Predicting the Winner!

Hanieh Baradaran Kashani
Student Id:260778773
hanieh.baradarankashani@mail.mcgill.ca

Inderjot Kaur Ratol
Student Id: 260661928
inderjot.ratol@mail.mcgill.ca

Zahra Khambaty
Student Id: 260577706
zahra.khambaty@mail.mcgill.ca

I. Introduction

Given the data of all the Miami full marathons happened between 2003-2016, we have to complete the following two prediction tasks:

- Participation in 2017- We use Logistic Regression optimized with gradient descent and Naive Bayes to find the participation probability of each participant ID present in the data.
- Finishing time per participant- We use Linear regression optimized with gradient descent

Having completed the above two tasks, the ultimate task is to predict the winner of 2017 Miami Marathon.

II. Problem Representation

The data file contained information about each participant's *Id, Name, Age, Sex, Rank, Finishing time, Pace and Year of participation*. These fields are represented as columns in the data file. Each participant has one or more than records, represented by a data row in the file, based on their participation in the previous years. The data file included a lot of spurious data which needed sanitization and normalization as a part of preprocessing step.

A. Pre-Processing of Data

Sanitization of data

- We checked the instances where age of the participant was less than 15 and removed it from the data set used for training and testing. This assumption about the age is supported by the rules of Miami Marathon [1] As per the rules, 16 is the minimum age for participation in a full marathon. Hence, we included the participants of age 15 as well because they will be eligible for the marathon in 2017.
- First, we sorted the year's column by grouping the participant's ID and conditioned the age such that if the current age is equal or lesser than the participant's age during the previous year, then increment the age by the difference between the two years in the current context. We also checked if the difference between the current age and previous age is equal to the difference in respective years and corrected the age accordingly.
- We converted the value in columns *Time and Pace* into seconds for easier calculations.
- Some instances had incorrect value in the Sex column, for example, the participant with ID- 1131 was categorized as 'M' in two records and 'U' in one. Hence, we categorized the person based on the category that appeared in the file with higher frequency than other categories. At the end, we had only two sex categories.
- There were a few participants who ran half-marathon in the year 2013. We modified their finishing time by multiplying their pace with standard marathon distance, i.e. 26.219 miles , to obtain the full marathon finishing times.
- We found that there were some cases where participant had the same ID, Name and Year of participation. We sorted the file based on year grouped by Id's and whenever we stumbled upon a case like mentioned above, we incremented the the last ID in file and assigned the ID to that inconsistent record.

Normalization of data

To normalize data, we used two different approaches which are as following:

- **Normalize numerical values**- We normalized the data using *Scikit learn*s normalize() method. The columns which are normalized using the above mentioned method are - Rank, Pace and Finishing Time.
- **Converting data into categorical distribution**- We converted the columns *Age and Year* into 13 and 5 categories respectively. The categories used for column *Age* are chosen based on the standard categories used in marathons [2] and year categories are as follows: 2003-2005 (1), 2006-2008(2),2009-2011(3), 2012-2014(4) and 2015-2016(5). The values in the brackets are the labels assigned to the categories, where recent years are given higher value.
- **Normalize the categorical data**- The column *Sex* is normalized using *labelEncoder* method provided by Scikit learn. The method automatically assigned labels -0,1,2 to U,M and F respectively.

The Table I shows the original and transformed data format of the columns in the data file.

B. Feature Selection

1) Aggregating records per participant: Before selecting the features, we need to have unique record per participant

[1] http://www.themiamimarathon.com/contact/faqs/

[2] http://www.baa.org/races/distance-medley/event-information/age-group-information.aspx

TABLE I. DATA COLUMNS AND CONVERTED FORMATS

Column Name	Original Format	Converted Format
Id	Integer	Integer
Name	String	Not used
Age	Integer	Categorical→ Normalized
Sex	String	Normalized
Rank	Integer	Integer
Pace	DateTime	Converted into seconds
Time	DateTime	Converted into seconds
Year	Integer	Categorical→Normalized

to correctly make predictions. For aggregating the data per participant, we used the following criteria:

- "Id" is used to identify the unique record of a participant.
- "Name" is not required as a feature since the "Id" column can be used to identify the participant.
- For "Age", we used the label with the maximum value. As mentioned earlier, we used labels for each age category, for example 13, 0 for the age category 80+ and [15-19] respectively.
- Rank is the average rank of a participant over all the years in which he/she has participated. Similarly, we have averaged Pace and Finishing Time columns.
- "Sex" value remains the same.
- We chose the maximum value of labels used for "Year".

