted) vote of their predictions [18]. Similar to XGBoost, ensemble method is a popular technique employed by winning teams in Kaggle's machine learning competitions [19]. In our work, we employ a simple average ensemble method from a deep learning model and a wide learning model. While the idea of leveraging deep and wide learning has already been introduced by Cheng et al. in [20], our work is very different from their work because we combine deep and wide learning in an "ensemble method" to serve the purpose of binary classification. Cheng et al. do not utilize ensemble method, instead, they jointly train wide linear models and deep neural networks to combine the benefits of memorization and generalization for recommender systems.

III. Data Description and Feature Engineering

Before performing feature engineering, it is important to understand the original data in detail and exploit all available prior knowledge. Thus, the first part of this section is reserved for data description followed with a discussion of important features used in our model.

A. Data description

Available training data represented a collection of a total of 299680 labelled games played between 2 players from a set of total 4 bots: {A1,A2,B1,B2}. In each game, both bot-players, used one of 400 card decks, each including exactly 30 cards associated with specific hero class. Throughout the games 348 distinct cards and 9 distincts hero classses were observed. In each deck, a card must belong to either the hero class which the deck relates to or the neutral class.

The testing set included in turn only the composition of 200 new testing card decks linked to the same set of up to 9 hero classes and again the same 4 bot-players {A1,A2,B1,B2}. No details of the testing games were provided and the competition task was to estimate accurate win-rates for all 800 combinations of 4 bot-players playing off 200 testing decks.

In addition to the games inital setup data of players, decks and the result, training data also included detailed turn-by-turn gameplay data but since the same was not available for the testing set it was simply ignored.

B. Important features

Given that the task was to predict the likelihood of wining a game played by a particular bot using a specific deck, it was clear that the bot-player, hero class and the cards from the linked deck were the first choices for important features. Two types of features were extracted from the cards as follows:

- Card cardinality features: represented simply the number of cards of each type present in the deck and their observed values: {0,1,2} were mostly capturing just the presence of a specific card in a deck. We have narrowed the available set of cards down to 296 that appear in both training set and testing set since it does not help the model if we train on certain cards that do not exist in the testing set and vice versa.
- Card property features: each card has a set of properties that describe the *cost*, *health*, *attack* and *armor* of the card. In addition, there are properties to specify the card type (*hero*, *minion*, *spell*, and *weapon* [1]) and card rarity (*free*, *common*, *rare*, *epic*, and *legendary*). Since these properties are good indicators of the card's strength, we also consider them as features. However, if we build such a set of features for each individual card as card cardinality features, our number of features will increase significantly and is not manageable. Thus, we chose to generate card property features from the statistics of card properties in terms of the summary of card property values and maximum values

[1]Note that we do not consider two card types: *enchantment* and *hero power* because they do not help to improve our model's performance.

from certain card properties. In total, we generated 17 card property statistics features for each deck.

Given that the card properties are not provided in the dataset, we have to collect such information from a Hearthstone API website http://hearthstoneapi.com/.

IV. Win-rates Prediction Model

In our previous work, we proved that ensemble model usually outperforms single models in binary classification [21]. Thus, we continue to choose ensemble as our final model. In this competition, we constructed the ensemble model from three different models: two logistic regression models and one deep learning model.

A. Logistic regression models

Our first two models are based on the basic logistic regression method. As discussed in the above section, we have a total of 316 features: 1 feature for the bot id, 1 feature for the deck's hero and 297 cardinality features for the cards themselves and 17 statistics features from card properties. Initially, we trained a single logistic regression model on all features. However, we soon realized that training the model separately on either players or heroes leads to better result. Thus, we decided to employ two different logistic regression models as follows:

- The first logistic regression model consists of 4 sub-models trained separately on 4 different bots, each using 300 features KBest selected from 316 available features. Note that while we chose to use 300 features for the sub-models, these features are not the same for all sub-models. In our implementation, we used KBest to select 300 features from the training data for each sub-model and the selected features are slightly different from sub-model to sub-model.
- The second logistic regression model includes 9 sub-models trained separately on 9 different heroes of the deck, each using 100 features KBest selected from 316 available features. Similar to the above 4 sub-models trained separately for each bot, the 100 feature sets of each sub-model here are also different based on the feature selection returned from KBest method executed before training the sub-models.

It is interesting to see that the number of features used by the second model is much smaller than the number of features used by the first model (100 compared to 300). It is simply because for each sub-model of the second model, since the main difference between training data are only in cards belonging to the deck's hero or neutral cards, the total number of useful features is small.

