Linear Models to Predict the 2017 Miami Marathon Results

Selin Jessa — 260569186 — selin.jessa@mail.mcgill.ca
Yue Wang — 260719779 — yue.wang8@mail.mcgill.ca
Nirmal Kanagasabai— 260716737 — nirmal.kanagasabai@mail.mcgill.ca

Abstract

Here we present two linear models for predicting participating in the 2017 Miami Marathon, and a linear regression for predicting participants' completion times in 2017, towards the completion of Project 1 in COMP 551 - Applied Machine Learning.

1. Introduction

The Miami Marathon occurs annually in the early Spring, and we are tasked with using linear models trained on data from 2003-2016 to predict whether all participants from these years will participate in 2017 (Task Y1), and what their marathon time will be (Task Y2). We employ two linear models, logistic regression (LRC) and a Naive Bayes Classifier (NBC) to complete Task Y1, and use linear regression (LR) to complete Task Y2.

2. Problem Representation

2.1. Preparation of Training Data

In order to inform our selection of relevant features, we cleaned the raw dataset by applying several filters. These include filtering out records with ID not associated with a name and reshaping the data into a wide format to in order to collect data for each unique participant across the years 2003-2016 - these constitute our examples or observations. As shown in Figure 1, we observed that the data for the year 2013 corresponded to half-marathon times and we used this to inform our later analysis. In addition, we observed that in some instances, two participants with the same name participated in the same year, and we manually filtered out one example in each pair either according to any data for that participant from previous years, or by discarding one randomly. Given this cleaned data, we then proceeded to select or recode individual features based on the task. These data-cleaning procedures are executed by the file clean_data.R.

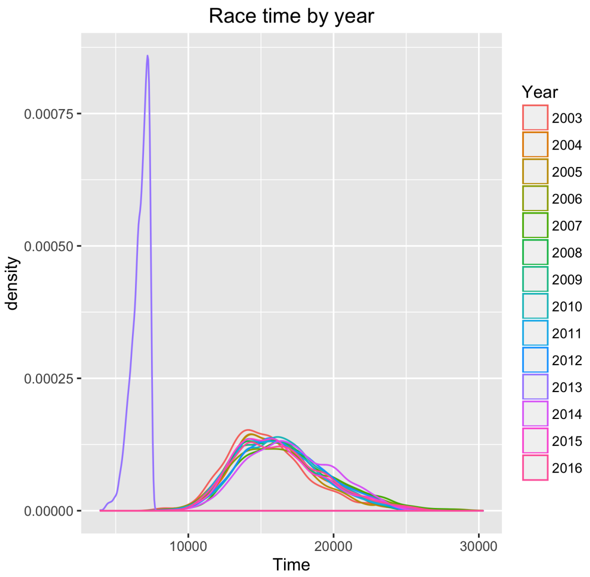

Figure 1. Density plot of participants' race times

2.2. Feature Selection for Task Y1: Predicting Participation

To predict whether a runner would participate the following year, we consider the following features, some of which were been recoded based on the cleaned data. We describe each feature and justify its use, as appropriate.

1. **Sex** (categorical): We encode male as 1 and female as 0.

2. **Current age** (continuous): The current age as computed from the age associated with the most recent record for each participant.

3. **Current age** (categorical): Given that it appeared different methods of recording age were used over the years, we also computed a binned age, where the most recent age to nearest 10 was computed.

4. **Number of races previously run** (continuous): In-

Project 1: COMP 551 (Applied Machine learning), January 2017, Jessa, Wang, Kanagasabai.

tuitively, we consider the participation history of each runner, and hypothesize that a person who participated many times before would be more likely to participate again.

5. **Years since last race** (continuous): Another important feature of the history of a participant, which allows us to consider when the data is outdated. We hypothesize that a runner who has not participated recently is less likely to participate in the next race.

