Team: 11
Team members: Simon Fung, Austin Laswell, Nathaniel Freuler
Title: Predicting Horse Race Outcomes from the Hong Kong Jockey Club

Introduction:

The problem this project aims to solve is the uncertainty and limited information available to the average person when betting on horse races. While bettors often have access to win odds and a few basic statistics about the horse, they do not have an easy way to utilize the full depth of data available. Through a model that incorporates this entire dataset, this project empowers gamblers to make more informed and data driven bets.

The motivation behind this project came from the challenge of adding statistical insight to something as unpredictable as horse race betting. The majority of gamblers rely on win odds so this project aimed to leverage more than just that piece of information. The approach for this project was as a binary classification task that predicted whether a horse would finish in the top three or not. It was structured based on how bets would normally pay out, and is more useful than predicting the exact placement of the horse. The final model was compared to multiple baselines that included a model that always predicts zero for top 3, a random choice, and a simple logistic regression baseline. Precision was the main evaluation metric chosen since gamblers lose money on false positives. Recall was also tracked however less important since false negatives represent missed wins than actual losses.

The final XGBoost model outperformed all baselines, including a naive classifier, random guessing, and logistic regression. It scored highest in precision, indicating it would make fewer but more reliable betting recommendations, minimizing financial risk to the bettor. The strong performance by the model shows that utilizing historical race data and engineered features provides a clear advantage over relying solely on win odds or intuition.

To aid downstream models, GloVe inspired embeddings were used to capture the global relationships between horse, jockeys, and their competitors. For an unsupervised method, GloVe was not an obvious first choice because it was originally developed for text classification problems. Predicting race outcomes are essentially one in the same and GloVe is often used for ranking text in search recommendation systems. Unfortunately, GloVe's greatest strength is also a weakness. While it is great at capturing global relationships, it lacks the ability to identify localized patterns. As a complement to GloVe, UMAP embeddings were also explored to uncover local structures GloVe could not.

Both unsupervised approaches were successful at reducing the complexity of the data and improving predictive power. Glove reduced thousands of horse combinations into a series of embeddings of less than a hundred latent features while UMAP was used to further reduce the GloVe embeddings into smaller clusters. These latent features produced the highest error reduction in Gradient Boosted Tree (GBT) models when compared to the raw alone. That said, it is incredibly difficult to outperform the intuition of the crowd ("The Odds"). The majority of models using extracted, engineered, and raw features underperformed "The Odds" in terms of accuracy, precision, and recall. A handful of GBT models were able to match the precision and accuracy of "The Odds", but at the cost of a decrease in recall. Lastly, Models that utilized the odds with extracted features were able to outperform the intuition of the crowd by ~10 pts in precision, but also at the cost of decreased recall. The key takeaway from the unsupervised analysis is that the "Odds" encode latent information unknown to the analyzed dataset.

Related Work:

"Ga Yau: Machine Analysis of Hong Kong Horse Racing Data" – Stanford CS230 winter-2021 course project

Accessible at: https://cs230.stanford.edu/projects_winter_2021/reports/70738477.pdf

The study used a hand-picked subset of publicly available HKJC race variables and trained Gradient-Boosting models alongside shallow fully connected neural networks to predict the single winning horse.

In contrast, our project targets the Top-3 finishers (order-agnostic), feeds the full HKJC feature set (with feature engineering), and enriches it with weather data to provide a more comprehensive input.

Kwok, K. “Hong Kong Horse Racing Prediction — Part I–III” – Medium series (2023)

Accessible at: https://medium.com/@kahoikwok/data-analysis-project-hong-kong-horse-racing-prediction-part-i-e63103155658

This three-part blog scrapes HKJC data with Selenium, develops a horse & jockey scoring system using the past race data, visualises them in Power BI, and then fits a simple placement model with a focus on long-distance races.

In contrast, our project employs a full supervised machine-learning pipeline: we first learn dense embeddings for horses, jockeys, and race contexts via unsupervised methods, then feed those embeddings—together with raw race and weather features—into different models that predicts whether a runner will finish in the first 3 placing.

Wong, Y. “Horse Racing Prediction using Deep Probabilistic Programming with Python and PyTorch (Uber Pyro)” – CUHK BEng Thesis (2018)

Accessible at: https://www.cse.cuhk.edu.hk/lyu/_media/thesis/report-1805-1.pdf

The thesis applies deep probabilistic programming to build a Bayesian neural network that predicts each horse’s exact finishing position and derives a win/place betting strategy. In the prediction model, embeddings are deliberately omitted.

