NANYANG TECHNOLOGICAL UNIVERSITY

March Machine Learning Mania 2016
Predict the 2016 NCAA Basketball Tournament

Submitted in Fulfilment of the Coursework Requirements
for the CE/CZ 4041 Machine Learning
by

Joe Tan Chin Yong	U1521434C
Liu Zeyan	U1421784C
Wei Yumou	U1320554F
Xie Dai	U1340229K

School of Computer Science and Engineering

Contents

Chapter 1

Introduction

The National Collegiate Athletic Association (NCAA) Division I Men's Basketball Tournament is one of the most popular annual sport festivities in the United States. Every year, the Tournament attracts a sizeable pool of audience, with the national champion becoming one of the hottest topics throughout the whole year. In the meanwhile, interests in predicting the winning team of a particular tournament match are escalating [6] and online machine-learning communities like Kaggle are organising annual competitions to encourage creative solutions. The purpose of this project is to present a feasible machine learning-based solution to Kaggle's competition "March Machine Learning Mania 2016" [4] that accurately predicts a team's *probability* of winning a particular tournament match based on historical match data provided by Kaggle.

1.1 Background

According to Wikipedia [8], the Tournament is played during every March and April based on the rule of single-elimination. Out of the 68 participating college basketball teams, 32 *conference* match champions are automatically qualified for the Tournament, while the other 36 teams are admitted at the discretion of a NCAA selection committee based on a criterion known as Rating Percentage Index [9].

The total 68 teams are then ranked by the selection committee from 1 to 68, and distributed amongst the four regions nominally known as East, West, South and Midwest. The top four teams receiving a rank from 1 to 4 are distributed to and given a *seed* of 1 in each of the four regions, followed by the next four teams with a rank of 5-8 that receive a seed of 2 in each region. The process continues until only the last eight teams are left whereby they have to fight with one of the other teams for the 16^{th} seed position for each region, which marks the commencement of

the Tournament and is officially known as the *First Four* round.

At this point, the number of contesting teams are reduced to 64. During the next *First Round* in each region, a team with a higher seed position plays against a team with a lower seed position. For example, there are matchups between teams with the 1^{st} seed and teams with the 16^{th} seed, between the 2^{nd} seed teams and the 15^{th} seed teams and so on. Subsequently, the 32 winning teams advance to the *Second Round* where they play against one of the other teams, after which the 16 remaining teams are known as the *Sweet Sixteen*.

The number of contesting teams continues to halve until the four regional champions are determined. During the *National Semi-final*, the regional champion with the 1^{st} seed position plays against the regional champion with the 4^{th} seed position, while the other two teams play against each other. *National Final* is the last round and conducted between the two winners of the National Semi-final to determine the National Champion. However, there is no consolation game for the third place in the Tournament.

1.2 Problem & Evaluation Method

The problem of this Kaggle competition is to predict a team's probability of winning a particular tournament match. In a real tournament, there should be 68 teams and 67 matches in total because of the rule of single elimination (only the champion is not *eliminated* after the 67 matches). But how teams are paired up in these matches are not known in advance, except for the first 32 matches whose contesting pairs can be derived from the seed results. Therefore, Kaggle requires participants submit predictions for all possible matchups between any two of the 68 teams, which amount to $68 \times (68-1)/2 = 2278$ matches according to the Handshaking lemma.

According to Kaggle [4], the method for evaluating predictive models and compiling the leaderboard is based on cross entropy or log loss:

$$LogLoss = -\frac{1}{n}\sum_{i=1}^{n}[y_i \ln \hat{y}_i + (1-y_i)\ln(1-\hat{y}_i)] \tag{1.1}$$

where

- n is the number of games played
- $\hat{y}_i$ is the predicted probability of team 1 beating team 2
- y_i is 1 if team 1 wins, 0 if team 2 wins

The goal of any predictive models is to *minimise* the log loss. Based on the LogLoss evaluation method, a completely incorrect prediction will lead to a score of ∞, for example, a prediction of 1 when the actual outcome is 0. To avoid such an unpleasant score, Kaggle uses a threshold function to scale the submitted probability into a reasonable range as follows:

$$p' = \max(\min(p, 1 - 10^{-15}), 10^{-15}) \tag{1.2}$$

where p is the submitted probability and p' is the adjusted probability.

