1. Data Cleaning

- We chose to remove following predictors from the analysis: “AgeRestriction”, “Barrier”, “ClassRestriction”, “FoalingCountry”, “FoalingDate”, “GoingAbbrev”, “GoingID”, “HandicapDistance”, “HandicapType”, “RaceOverallTime”, “RacePrizemoney”, “TrackID”.
- “AgeRestriction”: although we admit that there is a general trend that younger horses perform better in shorter distance track while older horses perform better in longer distance track since "FinishPosition" was more reliant on "Age" and "Distance" rather than its "AgeRestriction", we figured that the "AgeRestriction" alone was not as impactful as it depends more on the distance of the track.
- “Barrier”: since the gate does not start right at the gate immediately, we believe that there is not much correlation between the “Barrier” and the “FinishPosition” of the horse.
- “ClassRestriction”: this predictor contains 2243 unique values while containing multiple information in a single variable (too much information).
- “FoalingCountry”: most horses were born in France, and even if this was not the case, “FoalingCountry” was not so reliable for our analysis to estimate the winning probability because logically there can always be those horses who exceptionally perform well.
- “FoalingDate”: this predictor is supposed to explain when the horse was born which is identical to the “HorseAge” – due to this, we have only kept “HorseAge” as our predictor instead.
- “GoingAbbrev” and “GoingID”: we figured “GoingAbbrev” and “GoingID” are identical across the same RaceID. This shows that horses in the same race would share the same track condition. Due to this, we expect that these two predictors will not affect the winning probability between horses since it will be held constant within the same race. “GoingID” has a corresponding numerical code by "GoingAbbrev". For the same reason as “GoingAbbrev”, “GoingID” was not included in our predictor.

- “HandicapDistance” and “HandicapType”: we decided to omit these two variables because we figured both have a very weak correlation with the FinishPosition.
- “RaceOverallTime”: we figured all horses in the same race (RaceID) share the same race time overall. This would not correlate with the probability of each horse’s winning.
- “RacePrizemoney”: since this price is the overall prize involved in that race, this predictor is not expected to have a correlation with each horse's winning.
- “TrackID”: since every race shares the same track, we omitted this variable as our variable.

2. Description of the Model

Note: hyperparameters used for the models can be seen from the published code.

- Winning categories were assigned as a binary variable for every horse who obtained prize money.
- Then, a logistic model was performed with the chosen predictors which are "BeatenMargin", "Prizemoney", "PIRPosition", "PriceSP", "NoFrontCover", "WideOffRail", "Disqualified", and "FinishPosition" to create winning probabilities based on winning categories that we have created. A logistic regression model from Scikit-Learn was used. Before performing the logistic regression, StandardScaler was used to scale the predictors appropriately.
- The probabilities generated by logistic regression were scaled using “Prizemoney ” predictor. Constant factors were added to both “Prizemoney” and generated probabilities to avoid 0 probability.
- Data was split according to the information provided by the instruction: the train data consists of every race start time before than 2021 Nov 1st whereas the test data consists the rest of the data.
- Then, Stacking Regression was implemented to perform machine learning in our dataset. StackingRegressor from Scikit-Learn library from Python was chosen to be

used. The estimators used for the StackingRegressor consists of random forest regressors and MLPRegressor from the identical library.

- OneHotEncoder was used to vectorize the categorical data while StandardScaler was used to scale the numerical data.
- Then, Pipeline from Scikit-Learn was used to implement processors with regression models.
- We wanted to include more models such as Gradient Boosting in our model. However, the computation time was taking too long. Thus, only the mentioned models were used.
- Mean Squared Error was computed to analyze the performance of the prediction.
- The computation time was around 25 minutes, and the forecast of the test data successfully reflected the top 7 horses for each race with 58.67 % accuracy.
- A separate R Markdown file was created to calculate the above accuracy. This file can be found under Test folder in the repository.