Our project instead targets the Top-3 classification and augments the standard HKJC features with weather, equipment-change flags, and unsupervised embeddings that feed an XGBoost classifier.

Data Source:

This project leverages a rich collection of datasets from the two sources – the Hong Kong Jockey Club (HKJC) and the Hong Kong Observatory (HKO), offering over a decade of comprehensive horse racing data enriched with contextual information such as weather conditions, jockey and horse profiles, competitors and closing betting odds.

These datasets were acquired through a custom-built web scraper and public APIs, then unified and engineered into machine learning-ready formats.

Sourced from HKJC:

- Race Details and Results
- Horse and Jockey Profiles
- Race Comments and Gear used

Sourced from HKO API:

- Weather Data

Most of the data spans 2010–2025, encompassing over 150,000 race records, some data are not unavailable for certain time periods. For example, only partial horse specific information (e.g. color, sex and country of origin but not age) are available for retired horses; Temperature is available since 2010, but humidity, rainfall and UV index are only available after September 2019. Data imputations are discussed in later sections.

Below is a summary of selected features. The data description for all features is in appendix 1.

1. Race Results (HKJC Local Results)
Data Type: Structured HTML tables parsed into CSV
Size: 12,343 races with at most 14 participating horses in each race, total 154822 rows
Time Span: 2010-01-01 to 2025-06-22.
Importance: This is the central dataset used to define the prediction target (e.g., whether a horse/jockey finishes in the top 3) and supplies most race-level features.

Key Features from Race Data:

Feature	Description	Type
Placing	Finishing position of the horse	Numeric
Horse No.	Horse number in race	Numeric
Horse	Horse name	String
Jockey	Jockey name	String
Trainer	Trainer name	String
Act. Wt.	Actual weight carried	Numeric
Declar. Horse Wt.	Declared weight of horse	Numeric
Dr.	Draw position	Numeric
LBW	Lengths behind winner	Numeric
Win Odds	Win betting odds	Numeric
Date	Race date	Datetime
Course	Racecourse location	String
RaceNumber	Race number on card	Numeric
Race type	Race class/type	String
DistanceMeter	Race distance (m)	Numeric
Score range	The score ranges the competing horses are in	Numeric
Going	Track condition	String
Handicap	Assigned handicap	Numeric
Course Detail	The track of the course used in the race	String
Finish Time	Completion time	Datetime

2. Horse and Jockey Profiles

Data Type: Structured HTML tables parsed into CSV

Size: ~8,253 active and 7,023 inactive horses.

Importance: Provide static and semi-static attributes used to understand horse ability, lineage, and form.

Feature	Description	Type
HorseIndex	Unique horse identifier	String
Country	Horse origin	String
Age at race	Horse age on race day	Numeric
Colour	Horse color	String
Sex	Horse sex	String
Import Date	Date imported	Datetime
Owner	Owner name	String
Current Rating	Current handicap rating	Numeric
Sire	Horse sire	String
Dam	Horse dam	String
Dam's Sire	Dam's sire	String
Same Sire	Horses with same sire	String

3. Race Comments and Gear

Data Type: Structured HTML tables parsed into CSV

Size: 152,669 comments. (not every race is commented)

Importance: Adds human-labeled narrative context and equipment usage (gear) that may influence performance.

Feature	Description	Type
Comment	Official race commentary	String
Gear	Gear used on horse in the race and if it is the first time being used	String

4. Weather Data (Hong Kong Observatory)

Size: 1,328 daily observations matched to race dates.

Time Span: 2010-01-01 to 2025-06-22.

Importance: Provide weather context to enrich the HKJC dataset to understand impact of weather condition on horse/jockey performance.

Feature	Description	Type

MeanTemperature / MaxTemperature / MinTemperature	Daily temperatures available for all dates	Numeric
Rainfall(mm) / MeanUVIndex / RelativeHumidity	Environmental stress factors. Only available post-2019-09-10	Numeric

Feature engineering:

Competitor information:

Each entry in the dataset originally described one horse and all of that horse's own details for a given race, without capturing who else it competed against. To add information about the surrounding competition, new columns are constructed to list every other horse and jockey in the same race. For each race, the script gathers all participating horse identifiers and jockey names, then—row by row—stores the identifiers of all the other horses in columns HorseCompetitor1 through HorseCompetitor13, and likewise stores the other jockeys in columns JockeyCompetitor1 through JockeyCompetitor13. Because Hong Kong races have at most fourteen horses, each horse (entry) ends up with up to thirteen horse competitor slots and thirteen jockey competitor slots. As a result, a single row now carries not just the attributes of a particular horse but also a complete snapshot of its opponents—allowing models to consider how a horse's performance may depend on the specific competitors in that race.