When using the LogLoss function to evaluate submissions, Kaggle assumes the ground truth variable y_i is discrete, namely $y_i \in \{0, 1\}$. But in general, y_i should be a continuous variable representing the *true* probability that Team 1 will beat Team 2, and $\hat{y}_i$ still denotes the *predicted* probability. In this case, the cross entropy measures how close these two probabilities are. If there exists a *perfect knowledge predictor* that *always* gives correct predictions, namely $\hat{y}_i = y_i$ always holds, then its prediction performance can be described in Figure 1-1. Moreover, if the true probability follows an uniform distribution $y_i \sim U(0, 1)$, then in the long run the perfect knowledge predictor will have an expected score of

$$-\int_0^1 [y_i \ln y_i + (1 - y_i) \ln(1 - y_i)] dy_i = 0.5 \tag{1.3}$$

which is graphically equivalent to the height of a rectangle that shares the same base of length 1 as the performance curve's.

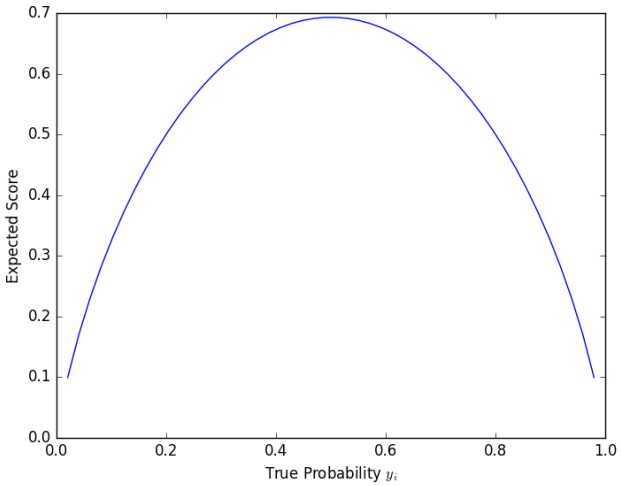


Figure 1-1: Performance of a Perfect Knowledge Predictor

1.3 Data Sets

Kaggle provides almost three decades' data about NCAA basketball matches, including matches during the regular seasons as well as the tournaments. The data sets are summarised in Table 1.1.

Data Set	**Description**
RegularSeasonCompactResults	Game-by-game results during regular seasons from 1985-2015
RegularSeasonDetailedResults	More detailed results during regular seasons from 2003-2015
Seasons	Different seasons present in the dataset
Teams	Different college teams present in the dataset
TourneyCompactResults	Game-by-game tournament results from 1985-2015
TourneyDetailedResults	More detailed tournament results from 2003-2015.
TourneySeeds	The seeds for all teams in a tournament
TourneySlots	The pair-ups between two teams based on their seeds

Table 1.1: A Summary of the Data Sets

However, not all data sets are useful. Only two of them are used in this project, namely **RegularSeasonCompactResults** and **Teams**.

1.4 Role Assignment

Table 1.2 shows the assignment of roles for each team member in this project.

Team member	**Role**	**Responsibility**
Joe Tan Chin Yong	Data Scientist	To build predictive models
Liu Zeyan	Data Analyst	To optimise selected models
Wei Yumou	Data Scientist	To build predictive models
Xie Dai	Statistician	To provide mathematical knowledge

Table 1.2: Role Assignment

Chapter 2

Preliminary Analysis

2.1 Challenges

In the attempt to tackle the problem, various challenges arise that are worth mentioning.