6. **Mean rank of participation record** (continuous): The performance of participation could also play an important role in future participation. Consider a marathoner who has great accomplishment in his record, he should have a tendency of participate again. We identify these athletes by their average rank of across records and mean pace.

For this task, the LRC model used all the features described above, while the NBC model used only the binned current age, the number of races previously run, and the number of years since the last race. In addition, in the LRC, rank and age were normalized using feature rescaling by the following formula: $x_i = (x_i - x_{min})/(x_{max} - x_{min})$.

2.3. Feature Selection for Task Y2: Predicting Time

1. **Age** (continuous)

2. **Sex** (binary, recoded as above)

3. **AvgTime[Year]** (continuous): Completion time in seconds for entire marathon

4. **AvgTimeForAllMarathons** (continuous): Average completion time in seconds across all years with data for given participant

5. **TotalNoOfRaces** (continuous): Equivalent to **Number of races previously run** above

In Linear regression, as the main goal is to predict the finishing time (the time the participant takes to complete the marathon), the focus is more on the average time that the user took to finish the race across all years (2003 to 2016). Initially, data widening was carried out on the raw dataset so that there is a separate feature-set for Age, Pace, Time across all years (2003 to 2016). The feature set was huge and a lot of them didnt contribute to the final prediction. Hence, we decided to reshape the data which could provide favourable and expected results. The features which didnt impact the final predictions were scraped. Few new features were derived to help our cause.

The data manipulation was carried out in R Studio and a small portion of them (manually). Considering the fact that the Marathon has been spread across a large span of time, it can be understood that the users might not have participated in all the years. Hence, we decided to included the total number of races (TotalNoOfRaces) feature into our dataset. The table below describes what features were chosen for carrying out linear regression.

We also looked up on the dates when Miami marathon was organized over the years. The source was “Athlinks and with that,” we searched for the weather history. We captured data for temperature (C), Precipitation (mm), Wind Speed (Km/h), Wind Direction, Visibility (Km) and Sea Level Pressure (hPa). However, we could observe that there was hardly any variation with respect to the temperature, sea level pressure and visibility. There were notable variations in rainfall and wind speed. However, we had to eliminate the choice of wind speed because we werent sure what direction the wind was blowing with respect to the race track (in favour of / against) the player. Including the rainfall didnt impact much and hence, we decided to drop that feature too!

3. Training the Models

We set aside ten percent of the training data, sampled randomly, as a final test set, and completed training and validation on the 90% of the data reserved for training. Here we describe the testing of the data, and below we report the results on the final held-out test set.

3.1. Training for Task Y1 Using Logistic Regression

3.1.1. Data Normalization

In order to better use ridge regularization to penalize our models complexity, we normalized the features. Ridge regression puts constraints on the size of the weights. However, this value will depend on the magnitude of each weight variable. Therefore, it is therefore necessary to normalize and standardize the variables. We normalize different features in different ways to better preserve their context, as well as distributions.

3.1.2. Add Different Weights to Imbalanced Classes

In the dataset, the number of samples with y1= 0 is ten times more than the number of samples with y1 = 1. Consider the imbalance in the dataset, we add weights to address the minority class. We called compute_class_weight() from scikit-learn to get the weights. The weights are used in updating the weights of features. Once a false negative error happens, a large penalty will be added to the weight update part.

3.1.3. Training Data

We use the record from 2003 to 2015 data as training data to predict 2016 participation. If we deal with time series data, the information would be learned 100% by the model. However, in order to predict the result of 2017, we need to evaluate the weights of every years record. It would be difficult to learn the weights of 2016 based in the data available. Therefore, we avoid this problem by only considering statistic features, and create new features to reflect as much information as possible from data available. For participation classification, we focus on the participation records of these years. More feature selection is introduced in feature selection.