Preprocessing:

Data type conversions

- Converted numeric values stored as text to int or float as appropriate
  - Placing was stored as text and need to be converted integer for numeric analysis
- Converted fields stored as numeric values to string for group by queries
- Converted dates stored as string such as "Race Date" to datetime objects for query tasks
- Missing horse data was usually a withdraw, records were excluded from analysis
- If raw data was missing and it was under a few percentage points of total data, the rows were excluded from training data

Trailing Statistics:

- Created trailing Placing and Finishing Time statistics to give non-sequential models such as GBT an understanding of past race performance
- Trailing averages were grouped by RaceNumber, Date, and Horse
- Date was joined back recursively to itself using pandas shift and rolling window to avoid data leakage from future races
- The final trailing averages used in model were Placing_TR1 through Placing_TR10

Embeddings:

- Calculated GloVe embeddings using a co occurrence matrix
  - One matrix for Horses and One matrix for Jockeys
- Co-occurrence calculated as the absolute difference between horse placings
  - co_occur += 1 / abs(horse_placing1 - horse_placing2)
- 50 dims per jockey and horse embedding, a total of 100 features
- Used the simplified gradient descent loss version of the GloVe algorithm
- Optimized embeddings based on an Average Pair Loss (APL) at or below 0.25 for horse embeddings and .05 for jockey embeddings
  - APL was considered optimized after 10 consecutive iterations lower than 0.0001 improvement
- Used UMAP to further reduce GloVe embeddings to identify local structures
- Created 10 UMAP embeddings for jockey and horse relationships, 20 total features
- Optimized UMAP embeddings based on a trustworthiness score >= 0.70
- Further optimized embeddings based on how well clusters could be visually Identified
  - Didn't want to lose predictive power by overly persevering neighbor relationships in original space
  - N_neighbors needed to be 5 or lower to achieve an 80% trustworthiness
    - 80% trustworthiness benchmark taken from Medium.com post "On the Validation of UMAP", (Frenzel, 2021)

List of Final Features: See Appendix 2

Supervised Learning:

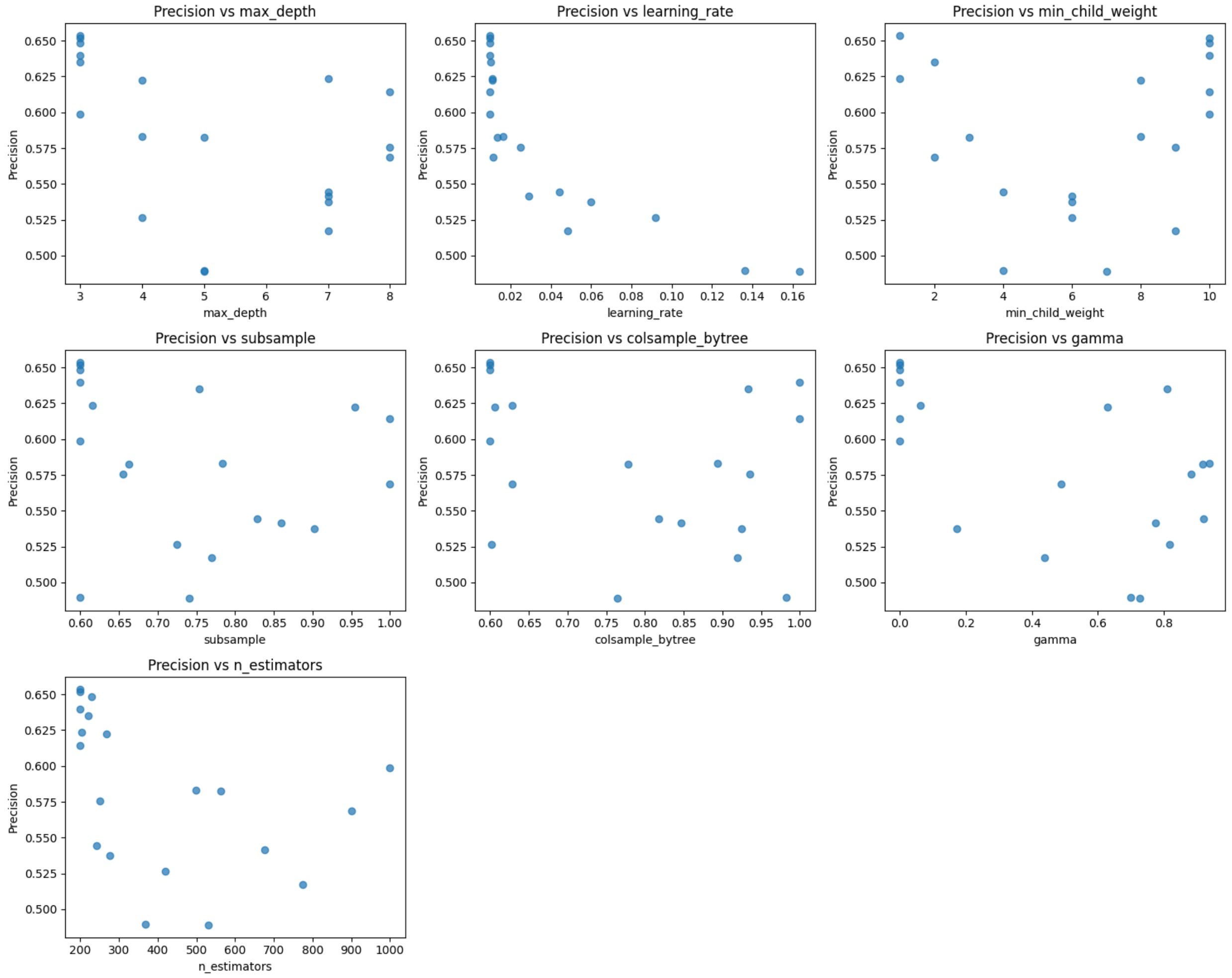
Methods:

The approach chosen for this project was predicting whether a horse would finish in the top three of a race as a binary classification task. The workflow began by training the model on a historical dataset of horse races from 2019 to 2025, with testing on the 2025 races to simulate real world usage. The project compared three distinct types of models: non-parametric baseline models, a logistic regression model, and a gradient-boosted tree model using XGBoost. The non-parametric models included naive and random classifiers, which served as baselines to contextualize model performance. Logistic regression was chosen as a simple and interpretable baseline. XGBoost was chosen as the final model due to its ability to handle missing values, mixed feature types, and feature important analysis.

There were 3 different feature representations present in our dataset. The first was standard race features like draw, weight, distance. The second included trailing statistics for the horse, such as past performances, what distance the race was, what time the horse ran, and the placement of the horse. The third feature representation was embedding vectors representing jockey and horse interactions. The model was tested with different subsets of these feature representations to assess the contribution of each to overall performance.

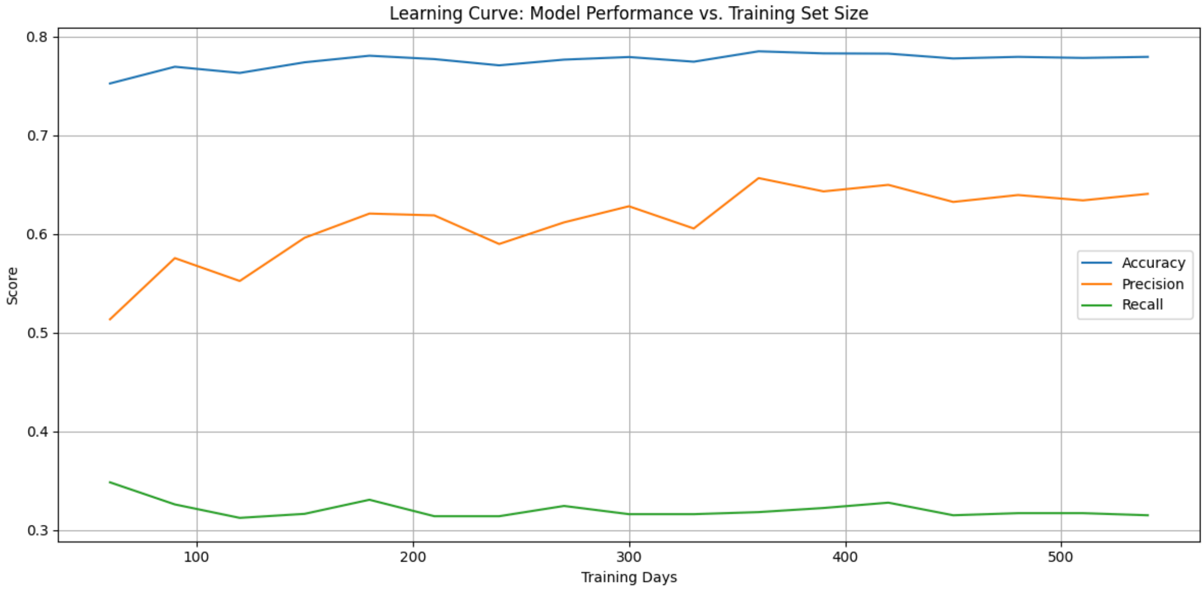
Hyperparameter Tuning:

For the XGBoost model, hyperparameters were tuned using a Bayesian search strategy optimized for precision. To see how sensitive the model's performance was to different hyperparameters, precision was plotted against each hyperparameter. For most of the parameters, the model tended to perform better with lower tree depths, likely helping to avoid overfitting to the training data. Also, the learning rate has a negative relationship with precision, showing that keeping a low learning rate has effectively avoided overfitting. Other parameters like subsample, gamma, and colsample_bytree had less consistent trends, likely having a more data dependent influence on the performance.



Learning Curve:

A learning curve was generated to determine the optimal amount of training data. Recall and accuracy stayed mostly consistent no matter how many days were used to train the model. However, precision increased and then plateaued around 350 days. Based on this trend, 480 days of training data was used for the final model, as that was about the midpoint of the plateaued region.



Results:

The primary evaluation metric was precision. In the context of gambling, false positives represent losing money, so minimizing them was prioritized. Recall in this case would be used to identify missed opportunities, but would not cost the gambler for missing out on them. While it would be nice to identify those bets, the gambler is not losing money the same way as they would be if they placed a bet that did not pay out.

The most conservative and highest precision model was the precision-optimized XGBoost. It only placed bets on about 1 out of 20 horses but had an 85% success rate when it did. The regular XGBoost model had lower precision but would place more bets, making it better suited for high volume betting with lower stakes.

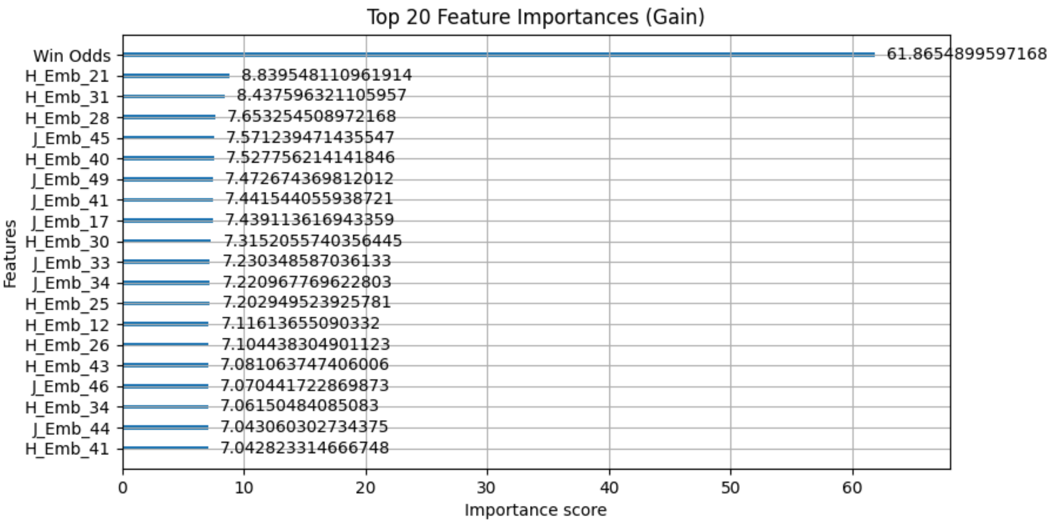
5-Fold Cross-Validation:

Model	**MAE**	**Accuracy**	**Recall**	**Precision**
Logistic Regression	0.358 ± 0.010	0.642 ± 0.010	0.619 ± 0.016	0.379 ± 0.008
XGBoost	0.235 ± 0.005	0.765 ± 0.005	0.238 ± 0.017	0.611 ± 0.027
Precision Optimized XGBoost	0.247 ± 0.002	0.753 ± 0.002	0.051 ± 0.008	0.800 ± 0.073