2.1.1 Feature Selection

Features are pieces of information that may be useful for predictions. It is generous of Kaggle to provide a comprehensive collection of data. However, not all data is relevant to making predictions. Before a set of features can be selected, careful choices must be made on which data sets to use and which portion of that data set is relevant to solving the problem.

Regardless of the machine-learning technique used, the most obvious choice of data sets to use is the historical match records for both regular and tournament seasons. However, some data sets contain match records that date back to as early as 1985, which are no longer relevant in today's context. After all, the NCAA tournament teams consist of *college students* who can only stay with a team for a maximum of four years before graduation. As players constantly leave and join the team, it is difficult to quantify the effect brought by changes in a team's composition on the strength of that team, given only the team-level data. Moreover, there were some substantial updates on the Tournament's rules at the beginning of the 2008-2009 season [7], which also affected the strategies teams used in the subsequent tournaments. Therefore, only the most recent match records are useful for prediction. For the purpose of this project, a four-year window from 2013 to 2016 is selected and all data used in this project fall within this window. In addition, only the match records from the regular seasons are actually used, since there are only a few records from the tournament seasons which add little value to making predictions.

There are a number of potential features that can be selected from the data set **RegularSeasonDetailedResult**. The potential features are further categorised as basic and additional features. Table 2.1 gives a summary of the basic features.

Feature	**Description**
Wteam	The id number of the team that won the game
Wscore	The number of points scored by the winning team
Lteam	The id number of the team that lost the game
Lscore	The number of points scored by the losing team
Wloc	The location of the winning team
Numot	The number of overtime periods in the game

Table 2.1: A Summary of the Basic Features

The additional features describe both *offensive* and *defensive* strengths of both winning team and losing team. Table 2.2 gives a summary of features describing offensive strengths, while Table 2.3 focuses on defensive strengths. Although the two tables list features from the winning team's perspective, another duplicated set of features also exists for the losing team.

Feature	**Description**
Wfgm	The number of field goals made by the winning team
Wfgm3	The number of three pointers made by the winning team
Wftm	The number of free throws made by the winning team
Wor	The number of offensive rebounds by the winning team
Wast	The number of assists by the winning team
Wstl	The number of steals by the winning team
Wblk	The number of blocks by the winning team

Table 2.2: A Summary of the Offensive Features

Feature	**Description**
Wfga	The number of field goals attempted by the winning team
Wfga3	The number of three pointers attempted by the winning team
Wfta	The number of free throws attempted by the winning team
Wdr	The number of defensive rebounds by the winning team

Table 2.3: A Summary of the Defensive Features

The potentially many features make model selection challenging, because they create an exponential number of possible models. If a feature has m possible values, to select the best models based on n features, at least $\Theta(n^m)$ models have to be examined, which is impractical given that only one submission is allowed at one time. Therefore, this project uses a greedy strategy where all other features are fixed at their optimal values when one feature varies to select a suboptimal model.

2.1.2 Interrelationship

No team is in isolation. The tournament matches are interactive processes whereby complex interrelationships exist amongst all the contesting teams, which adds another layer of complication in the attempt to predict match results. For example, given the history match records that Team A beaten Team B and Team C lost to Team B, one should intuitively conclude that Team A should have a higher probability of beating Team C. But what if another record shows that Team A once lost to Team C? In that case, the relative strength levels of the three teams will be hard to determine based purely on that intuition. Moreover, the match whose result is to be predicted may be the very *first* match ever between two teams. In other words, there are no historical records that give a direct assessment on the two teams' relative strengths. Such lack of knowledge must be complemented by some forms of inference based on the interrelationships amongst the teams. So a good predictive model should not only be able to take into consideration the current game record, but also explore the interrelationships amongst all the game records and generalise on unseen matches.