3.1.4. Stratified K Fold Cross Validation

We assume that our dataset represents a random sample drawn from a probability distribution. We think further subsampling into training set and validation set without replacement would have influence on the statistics of the sample. Therefore, we randomly split the dataset so that each class is correctly represented in training and the test set. We implemented this by Stratification option in python. Since lower K usually means cheaper and more biased, due to time and computational budget we chose k to be 10 and in order to reduce variance we simulated for 4 times with same K and different random folders, and finally averaged the results.

3.2. Training for Task Y1 Using Naive Bayes

To train the Naive Bayes Classifier, we first employed an assumption of normality, that is, that the three continuous variables used could be modelled with independent Gaussian distributions, as per the Naive Bayes Assumption. To estimate the class priors, we used the frequency of the positive class and added a Laplace smoothing term to handle the case of where a positive example is not seen in the training set (*i.e.* the set contained no positive-labelled examples), due to class imbalance in the dataset. For the class-conditional distributions for each feature, we used the maximum likelihood estimates for mean and variance, and averaged the parameters across 20 folds of cross-validation to obtain the final parameters. Figure 3.2 shows the accuracy on each set during cross-validation.

3.3. Training for Task Y2

The methodology to perform Linear Regression was different from Logistic Regression and Naive Bayes. Also, the focus was to predict the final finishing time. Hence, there was a need to re-shape the data to suit the needs. As it has been explained in the previous section, data analysis was carried out to effectively include or eliminate datasets. For tackling the missing data issue in AvgTime feature sets, we initially experimented with data imputation through MICE package in R. However, we wanted a simplified and an easier approach. Hence, we decided to subsitute the missing data by the persons mean (average) time across all years. This yielded better results. For the final prediction, we used the average time for 14 years which was elaborate and vast. It also included generic features (no. of races, age and gender). The time records in the data set had to be normalized and was set as a unit of seconds (for the entire marathon 42 kms).

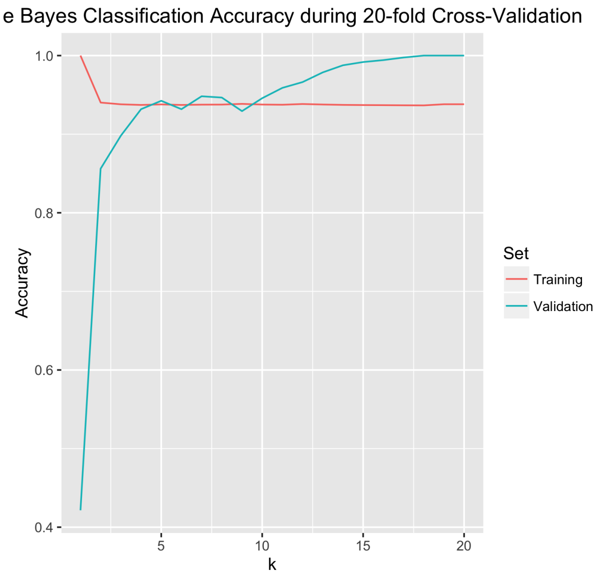

Figure 2. Training and validation set accuracy of Naive Bayes Classifier during 20-fold cross-validation.

We decided not to implement regularization for Linear Regression as the number of features were relatively less compared to the training data sets. The parameter alpha was set to 0.07 and reg to 0.02. We decided to iterate the Gradient Descent function 5000 times and have plotted the data for cost vs. number of iterations. In order to split the data, we employed k-fold cross validation method to split the provided data into training and test data and then compute the Mean Squared Error.

4. Results

4.1. Results of Predicting Participation

Because gradient descent could have multiple optimal solutions, we randomized the initial weights, and simulated for multiple times.

