Chan Maxwell Chun Sum (1155079378)
Leung Kwan Ho (1155077754)

CSCI3320 Project report

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
linear, although rbf or other kernel can be powerful in classification as they have more degree of freedom, they tend to fit with the training data but not predict well with testing data, causing overfitting. Although not all features seem to be linearly related with ranking, using linear kernel is already enough to do prediction for this dataset.

Prediction evaluation:

	f1	precision	recall
HorseWin	0.267176	0.191956	0.439331
HorseRankTop3	0.502960	0.366337	0.802098
HorseRankTop50Percent	0.713341	0.592220	0.896743

Running Time:
Fit (10-fold cross validation): 536.387577 s
Predict: 4.461310 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
Logistic regression has the highest F1-score among all models, although the fitting time is a bit long. However, the recall is much higher than precision in this model, which means that the model seems not specialize enough. Besides, this model won't be done well if the data is extremely imbalanced. We have tried to use the same model to predict if the horse win or not instead of predict ranking, but the F1-score drops to almost zero.
Naïve bayes model perform classification very fast, it's F1-score is just behind logistic regression. It also has a large gap between precision and recall, so it maybe also too generalizes. Unlike logistic regression, naïve bayes perform better when the features is independent, but the features used to train this dataset is not totally

independent, so this may affect the performance of this model.
SVM model uses very long time for both fitting and predicting data. It has a relatively high precision score and small gap between precision and recall for HorseWin only, because we are using linear model, it tends to be more generalize. If rbf kernel is used, the gap between precision and recall will be much smaller, as rbf did better to specialize the model. The distance between data points within a feature is important in SVM, therefore we need to do normalization on data in order to improve the result.
Random forest cannot achieve as high F1-score as other classifiers, but it can achieve relatively high precision and smallest difference between precision and recall, which means it done relatively well in balancing generality and specificity. Also, it can finish fitting and predicting quite fast. Random forest, like decision tree, is driven by sets of decision rules, so that it may handle feature with non-polynomial function better, just like race_distance in our data.

Logistic regression: When data is not very imbalanced, better for linear features. e.g. Classify flower with respect to their length and width
Naïve bayes: When the features are independent to each other. e.g. Text classification
SVM: Having large number of features, and data should be able to normalize. If timing is a constraint, having relatively small number of data. e.g. Image classification
Random forest: When features has non-numerical labels, or the features may not in linear or polynomial function. e.g. Disease classification base on human habit

By cross-validation, we further divide the training set into serval partitions, each time use one partition for validation and others for training. If the model is overfitted to training set, then it will have low score on validation set. If we choose the model with highest score, that's mean the model is less overfit and perform better on unseen data (validation set) when compare to other models. Therefore, this model tends to perform better on another set of unseen data, which is our testing set.

Take HorseWin as an example, as this label is extremely imbalanced, only around 8% of data is positive. If the model is too generalized that predict many positive label, just like LogisticRegression in this case, the recall value will be high, as predicted positive label can easily cover most actually positive data. However, the precision value will be low, as many predicted positive labels will be actually negative. NPV and TNR are just similar to precision and recall, but they focus on negative data instead of positive. Therefore, all these metrics cannot reflect the true performance of a model when they are used alone. This is the reason why we include F1-score in the evaluation, which represent the harmonic mean of precision and recall, such that it can better descript how good the model is when predicting imbalanced data.

4.1.1
linear, similar reason as choosing the SVM kernel. Although not all features are linear function, using linear can easily prevent overfitting.

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, 158.933256, 0.202083, 0.460417, 4.735417)
(gbrt_model, 216.880711, 0.239583, 0.564583, 4.095833)
With normalization:
(svr_model, 158.908674, 0.204167, 0.468750, 4.704167)
(gbrt_model, 216.883099, 0.239583, 0.564583, 4.095833)
After normalization, the result of SVR improved a bit, while the results of GBRT are almost the same.