2.1.3 The Curse of Model Popularity

The problem to solve can be categorised as a classification problem under supervised learning. But since the final answers to be submitted are in fact *probabilities*, non-probabilistic models like support vector machines will hardly work. Therefore, the first a few models attempted in this project include the most popular ones: logistic regression and multilayer perceptrons.

The inputs to these popular models are the seed positions of each team, since the seeds represent an official view on the relative strength of each team. However, the performances of these models are not as good as expected: the logistic model only gives a 283$^{\text{rd}}$ position on the leaderboard. Moreover, only teams in a tournament are assigned a seed but the number of tournament matches is not large enough to support accurate predictions. Although [5] suggests a combination of logistic regression and Markov chain to predict match outcomes, these popular models alone based on simply seeds are unlikely to give a good result.

2.2 Approach

In light of the challenges identified, the general approach is to quantitatively describe the *skill* of each team, from which the winning probabilities are derived. This section gives a rough idea about the approach used in this project, while the detailed mathematical principles are left to Chapter 3.

The system used to estimate the skill of a team is known as a *rating system*. An example of a rating system is the famous Elo rating system [2] widely applied in board games like chess and Go. In a Gaussian rating system, each team i is assumed to have a skill s_i that follows a Gaussian distribution $s_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$, where μ_i is the mean skill of a team and σ_i can be considered as the system's *belief* about the team's skill level.

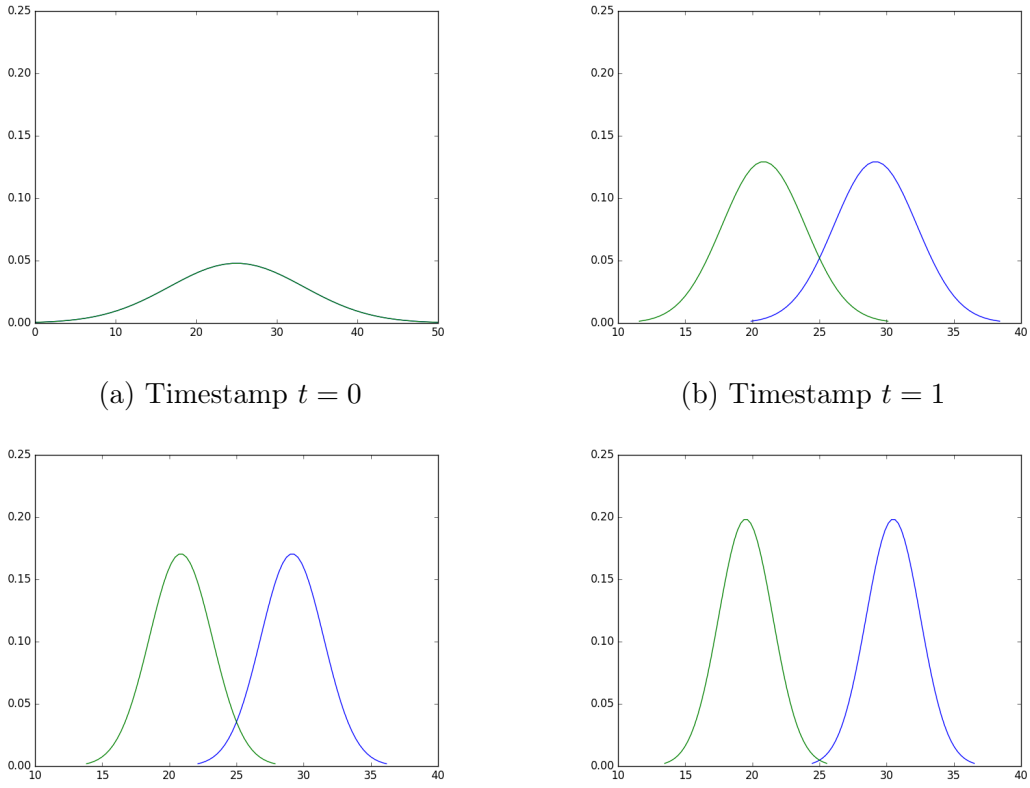
Depending on the rating system, the skill of a team is updated after a single match or a series of matches within a specific period. The amount of such a update depends on how *surprising* a match outcome is — if, prior to a match, Team A is expected to have a very high probability of beating Team B based on their *skill gap*, an outcome that Team A actually wins the game will *not* result in a significant update on either Team A's or Team B's skill; however, on the other hand, if it turns out that Team B performs superbly and beats Team A at the end of the match despite its lower skill level, which is known as an "upset" in sport term, there *will* be a significant increase in Team B's skill and a significant decrease in Team A's skill. But in either case, the belief of skills σ_i will shrink because the rating system becomes more confident about the two teams' skill level.