Table 1 shows the optimal weights we applied for predict the 2017 marathon. Here we could see obviously the number of races gives a considerable high contribution to the

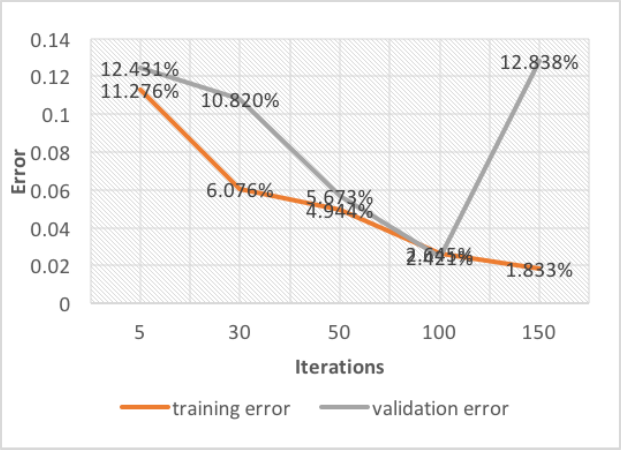

Figure 3. Error curve estimating the generalization performance of the logistic regression model.

prediction, whereas years since last participate have negative contribution to the decision of participation. This is reasonable since a person would have less chance to participate if he has longer time since his last participate, it is highly possible that he hasnt been practice for a long time, and vice versa. We can also notice that the sex feature gives negative contribution, which means that there might be a tendency that women are more willing to participate than men.

	Feature	Weight
1	bias term	-2.37
2	sex	-1.31
3	current age	0.40
4	num of races	4.98
5	years since	-9.27
6	mean pace	0.28
7	mean rank	-1.64

Table 1. Optimal weights for features in logistic regression model, obtained by gradient descent

From Figure 4.1 we want to estimate the generalization performance, the predictive performance of our model on future (unseen) data, as we could see when iterated for 100 times the model has obviously over-fitted. We finally use 50 iterations to learn all the training data and get our optimal weights to predict 2017.

The Naive Bayes model performed with 93.5% accuracy on the final test set. Below, we show the ROC curve for the model. Our model used a decision boundary on the log-odds ratio equal to 0.

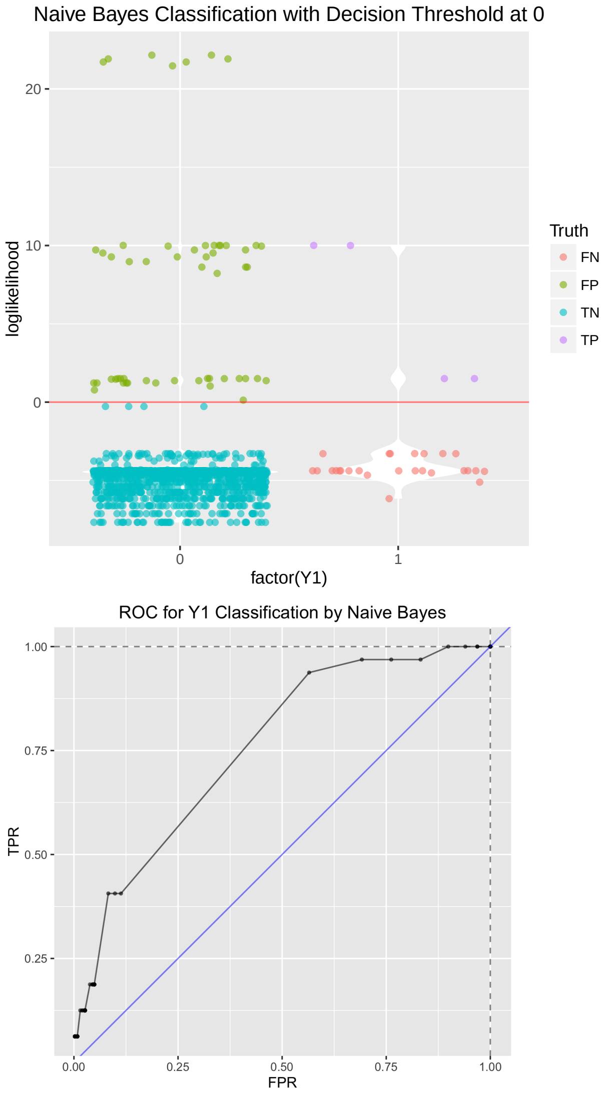

Figure 4. Performance of Naive Bayes Classifier for various thresholds in decision rule. **Top:** Results for threshold of 0 (*i.e.* log-odds ratio > 0). **Bottom:** ROC for Naive Bayes Classifier.