5

Basic Strategy

LogisticRegression	37.7
NaiveBayes	-59.0
SVM	-74.6

RandomForest	-7.2
SVR	-37.3
SVR with normalize	-45.0
GBRT	-12.7
GBRT with normalize	-12.7

For classification models, for each race, all HorseWin horses will be selected as candidate. If there is no HorseWin horses, then all HorseRankTop3 horses will be selected. If there is no HorseRankTop3 horses, then all HorseRankTop50Percent horses will be selected. If there is no HorseRankTop50Percent horses, then all horses will be selected. For the candidate horses, we choose the one with maximum declared_horse_weight. It's because win_odds and declared_horse_weight have a higher feature importance, and by experiment higher declared_horse_weight have a better result.

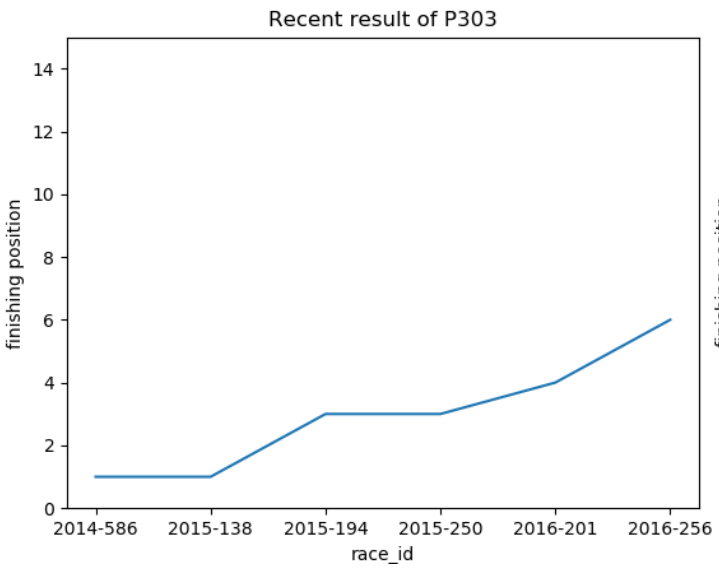
For regression models, although the prediction of finishing time can infer only one winning horse, as the accuracy of the prediction is not high enough, so we will pick the top 2 horses in each race, and further decide the only winning horse by the same method above, i.e. the one with maximum declared_horse_weight.

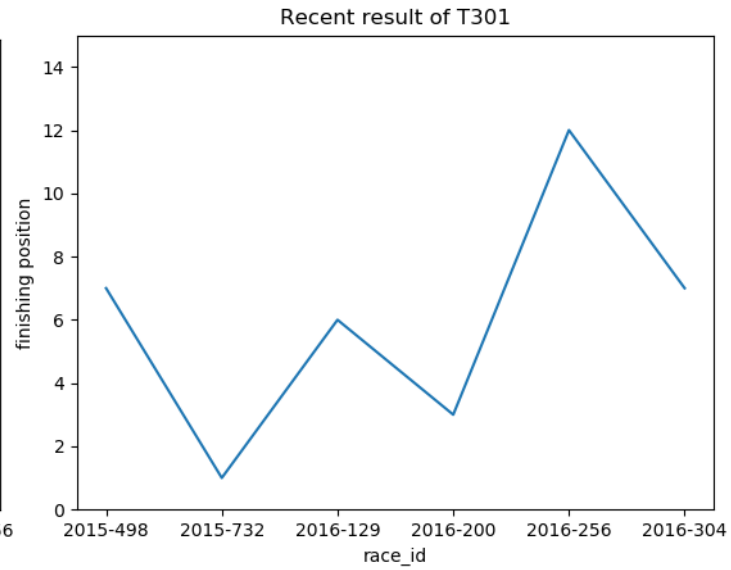
Our own strategy: 68.0

We make use of the best model from classification and the best model from regression, i.e. LogisticRegression and GBRT. So, we first select candidate from all horses that is HorseWin in LogisticRegression, and all horses that is top 3 in GBRT, for each race. To select the only winning horse from candidates, we use the same method as above, i.e. the horse with maximum declared_horse_weight. If the win_odds of that horse is small than 30, we will bet on it with $1, otherwise we won't bet on it since it is too risky.