2) Additional Features:

- **Interaction features**
  - Dot product of "Rank" and "Finishing time" - These columns exhibit an inverse relation i.e. Rank decreases with increase in finishing time. This can be shown graphically.
  - Dot product of running time and pace- These two columns are correlated and have a direct relation.
- **Additional Feature**- We introduced a new feature to strengthen the correlation between the number of years participated and the year category label. We multiply the number of years participated and range label obtained in aggregated set. Aggregated set refers to the set created by grouping the data per participant as explained above. This feature captures the relation between recent and previous participations and prefers recent ones over the previous ones.

3) Feature selection based on the model: Table II illustrates different features that are used in specific method. This feature selection is based on the results obtained after training the models and will be discussed at length in the section III.

III. TRAINING METHODS AND RESULTS

As we are using supervised learning methods, this requires us to have the predictions available for the data used to train the models. In order to fulfill the foremost requirement, we split our data into two sets according to the specific models.

TABLE II. FEATURE SELECTION PER METHOD- 1 IMPLIES USED, 0 IMPLIES NOT USED

Feature	Logistic Regression	Naive Bayes	Linear Regression
Id	0	0	0
Name	0	0	0
Age	1	1	1
Sex	1	1	1
Rank	0	0	0
Pace	0	1	1
Time	0	0	0
Additional Feature (Replacing Year)	1	1	0

A. Finalizing the input and output data

1) Logistic Regression and Naive Bayes: Input Matrix - extracted from the original data by excluding the records corresponding to year 2016. Candidates who only participated in 2016 are not considered in the training data set. *Output labels*- We create another vector of size m*1 where m is the number of participant in the input matrix which included predictions, in the form of binary values, representing participations in 2016.

2) Linear Regression: Input Matrix - extracted from the original data by excluding the records corresponding to year 2016 and including only those who has participated in the year 2016 so that we have a corresponding value of "Finishing Time" in the output labels to compare the performance of our model. *Output labels*- We create another vector of size m*1 where m is the number of participant in the Input matrix which contained the finishing times in 2016.

B. Division of Training and Testing data

We apply 3-fold cross validation to choose our training and testing models.

Analyzing Correlations between Features As in the correlation plot shown in Figure 1, we see that the two features, Rank and finishing time, have a high correlation but we dont remove the features now until we look at the VIF of the model to examine the multicollinearity of the model.

1) Logistic Regression with gradient descent-Using R:

a) First attempt: Table III shows the coefficients obtained on first training attempt with learning rate of 0.001 and 20 iterations. We keep checking the VIFs repeatedly to insure that there is no collinearity.

b) Second attempt: We measure the coefficient values for year category, year count and for the new introduced feature which is basically an interaction of year count and year category. We find that interaction term's coefficient is significant in our analysis and we keep the new feature by discarding the other two, i.e. year category and year count.

c) Third attempt: We focus on Rank and Time feature and their interaction which yields a very high value of VIF, hence we discard the Rank and Time.

We have tried seven models for logistic regression and captured the error rate for each model. The error rates where almost the same i.e. about 0.07. The value of error rate assured us that we are going in right direction and hence we finalized

TABLE III. COEFFICIENTS ON TRAINING ATTEMPT 1

Features	Intercept	Age	Rank	Time	Pace	New feature (year*category)	YearCount-before2016	Sex
Coefficients	-3.63990	-0.23382	-0.32684	0.0556	0.02903	1.45315	0.02170	0.18955