As an example, suppose two teams have the same initial skill estimates of s_i and s_j respectively as shown in Figure 2-1a. As the tournament begins, s_i and s_j are adjusted according to the outcomes of the matches, which is represented by the shift of skill curves as shown in Figure 2-1b and 2-1c. In the meanwhile, the skill curves become increasingly *taller*, indicating that the rating system is increasingly confident about their skill levels. Finally when the tournament ends, the rating system reports the final standings of the two teams as shown in Figure 2-1d.

A team's probability of beating another team in an *upcoming* match can be estimated from the two teams' skill difference.

$$P(s_i > s_j) = \Phi\left(\frac{\mu_i - \mu_j}{\sqrt{\sigma_i^2 + \sigma_j^2}}\right) \tag{2.1}$$

where Φ is the cumulative distribution function of a standard Gaussian distribution.



(a) Timestamp $t = 0$

(b) Timestamp $t = 1$

(c) Timestamp $t = 2$

(d) Timestamp $t = 3$

Figure 2-1: An Example of Skill Update

Chapter 3

TrueSkill™ Rating System

TrueSkill [3] is a Gaussian rating system developed by Microsoft Research in 2007 to rank players on Xbox Live for the purpose of creating competitive matches for players with similar skill levels. While the Elo rating system is applicable to only two-player matches, the TrueSkill ranking system extends the use cases to multi-player matches. Section 3.1 discusses the Approximate Bayesian Inference — the mathematical backbone of the TrueSkill rating system. Section 3.2 presents the process of building predictive models based on an open-source implementation of the TrueSkill rating system.

3.1 Approximate Bayesian Inference

The approach described in Section 2.2 can be modelled as a Bayesian Inference process. Define a vector $\boldsymbol{s} = [s_1, s_2, \ldots, s_n]^\intercal$ representing the individual *skill* of n teams in a match (in the case of NCAA basketball tournament, $n = 2$). Further assume that each skill follows a Gaussian distribution $s_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$ and collectively, the *prior* joint distribution of skills is a n-variate Gaussian distribution $p(\boldsymbol{s}) = \mathcal{N}_n(\boldsymbol{s}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$

In a match, each team is expected to exhibit a certain *performance* $p_i \sim \mathcal{N}(s_i, \beta_i^2)$ that varies around its skill. The match result can be represented as a set of rankings in ascending order for each team, namely, $\boldsymbol{r} = [r_1, r_2, \ldots, r_n]^\intercal$ where $r_1 \leq r_2, \ldots, r_{n-1} \leq r_n$. In reality, there are many factors that may affect the outcome of a match, but for the purpose of Bayesian Inference, the assumption is that it is the differences in team performances that cause the differences in team rankings. According to the Bayes' Theorem, the posterior distribution $p(\boldsymbol{s}|\boldsymbol{r})$ is given by

$$p(\boldsymbol{s}|\boldsymbol{r}) = \frac{P(\boldsymbol{r}|\boldsymbol{s})p(\boldsymbol{s})}{P(\boldsymbol{r})} = \frac{P(\boldsymbol{r}|\boldsymbol{s})p(\boldsymbol{s})}{\int P(\boldsymbol{r}|\boldsymbol{s})p(\boldsymbol{s})d\boldsymbol{s}} \tag{3.1}$$