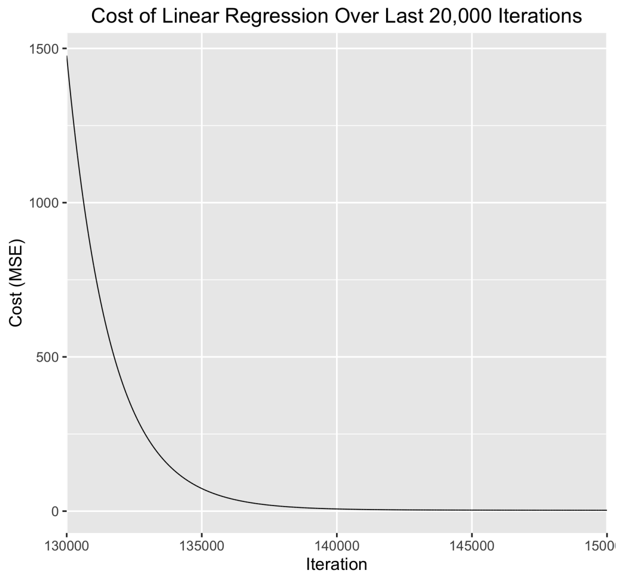

Figure 5. Change in cost function (MSE) over 150,000 iterations. Cost shown over last 20,000 iterations.

4.2. Results of Predicting Time

The Mean Squared Error (MSE) is computed for both training and validation data sets. For training, the MSE was 4012.96 and for validation, the MSE was 4546.15184835. This cost function is plotted against the number of iterations in the below graph:

5. Discussion

5.1. Predicting Participation

In a nutshell, our model predicts 99.72% of the existing participants would not participate again. However, we do have some cases where people do participate, for example, a woman participated for 10 times and is in her 40s, since clearly our model gives highest weights to the number of races feature, our model predicted she would participate again. We think this is reasonable, consider the facts that she might be a housewife with healthy lifestyle.

We note several possible improvements for using logistic regression to predict marathon participation. First, we could include interaction terms and corresponding regularization: if we are going to consider the interacting features, we could consequently add ridge regularization to penalize the increased complexity of model.

One characteristic of the data provided for this study was extremely imbalanced classes. Currently, our implementation of logistic regression accounted for class imbalance by assigning different weights, which is equivalent to a cost-sensitive classifier, because misclassifying a data instance from the minority class as the majority class is much more expensive than the other kind of misclassification. In future work, we could over-sample the minority class using SMOTE- a technique well-characterized in the literature.

5.2. Predicting Time

In case of linear regression, we have employed predictive mean matching to overcome the missing data issue. However, Data Imputation would be a beter choice and hence, the prediction would be much more better without a lot of fluctuations. For future scope, such data imputation techniques could be employed. Also, more feature sets which impacts the finishing time could be analyzed and added.

6. Statement of Contributions

NK was responsible for implementation of Linear Regression (LR) and carrying out all related data manipulations, as well as feature selection for Task Y2, and programming the solutions. **YW** was responsible for the completion of Task Y1 by logistic regression including feature selection and design, implementation of logistic regression, and prediction by logistic regression. **SJ** completed the data cleaning and recoding of features, implemented the Naive Bayes Classifier and completed the Y1 task using the model, produced figures, and compiled all authors' work into the final report. All authors contributed to the writing of the report equally. We hereby state that all the work presented in this report is that of the authors.