2.2.3

number of horses: 2155

number of jockeys: 105

number of trainers: 93

3.1.1

Prediction evaluation:

|  |  |  |  |
| --- | --- | --- | --- |
|  | f1 | precision | recall |
| HorseWin | 0.284875 | 0.172752 | 0.811715 |
| HorseRankTop3 | 0.518893 | 0.378283 | 0.825874 |
| HorseRankTop50Percent | 0.71189 | 0.622695 | 0.830908 |

Running Time:

Fit (10-fold cross validation): 74.744310 s

Predict: 1.498674 s

3.1.2

GaussianNB, it is because some of the features like those avg\_rank are in float number, and it is likely to have a normally distributed dataset. Since MultinomialNB nor BernoulliNB are for classification with discrete features, so they should not be used in this dataset.

Prediction evaluation (sklearn):

|  |  |  |  |
| --- | --- | --- | --- |
|  | f1 | precision | recall |
| HorseWin | 0.250823 | 0.151972 | 0.717573 |
| HorseRankTop3 | 0.498361 | 0.36248 | 0.797203 |
| HorseRankTop50Percent | 0.70126 | 0.612853 | 0.819473 |

Running Time (sklearn):

Fit (10-fold cross validation): 0.166620 s

Predict: 1.519994 s

Prediction evaluation (my implementation):

|  |  |  |  |
| --- | --- | --- | --- |
|  | f1 | precision | recall |
| HorseWin | 0.250431 | 0.149938 | 0.759414 |
| HorseRankTop3 | 0.49911 | 0.365949 | 0.784615 |
| HorseRankTop50Percent | 0.702245 | 0.614944 | 0.818434 |

Running Time (sklearn):

Fit: 0.006186 s

Predict: 79.991875 s

The result of my implementation of naïve bayes is very similar to the implementation of sklearn, but my implementation uses much more time in prediction.

3.1.3

rbf, since the dataset is large and dimension is relatively small. Also, some of the features seem not to have a linear relationship with the ranking, such as race\_distance, so it is hard to decide a suitable degree of function for the classification.

Prediction evaluation:

|  |  |  |  |
| --- | --- | --- | --- |
|  | f1 | precision | recall |
| HorseWin | 0.120425 | 0.138211 | 0.106695 |
| HorseRankTop3 | 0.35208 | 0.391938 | 0.31958 |
| HorseRankTop50Percent | 0.60289 | 0.537273 | 0.686764 |

Running Time:

Fit (10-fold cross validation): 621.381144 s

Predict: 7.091107 s

3.1.4

Prediction evaluation:

|  |  |  |  |
| --- | --- | --- | --- |
|  | f1 | precision | recall |
| HorseWin | 0.247532 | 0.194279 | 0.341004 |
| HorseRankTop3 | 0.460889 | 0.385593 | 0.572727 |
| HorseRankTop50Percent | 0.663227 | 0.590062 | 0.757103 |

Running Time:

Fit (10-fold cross validation): 4.006915 s

Predict: 1.485075 s

3.4

4.1.1

rbf, similar reason as choosing the SVM kernel. It’s because some of the feature may not have a linear relation with the finishing time, and determine a suitable poly-function is not easy.

epsilon means the maximum margin from the model that can be accepted without penalty, one of the usage is to deal with noise. Setting this value too high will likely to make the model too general as it accepts much more data apart from noise, but setting this value too small may cause overfitting, such that the model fit with the training data but cannot predict well with the test data. So, by testing of different values, 0.5 is chose for epsilon.

C means the penalty value for each error, i.e. outside the maximum margin mentioned above. Set this to a higher value will make the model more aware to the error term, such that the RMSE will decrease. However, a too high value will make the model too specific to predict the finishing\_time, but not general enough to predict the top1 horse within wach race. So, by testing of different value, 5 is chose for C.