The above formula demonstrates the use of Bayes' Theorem to calculate the updated skill distribution (posterior) given the match outcome (evidence) and the expectations (likelihood and prior) on the match outcome before the match. However, the posterior distribution is no longer a Gaussian distribution and oftentimes, the integral is intractable. A common solution is to calculate a Gaussian distribution $\mathcal{N}_n(\boldsymbol{s}; \hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\Sigma}})$ as an approximation to the posterior distribution $p(\boldsymbol{s}|\boldsymbol{r})$ so that the Kullback-Leibler divergence between the two distributions is minimised.

A number of approximation techniques are available. For the TrueSkill rating system, it approximates the posterior distribution by setting up a *factor graph* which is a bipartite graph containing two kinds of vertices: variables and factors. A variable stores some value of interests and a factor represents some operations on one or more variables. The Bayesian Inference process is modelled as passing some *message*s, which are some real-valued functions of a variable or a factor, throughout the factor graph. Figure 3-1 shows an example factor graph of a simple two-player match. Based on skill s_i and performance p_i, it calculates the expected performance difference $p_1 - p_2$ and compares that with an externally supplied outcome d. The $\mathbb{I}$ denotes an indicator function to assert whether d is consistent with the expectation, depending on which a different update is performed on s_1 and s_2.

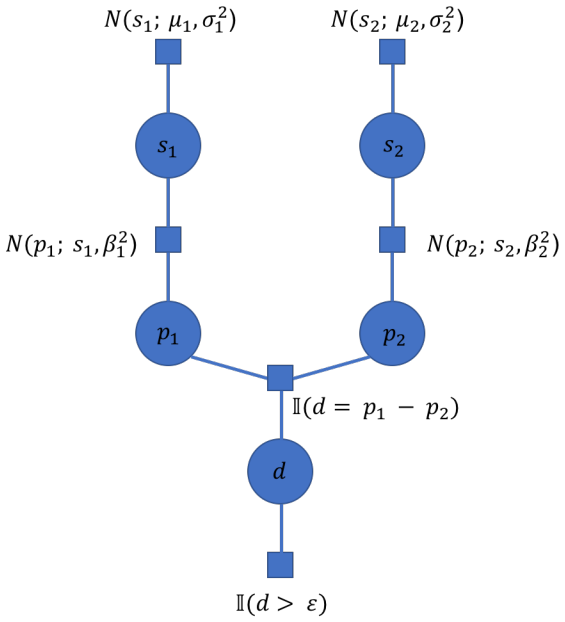


Figure 3-1: An example of factor graph

The ultimate goal of a factor graph is to obtain a set of update equations that indicate how each μ_i and σ_i should be updated. However, passing messages throughout larger factor graphs can involve very intensive computations. As suggested by [1], there is an alternative but easier way of getting the same set of update equations based the following theorem.

Theorem 1. Let $\boldsymbol{z}$ be a k-dimensional random vector $[z_1, z_2, \ldots, z_k]^\intercal$ with a probability distribution function of the form

$$\frac{\phi_k(\boldsymbol{z})f(\boldsymbol{z})}{\int \phi_k(\boldsymbol{z})f(\boldsymbol{z})d\boldsymbol{z}} \tag{3.2}$$

where ϕ_k is a k-variate standard Gaussian distribution function. Then, the expectation of $\boldsymbol{z}$ is given by

$$E[\boldsymbol{z}] = E\Big[\frac{\nabla f(\boldsymbol{z})}{f(\boldsymbol{z})}\Big] \tag{3.3}$$

and also

$$E[z_i z_j] = \delta_{ij} + E\Big[\frac{\nabla^2 f(\boldsymbol{z})}{f(\boldsymbol{z})}\Big] \tag{3.4}$$

where $\delta_{ij} = 1$ if $i = j$ and $\delta_{ij} = 0$ otherwise.