B. Deep learning model

The third model is a deep learning model. For this model, we simply constructed a network with 5 Dense layers: 200, 100, 50, 25 and 1. For the first four layers, we used the basic *relu* activation. With the last layer, since this prediction issue is a binary classification, as expected, we used *sigmoid* for it. We compiled the model with *adam* optimizer. Our model was trained on all 316 features. For the epoch and batch size, we implemented a grid search to search for optimal parameter settings among epoch 5, 10, 15 and batch size 1,000, 2,000, 3,000, 4,000, 5,000, 10,000, 15,000, 20,000 and based on the experimental results, we chose to train the model in 5 epochs using batch size 5,000.

V. Overfiting prevention and model improvement

In this section, we will introduce several tricks we used to improve the model performance as well as to prevent overfitting.

A. Model improvement

We observed that we have two players in each game. However, so far we have only used data of the first player to train the model. We know that having more training data usually improves the model's performance. Thus, we simply tried to include training data for the model from both players of the games. It means that for each game, we use features of player 1 as the first sample and features of player 2 as the second sample. This way, the training set doubled in size up to 599,360 samples and the accuracy of our models improved from 0.4 to 0.5 point.

B. Overfitting prevention

Since the task of the competition challenge is to predict the likelihood of winning a game played by a bot using a specific deck, our initial models were trained using only data of a single bot or deck's hero as described in the above section. However, it is generally expected that the prediction accuracy would be better if we also consider information/features of the opponent player (as in practice some particular player may have better result when playing with certain players and vice versa). The problem of having extra features from the opponent player, however, is that we do not have opponent information in the test set to generate predictions. It means that to leverage information of the opponents in training the model, we need to find a way address this issue.

Fortunately, since the competition challenge description says that the opponents of games in the testing set are only selected from the set of provided 400 decks used in the training set, we decided to use a brute-force strategy to generate extra information for a test sample by considering all possible combinations of opponents, which gave us 4 * 400 = 1.600 test cases. The prediction results of these 1.600 test cases are then averaged to obtain the final prediction of the test sample. It means that given a test case of a bot and a deck, we generate a total of 1.600 test cases for that pair of bot and deck again 1.600 possible cases. It also means that we increase the size of our test size 1.600 times. In summary, the following changes were made in model training and predictions generation processes:

- We double the size of the features used in our training models, considering similar features of the opponent, in addition to features of the players. Specifically, for the two logistic regression models, we respectively used a total of 600 and 200 features selected by KBest selection method. On the the hand, the total number of features used by the deep learning model is 632.

- Given that the input size of the deep learning model has been double, we added an extra Dense layer size 400, again with *relu* activation, to the existing model, making the total number of layers to 6 instead of 5 as in the previous one.
- For the predictions, given each pair of a bot and a deck in the test set, we generated predictions for all 1,600 possible games between the bot using the deck against 4 bots * 400 decks. The final prediction is the average result obtained from these games's predictions

Note that while we double the size of the features by considering features from both the player and its opponent, we keep the number of training samples unchanged at 599,360 samples. It is because we see that training data of a bot *X* playing in a game with a bot *Y* is still different from training data of the bot *Y* in a game playing with the bot *X*.

From our submissions to the public board, we could see that this technique helped to improve our scores from 0.3 to 0.5. In addition, while we were only in the 4^{th} position of the public leader board with a gap of almost 1.0 point (a big gap) compared to the team in the 1st position, since our solution did not suffer much overfitting (which we believe that due to this strategy), we jumped to the 1st position and became the winner of the competition when the final ranking list was released.

VI. Implementation and Evaluation

We implemented all models in Python. For the logistic regression models, we relied on scikit-learn library, http://scikit-learn.org/stable/index.html. On the other hand, for the deep learning model, we employed keras.io library, https://keras.io/. Our single models respectively got score of -5.8892, -5.9148 and -6.0017 and the final ensemble model, which used a simple weighted average with coefficients 1.1, 1.08 and 1.0, reached the score of -5.4017 in the public leader board, a good improvement of approximately 0.5 compared to the results of single models. Note that we simply chose the co-efficients of the ensemble model based on the performance/scores of single models in the public leader board. Actually, if we had had more time, we could have tried a stacking technique to implement the second final layer model getting results from the first prediction models. In this way, we can also optimize the co-efficients and could even lead to a better result.

VII. Conclusion and Future Work

This paper has introduced not only our winning model from the AAIA'2018 data mining challenge, but also details of a step-by-step model building process, from the early problem and data understanding through feature engineering up to the model fine-tuning and ensembling the single models for improvements. We also prsented several techniques that we used to improve the model performance avoided model overfitting with that seemed to be critical in receiving excellent final testing score and wining the competition.