4.1.2

quantile, it is more robust in measuring the error, as it is not only considering the mean of the variable. Although the RMSE of using ls/lad/Huber will be lower since they are related to mean-square, quantile function will have a better result on predicting the winning horse.

learning\_rate means how much the model adjust in each round of boosting, while n\_estimators means the total number of rounds to do boosting. Large learning\_rate may cause over-shooting that adjusting too much and miss the target, but small learning\_rate will need many rounds to converge. Small n\_estimators may not enough to adjust the model to the data, while large n\_estimators may overfit the data and increase the training time. So, usually we tune learning\_rate and n\_estimators in a reverse direction, such as decreasing learning\_rate and increasing n\_estimators. By testing of different combination, 0.03 is chosen for learning\_rate and 300 is chosen for n\_estimators.

max\_depth means the maximum tree depth of the estimator, which should be related to the relationship of the feature. A small depth will not be enough to fit the model to the data, while a large depth will make the tree overfit to each feature, which is not good when there is relationship among features. So, by testing of different value, 3 is chosen for max\_depth.

4.2

Without normalization:

(svr\_model, 1915.443705, 0.066667, 0.266667, 6.46875)

(gbrt\_model, 216.880711, 0.239583, 0.564583, 4.095833)

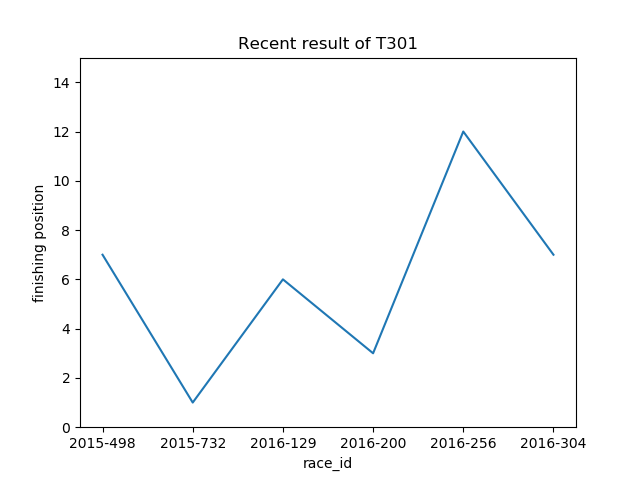
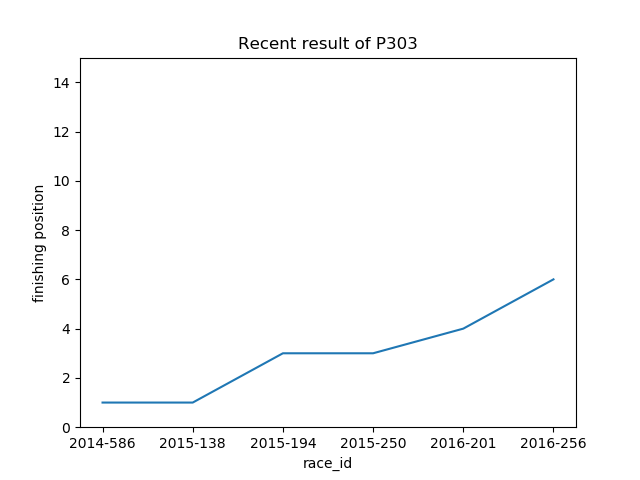
With normalization:

(svr\_model, 705.786726, 0.110417, 0.272917, 6.379167)

(gbrt\_model, 216.883099, 0.239583, 0.564583, 4.095833)

After normalization, the result of SVR improved a lot, as we can see the RMSE drop from 1915 to 706. However, the result of GBRT did not change much, the results are almost the same.