Base on this strategy, the total amount we get will be proportional to the amount we bet, e.g. if we bet $10 instead of $1, then we will get 680.0

6.1

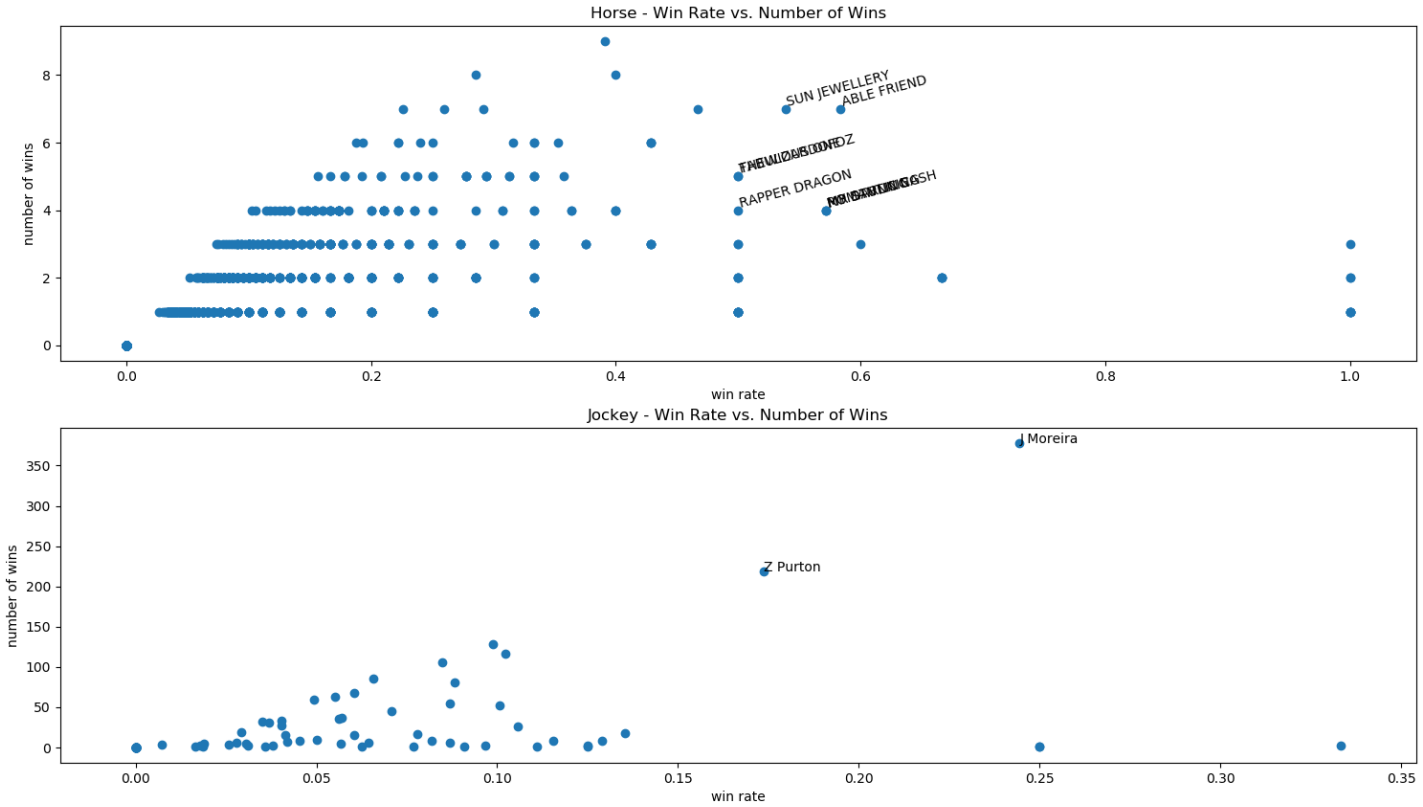




From the plot of horse P303, it has an increasing trend in ranking, which means the performance of this horse is worsening.