Model	**MAE**	**Accuracy**	**Recall**	**Precision**
Dummy Variable (Always 0)	0.25	0.75	0	0
Random Choice	0.38	0.62	0.24	0.24
Logistic Regression	0.34	0.66	0.63	0.40
XGBoost	0.22	0.78	0.32	0.64
Precision Optimized XGBoost	0.25	0.75	0.05	0.85

Feature Importance:

Win odds is by far the most dominant and predictive feature, with an importance score of over 60. The next most important feature was a mix of the horse and jockey embeddings, suggesting that the model is also learning useful representations from past race data. Although no individual embedding or trailing statistic comes close to win odds in terms of predictive power.



Ablation Analysis:

To understand the contribution of the different feature sets, an ablation analysis was done by removing subsets of features and noting changes in model performance. Removing the embeddings or trailing statistics had little effect when win odds was included, which confirms its overwhelming influence. However, when win odds was excluded, these feature sets had a larger impact. In this case, the best performing model included trailing statistics and race features but not embeddings. The final win odds value is only available right before the race starts, so using a model that does not include win odds might allow for earlier bet planning in exchange for a small sacrifice in precision. It is worth noting that - the result indicates that trailing record (recent performance stats) is very likely acting as a proxy for win odds. Odds effectively compress recent form into a single number (like a form of embedding), so once they are available, trailing stats contribute minimal incremental information. Conversely, in a scenario where odds are not yet posted, trailing stats boost precision by 0.08, making them valuable for earlier race-planning.

Win Odds Included:

Feature Set	**MAE**	**Accuracy**	**Recall**	**Precision**
All Features	0.22	0.78	0.32	0.64
No Embeddings	0.22	0.78	0.31	0.64
No Trailing Stats	0.22	0.78	0.32	0.64
Only Race Features	0.22	0.78	0.31	0.64

Win Odds Excluded:

Feature Set	**MAE**	**Accuracy**	**Recall**	**Precision**
All Features	0.25	0.75	0.18	0.53
No Embeddings	0.25	0.75	0.11	0.60
No Trailing Stats	0.26	0.74	0.11	0.45
Only Race Features	0.26	0.74	0	0

Failure Analysis:

The majority of the models failures were false negatives, which makes sense given that the model was optimized for precision at the cost of recall. Included below are two examples of false negatives and two examples of false positives. For the false negatives, both instances involve the same horse, A Americ Te Specso. That is a horse with high win odds (not favored to win) but ended up placing first in both cases. It may suggest that the model may be weighing win odds too heavily and not giving enough weight to prior performance. This is expected for a conservative model, it is less likely to take risks on longshot horses, even when there may be some historical evidence that indicates they may perform well. As for the false positives, Beauty Destiny was a highly favored horse due to its low win odds, but ended up placing significantly lower than expected. Again, this suggests that the model is overvaluing win odds, which is problematic when favorites underperform. The horse, Another World, was another highly favored horse, but placed just outside of the top 3 finishes, finishing 4th. These kinds of edge cases are somewhat unavoidable given the hard cutoff, and they are expected occasionally. To address these failures, the model could be improved by reducing its reliance on win odds. It may also benefit from weighting recent performances higher.

Type	**Horse**	**Win Odds**	**Probability**	**Placement**
False Negative	A Americ Te Specso	21.0	0.224162	1.0
False Negative	A Americ Te Specso	17.0	0.251652	1.0
False Positive	Beauty Destiny	3.0	0.776582	10.0

False Positive	Another World	2.2	0.823925	4.0

Unsupervised Approach One | GloVe Embeddings

To improve model performance and feature intuition, GloVe inspired feature embeddings were leveraged to provide downstream supervised machine learning models context around the global relationships between horses, jockeys, and their opponents. The intent was to teach the models how similar each jockey and horse was relative to their competitors. However, given the breadth of data and the complexity of interactions between the training features, defining those relationships proved difficult. The embeddings were chosen for their ability to define these relationships as latent numerical values for machine learning models to consume. Additionally, the features encoded in these embeddings helped reduce downstream machine learning tasks dependency on high dimensional space of thousands of binary encoded fields representative of each horse/jockey's competitive set within a race.