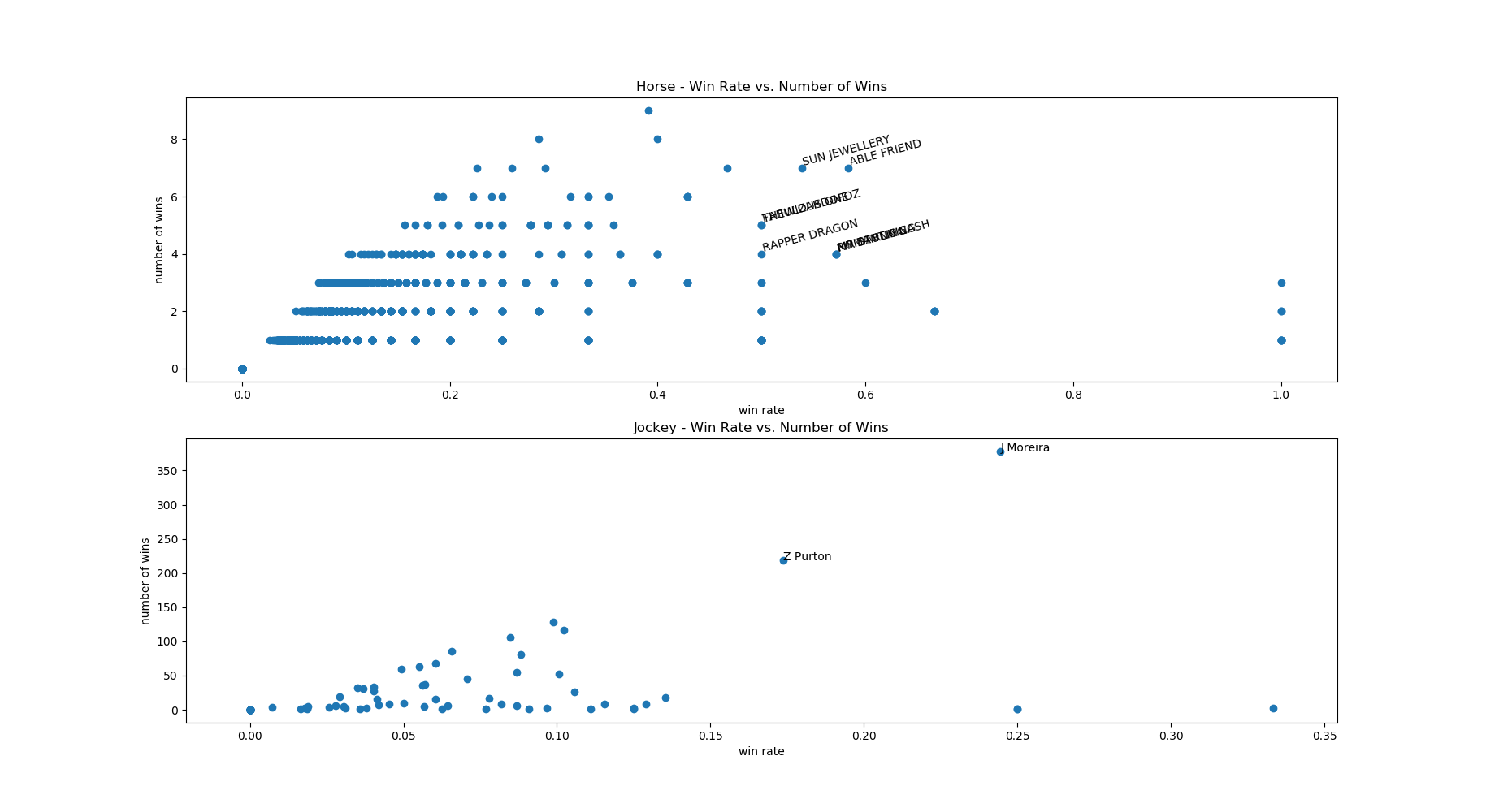
6.1



From the plot of horse P303, it has an increasing trend in ranking, which means the performance of this horse is worsening.

From the plot of horse T301, the line going up and down, which means the horse’s performance is fluctuating.

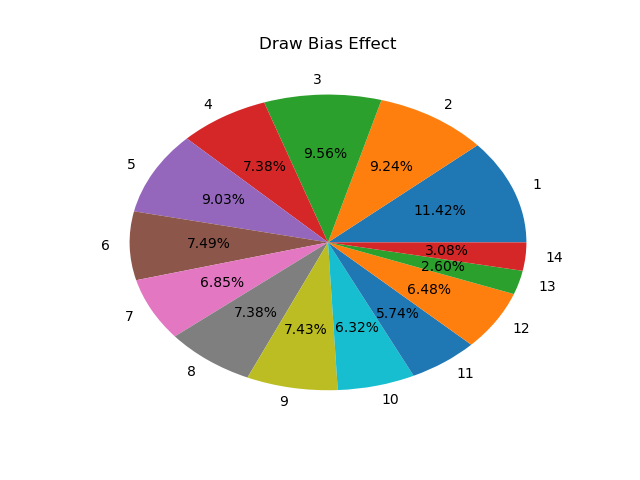
6.2



Best horse: Able Friend. Although there are other horses having win rate of 100%, these horses only join a few races (<4 races), which is hardly to determine if these horses will continue the performance afterward. Therefore, for all horse having more than 4 races, Able Friend have win rate > 50% and it’s win rate also the highest, so it is the best currently.