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

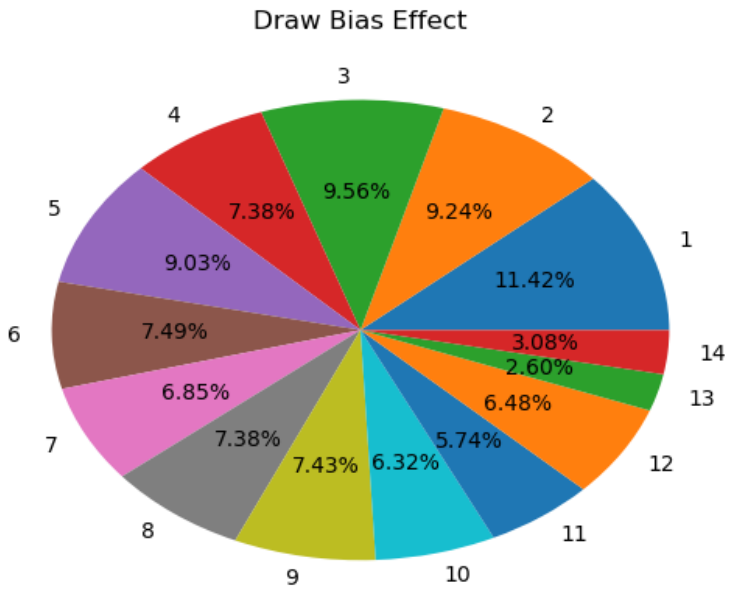
6.2



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.

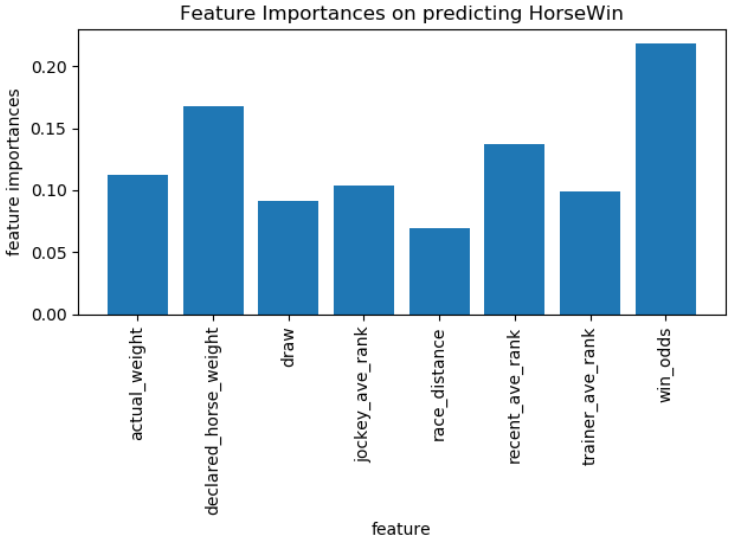
Best jockey: J Moreira. He has the highest number of wins. Although there are jockeys having higher win rate, these jockeys only participate in very few races, so they should not be classified as best jockey at the moment.

6.3



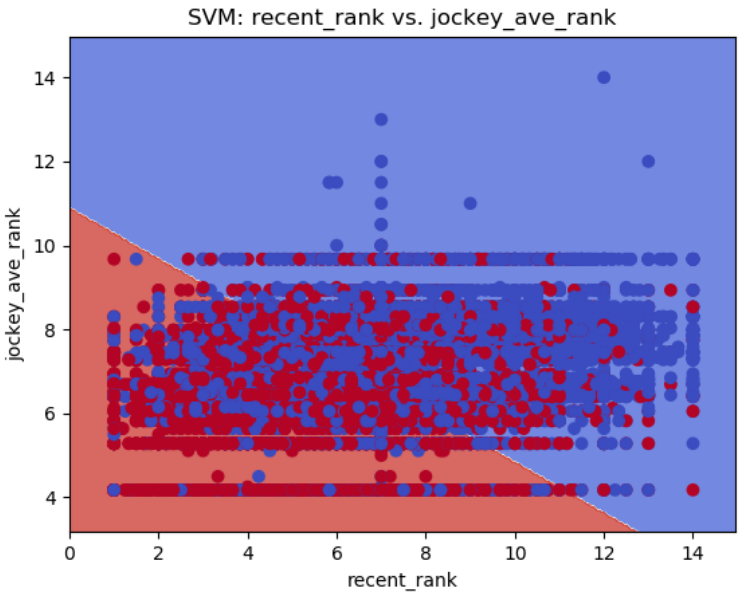
The plot shows that the draw 1 has a highest chance of winning, and the winning chance tends to decrease as the draw number increase. Note that the draw 13 and draw 14 have a relatively low percentage just because not all races have 14 horses, some of them only have 12 horses. The lower draw number has 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 show that the jockey_ave_rank is more important.