Fundamentally, the task of ranking outcomes in a horse race isn't that much different than how GloVe embeddings are utilized in natural language tasks. For instance, a list of distinct racing horses is comparable to a corpus of words and each horse can be considered a token within a race "document." Just like word tokens within a text document, each horse co-occurs together at a distance defined by their finishing places relative to each other. Once the similarities between word tasks and ranking horse outcomes were understood, GloVe inspired embeddings became a natural choice given its ability to preserve global relationships between words within a corpus through co-occurrence matrices (ie. How similar or dissimilar in skill is Jockey A compared to the competitive set).

The version of the GloVe algorithm used to calculate the race embeddings was adapted from a Medium.com post, "Implementing Glove from scratch - Word embeddings for transformation" (Deepanshusachdeva, 2023). This version of the GloVe algorithm was chosen for its simplicity and interpretability (opposed to neural network implementation). The primary adaptation to this approach was how the co-occurrence matrix was calculated. Firstly, the race data was not in the form of a corpus and text documents. To handle the difference in data structure, the corpus was modeled as the distinct horses and/or jockey's in the race dataset, text documents were equivalent to a set of rows for a single race (conceptually, each row is a sentence and each horse/jockey a token), and the distance between tokens was calculated as the absolute variance between the each horse/jockey's finishing place relative to every other opponent in the race.

GloVe Embedding Unsupervised Evaluation

The embeddings were tuned by adjusting the number of dimensions within each embeddings (Dims), Learning Rate (LR), Total Loss (TL) (predicted value - actual value for all comparisons), Average Pair Loss (APL) ((predicted value - actual value)/all comparisons)), and an early stopping rule of 10 consecutive improvements less than 0.0001 in APL.

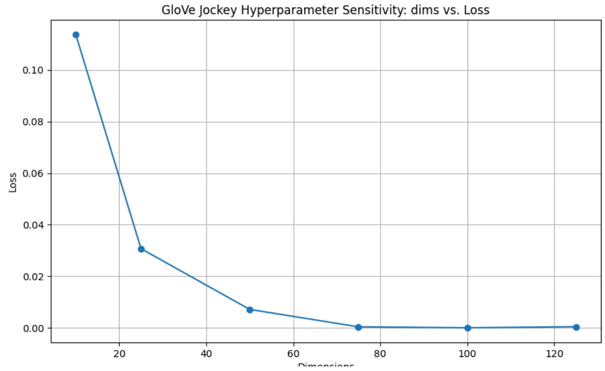
The most important hyper parameters were APL, LR, and Dims. APL proved more informative than TL when comparing best fit over multiple embeddings. To clarify, because many embeddings were trained on varying sizes of data, TL would decrease or increase in respect to the number of rows; making it difficult to compare best fits across all embeddings. After 10 consecutive improvements less than 0.0001 in APL, the average best fit horse and jockey embedding APL converged at ~ 0.25 and ~ 0.05, respectively.

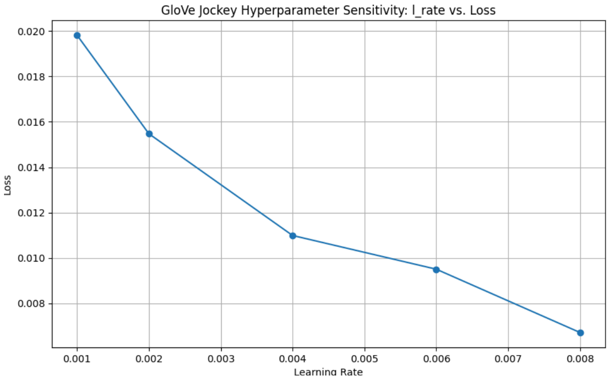
LR and early stopping became very important due to time constraints and the number of embeddings required for the prediction tasks. Thousands of embeddings were created by each date in history to avoid data leakage. Lower learning rates generated better APL results, but needed to be weighed against the incremental improvement gained vs the cost of time. The LR with best incremental improvement given time constraints was 0.01.

Finally, increasing the number of dims improved APL fit. Increasing dimensionality came at the cost of compute. 50 dims per embedding was chosen as compromise between improved fit and time constraints. Even with 8 threads, it took over 2 hours to compute 7 years of horse embeddings.