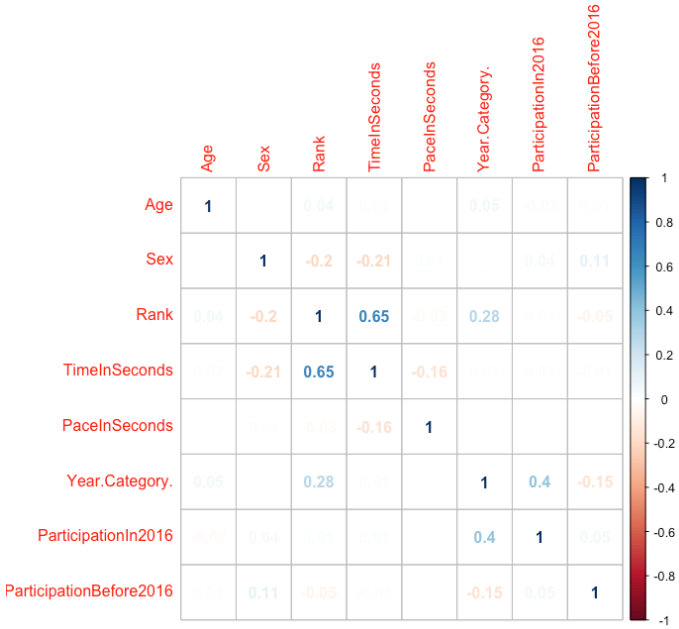

Fig. 1. Correlation of Features

our model. **Hyper Parameters**- We used 100 iterations of gradient descent and a value of 0.001 for the learning rate. The error was lowest at this point and becomes constant afterwards. Figure 2 shows the error rate versus number of iterations graph for logistic regression.

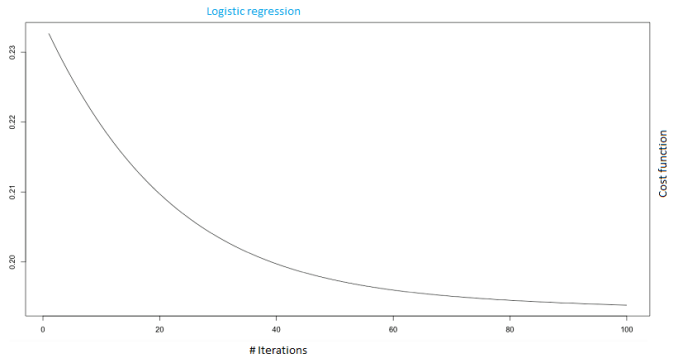

Fig. 2. Error vs number of iterations

As the predictions are the probabilities, we have taken more than 0.5 values as 1 and others as 0 whereas 1 indicates participation and 0 represents the non-participation.

2) Logistic regression with Python: As we had the features ready for logisitic regression, we tried implementing the same with Python. It did not work as expected and gave us an accuracy of 58.04%. We were not able to find the error due to shortage of time.

3) Naive Bayes: We implemented Gaussian Naive Bayes for predicting the participation because our normalized data fits into Gaussian distribution. We used the same features as logistic regression. Naive Bayes assumes an inherent independence between the features and we made sure by using the VIF values and correlations graph.

a) First attempt: We ran the algorithm using the same features and dividing the training and test data using 3-folds cross validation as mentioned above. It gave us an accuracy of 97.12%. But, it predicted all 0's.

b) Second attempt: Naive bayes does not handle unbalanced classes properly. If the data is dominated by one class, it introduces a bias and Naive Bayes does not handle that properly. *Solution* - We had two solutions - Over sampling and under sampling. Over sampling did not seem a good option, so we tried under sampling. We balanced the classes and ran Naive Bayes and Voila!!, it predicts both 0's and 1's. We got an accuracy of 96.4%. And now, we can say that accuracy is a good measure for performance estimation when we have balanced classes.

4) Linear Regression with gradient descent: Linear Regression predicts continuous values. Hence, it is used to predict the running time. As running time is not same as predicting the participations, we changed our feature selection.

a) Features selected: We used "Age","Sex" and "Pace" to predict the running times. We excluded the information about years as it does not have any significance while considering the performance in terms of running time. The features which effect the most are the current age, sex and average pace of the participant over the year. We did not use time to predict the running time as Pace and Time are directly correlated and using one of them, Pace, gave us better accuracy. Rank signifies the position scored by a participant in a specific marathon. Ranks obtained over the different years by a particular participant are not relative because of the non-fixed number of participants in marathons and hence not significant.

b) Hyper parameters: For Linear Regression, we used 100 iterations to minimize the error while using a learning rate of 0.0001.The error was lowest at this point and becomes constant afterwards. Linear regression gave an accuracy of 99.23% which is very impressive. The accuracy value and decreasing error rate gave us assurance about the right choice of features and implementation. Figure 3 shows the cost vs number of iterations used in gradient descent. We did not use regularization as the results seemed pretty descent.