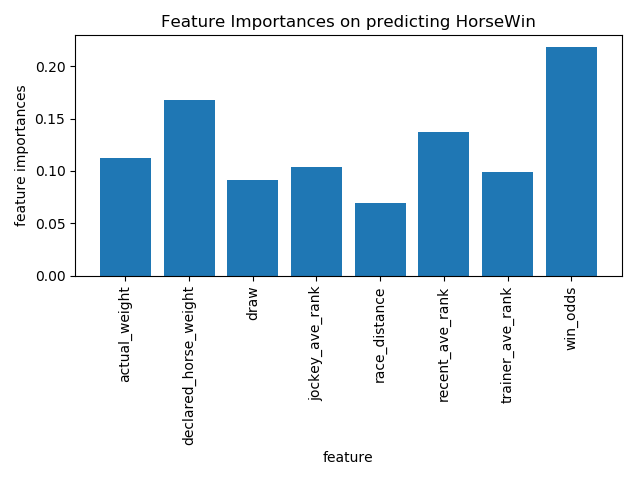
Best jockey: J Moreira. He has the highest number of wins. Although there are jockeys having higher win rate, these jockeys only participate in very few races, so they should not be classified as best jockey at the moment.

6.3



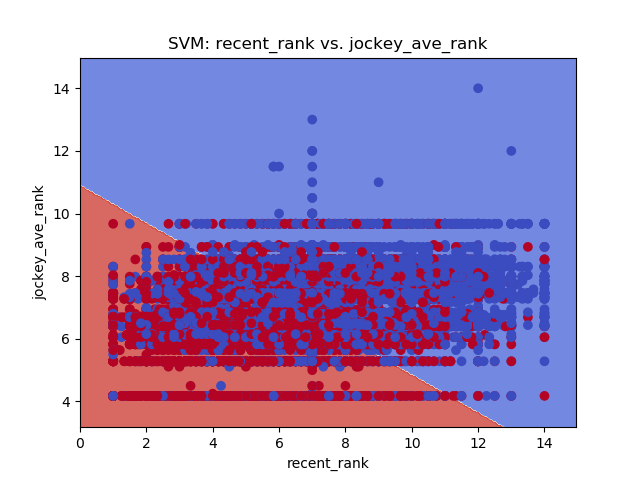
The plot shows that the draw 1 has a highest chance of winning, and the winning chance tends to decrease as the draw number increase. Note that the draw 13 and draw 14 have a relatively low percentage just because not all races have 14 horses, some of them only have 12 horses. The lower draw number has a considerable advantage to win, when comparing draw 1 and draw 11, draw 1 has doubled the win rate. Although the increase is only a few percentage, that’s because there are much more factor affecting the winning rate.

6.4



The plot shows that the feature “win\_odds” have the highest importance on predicting the winning horse, which is reasonable as people usually bet on the horse that have a higher chance to win. The feature “race\_distance” have the lowest importance, which is because the distance is the same for all record in the same race, so it can’t have a large importance. It is interesting to find that the “declared\_horse\_weight” have a relatively higher importance, when compare to the “actual\_weight”, since the “actual\_weight” should be related to the horse’s previous performance.

6.5



This plot shows that higher recent\_rank or higher jockey\_ave\_rank does have a higher chance to rank higher. SVM does try to find a best line to separate 2 classes, although they can’t be totally separated. As we can see that the blue plane has more blue points on it while the red plane has more red points on it. The line seems to pass through the point (7,7), which is the mean point, but have a lower y-intercept than x-intercept, which may shows that the jockey\_ave\_rank is more important.