To link Equation 3.2 with the Bayes' Theorem in Equation 3.1, notice that the match outcome is dependent on performances and performances are related to skills; therefore, the likelihood probability $P(\boldsymbol{r}|\boldsymbol{s})$ can be expressed as some function of skill $f : \boldsymbol{s} \to [0, 1]$. Furthermore, since each $s_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$, define the z-score vector $\boldsymbol{z} = [z_1, z_2, \ldots, z_n]^\intercal$ where

$$z_i = \frac{s_i - \mu_i}{\sigma_i} \tag{3.5}$$

Therefore, the probability distribution $p(\boldsymbol{z}) = \phi_n(\boldsymbol{z})$ and the likelihood probability $P(\boldsymbol{r}|\boldsymbol{s})$ is also a function of $\boldsymbol{z}$. Then according to Equation 3.1,

$$p(\boldsymbol{z}|\boldsymbol{r}) = \frac{P(\boldsymbol{r}|\boldsymbol{z})p(\boldsymbol{z})}{P(\boldsymbol{r})} = \frac{\phi_n(\boldsymbol{z})f(\boldsymbol{z})}{\int \phi_n(\boldsymbol{z})f(\boldsymbol{z})d\boldsymbol{z}} \tag{3.6}$$

and therefore by Theorem 1, the values of interest μ_i^{new} and σ_i^{new} are related to $E[z_i]$ through

$$\begin{aligned} \mu_i^{new} &= E[s_i] = E[\mu_i + \sigma_i z_i] = \mu_i + \sigma_i E[z_i] \\ &= \mu_i + \sigma_i E\Big[\frac{\partial f(\boldsymbol{z})/\partial z_i}{f(\boldsymbol{z})}\Big] \end{aligned} \tag{3.7}$$

and

$$\begin{aligned}\sigma_i^{new} &= Var[s_i] = \sigma_i^2 Var[z_i] = \sigma_i^2(E[z_i^2] - E[z_i]^2) \\ &= \sigma_i^2\Big(1 + E\Big[\frac{\nabla^2 f(\boldsymbol{z})}{f(\boldsymbol{z})}\Big]_{ii} - E\Big[\frac{\partial f(\boldsymbol{z})/\partial z_i}{f(\boldsymbol{z})}\Big]^2\Big)\end{aligned} \tag{3.8}$$

3.2 Implementation

Chapter 4

Time-Dependent Edge Cost Estimation

Bibliography

[1] Ruby C.Weng and Chih-Jen Lin, *A bayesian approximation method for online ranking*, Journal of Machine Learning Research **12** (2011), 267–300.

[2] Arpad Emmerich Elo, *The rating of chess players, past and present*, Arco Publishing, New York, 1978.

[3] Ralf Herbrich, Tom Minka, and Thore Graepel, *Trueskill(tm): A bayesian skill rating system*, MIT Press, January 2007, pp. 569–576.

[4] Kaggle Inc., *March machine learning mania 2016*, https://www.kaggle.com/c/march-machine-learning-mania-2016, March 2016.

[5] Paul Kvam and Joel S. Sokol, *A logistic regression/markov chain model for ncaa basketball*, Naval Research Logistics **53** (2006).

[6] Net Prophet, *Exploring algorithms for predicting ncaa basketball games*, http://netprophetblog.blogspot.sg/, April 2017.

[7] Net Prophet, *Five mistakes kaggle competitors should avoid*, http://netprophetblog.blogspot.sg/2015/02/five-small-thoughts-for-kaggle.html, February 2015.

[8] Wikipedia, *Ncaa division I men's basketball tournament*, https://en.wikipedia.org/wiki/NCAA_Division_I_Men%27s_Basketball_Tournament, April 2017.

[9] Wikipedia, *Rating percentage index*, https://en.wikipedia.org/wiki/Rating_Percentage_Index, March 2017.