IV. RESULTS-WINNERS

We have two winners based on the two algorithms used for participation. We sorted the file based on the running time from smallest to largest value and checked for the first value where we get a 1 in participation.

1) Winner 1: - This is predicted using Logistic regression and Linear regression. We use the results produced by Hanieh's algorithm. **Winner is - ID 248**

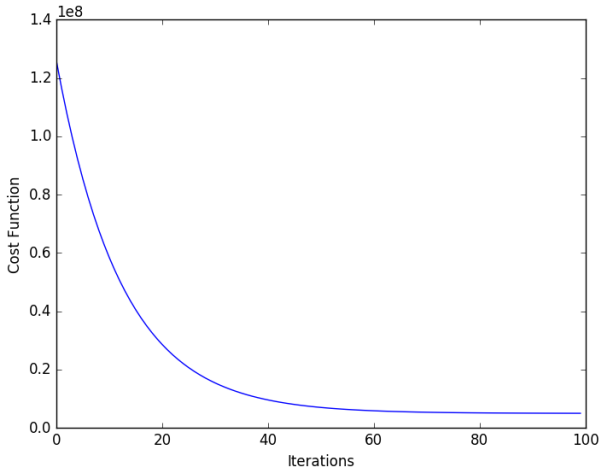

Fig. 3. Cost function vs number of iterations in Linear Regression

2) Winner 2: - This is predicted using Naive Bayes and Linear regression. **Winner is - ID 13290**

V. Discussion

Understanding the given data and deciding a way to use this data according to the given tasks was not easy. It took us a little more than expected to decide the criteria of selecting the initial data to train and test our models. Then, deciding how to aggregate the data by grouping over ids, i.e. taking mean values, maximum etc. was challenging as some features cannot be simply averaged, for instance year and age. Categorization of age and year helped a lot in aggregating the data per participant. We do not claim that our solution for aggregation is optimal but it worked for us (or we would like to believe it). We carefully chose the features per method and trained them iteratively using 3-fold cross validations. Cross validations proved to be extremely useful especially in the case of Naive Bayes when we decided to balance the classes. It was hard to decide about the ratio in which we should sample the classes. We tried 1:1, 2:1 and 1.5 times of one class compared to the other. Finally, we decided to go with the 1.5 times distribution as it resulted in the highest accuracy.
Overall, we learnt a lot from this exercise and hope to do better in the future projects.

VI. Statement of Contributions

As this project was a team exercise, we had a good distribution of work amongst team members. We all equally participated in deciding the initial data on which we trained and tested our models. Preprocessing of data is done by Inderjot whereas Zahra and Hanieh validated the preprocessed data. Hanieh implemented Logisitic regression with gradient descent in R and she normalized the data as well for her part. Inderjot created the files we used for training and testing the models. These files had input vectors and output labels separated for both participation and running time predictions. The criteria of creating these files is explained above in section III. Inderjot implemented all the three methods in Python and normalized the data as well whereas Zahra helped her by pulling the resources for Naive Bayes implementation as it was a difficult model to train with the given data. Finally, Zahra put together the final predictions file. Inderjot created the report while Hanieh wrote the report for her part and Zahra helped Inderjot in putting together the whole report.

References

[1] http://machinelearningmastery.com/implement-logistic-regression-stochastic-gradient-descent-scratch-python/

[2] http://machinelearningmastery.com/linear-regression-tutorial-using-gradient-descent-for-machine-learning/

[3] http://scikit-learn.org/stable/modules/preprocessing.html

[4] http://www.runningintheusa.com/More/RaceFAQ.aspx

[5] http://machinelearningmastery.com/naive-bayes-classifier-scratch-python/