A. Future work

Even though our final result is good, as discussed in Section VI, we believe that there is still room for improvement if we have a better approach to ensemble our three single models into a final one. Besides, as discussed earlier, we did not try to utilize the massive amount of data on how games are played between the two players. Actually, we believe that the detailed game information once extracted and mined properly can provide some useful information about the tactics of the bots on playing with certain decks, and hence could be also useful to improve the model's performance.

References

[1] AAIA'18 Data Mining Challenge: Predicting Win-rates of Hearthstone Decks, https://knowledgepit.fedcsis.org/contest/view.php?id=123

[2] L. Deng, "Three Classes of Deep Learning Architectures and Their Applications: A Tutorial Survey", *APSIPA Transactions on Signal and Information Processing*, 2012

[3] Y. Bengio, A. Courville, and P. Vincent, Representation learning: A review and new perspectives, *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 17981828, 2013.

[4] D.R. Cox, "The regression analysis of binary sequences (with discussion)," *J Roy Stat Soc B.*, vol. 20, pp. 215242, 1958.

[5] C.R. Boyd, M.A. Tolson, and W.S. Copes, "Evaluating trauma care: The TRISS method. Trauma Score and the Injury Severity Score," *The Journal of trauma*, vol. 27, no. 4, pp. 370378, 1987.

[6] M. Kologlu, D. Elker, H. Altun, and I. Sayek, "Validation of MPI and OIA II in two different groups of patients with secondary peritonitis," *Hepato-Gastroenterology*, vol. 48, no. 37, pp. 147-151, 2001.

[7] S. Biondo, E. Ramos, M. Deiros, et al. "Prognostic factors for mortality in left colonic peritonitis: a new scoring system," *J. Am. Coll. Surg.*, vol. 191, no. 6. pp. 635-642, 2000.

[8] J.C. Marshall, D.J. Cook, N.V. Christou, et al. "Multiple Organ Dysfunction Score: A reliable descriptor of a complex clinical outcome," *Crit. Care Med.*, vol. 23, pp. 16381652, 1995.

[9] J.R. Le Gall, S. Lemeshow, and F. Saulnier, "A new Simplified Acute Physiology Score (SAPS II) based on a European/North American multicenter study," *JAMA.*, vol. 270, pp. 29572963, 1993.

[10] J. Truett, J. Cornfield, W. Kannel, "A multivariate analysis of the risk of coronary heart disease in Framingham,", *Journal of chronic diseases*, vol. 20, no. 7, pp. 511524, 1967.

[11] F.E. Harrell, Regression Modeling Strategies, Springer-Verlag, ISBN 0-387-95232-2, 2001.

[12] M. Strano, B.M. Colosimo "Logistic regression analysis for experimental determination of forming limit diagrams," *International Journal of Machine Tools and Manufacture*, vol. 46, no. 6, pp. 673682, 2006.

[13] S.K. Palei, S.K. Das, "Logistic regression model for prediction of roof fall risks in bord and pillar workings in coal mines: An approach," *Safety Science*, vol. 47, pp. 8896, 2009.

[14] M.J.A. Berry, "Data Mining Techniques For Marketing, Sales and Customer Support," Wiley, pp 10, 1997.

[15] J. Lafferty, A. McCallum, and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," *Proc. 18th Int. Conf. on Machine Learning. Morgan Kaufmann*, pp. 282289, 2001.

[16] X. He, and R.S. Zemel, and M.A. Carreira-Perpinn, "Multiscale conditional random fields for image labeling," *IEEE Computer Society*, 2004.

[17] K.Y. Chang, T.p. Lin, L.Y. Shih, and C.K. Wang, "Analysis and Prediction of the Critical Regions of Antimicrobial Peptides Based on Conditional Random Fields," *PLoS ONE*, 2015.

[18] T. G. Dietterich, "Ensemble Methods in Machine Learning", *Proc. of the* 1^{st} *Int. Workshop on Multiple Classifier Systems*, pp. 1-15, 2000.

[19] Kaggle, https://www.kaggle.com/.

[20] H. T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir, R. Anil, Z. Haque, L. Hong , V. Jain, X. Liu and H. Shah, "Wide & Deep Learning for Recommender Systems", *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, DLRS, pp. 1-10, 2016

[21] Q. H. Vu, D. Ruta, L. Cen. "An ensemble model with hierarchical decomposition and aggregation for highly scalable and robust classification", *Proceedings of the AAIA*, 2017