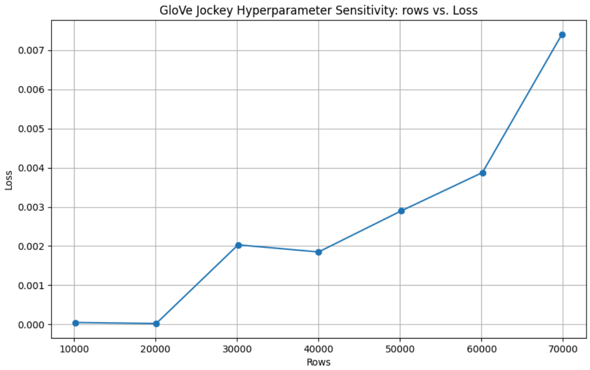
GloVe Embedding Sensitivity Analysis

Average Pair Loss (APL) decreases with the number of dimensions. Intuitively this makes sense because there are more features to capture more nuanced details in the data. There was no significant difference after 75 dims. APL decreased as the learning rate decreased, however, computational expense greatly increased after 0.01 making it cost prohibitive to calculate embeddings for every historical date to avoid data leakage during model training.





APL increased with the size of the training set. This is counter intuitive because the expectation is for APL to decrease with the number of observations. Anecdotally, the average horse retires after 7 years which may cause APL to increase overtime due to variations in horses.



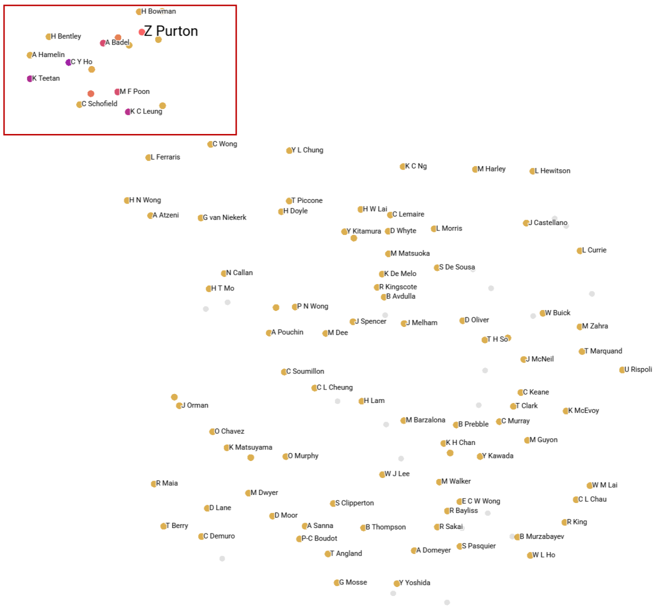
Final GloVe Hyper Parameters: learning_rate = .01, dims = 50 and rows = ~ 70,000 or 7 years of data

Visual Exploration of GloVe Embeddings

To further understand the feature relationships represented within the GloVe inspired embeddings: UMAP, T-SNE and PCA clustering were visually explored using the embedding projection tool https://projector.tensorflow.org/.

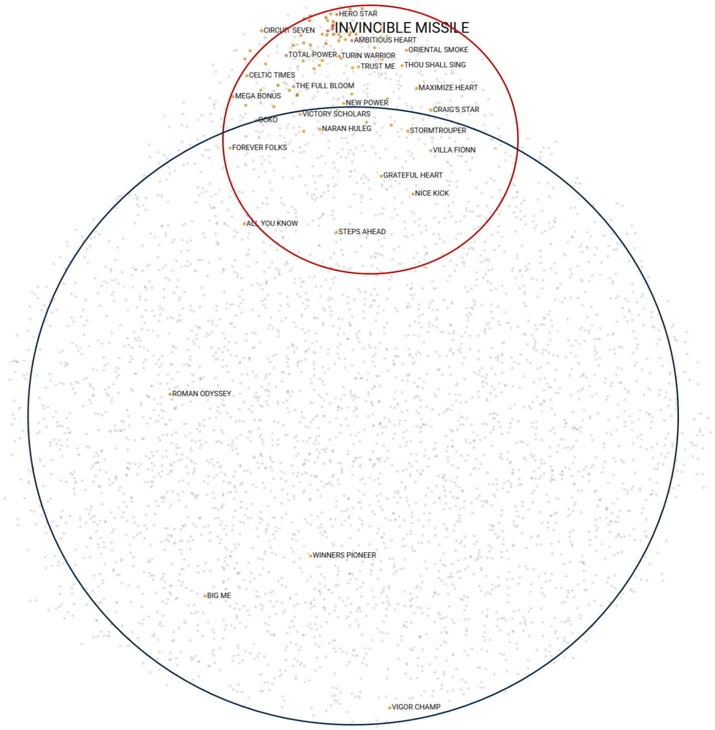
The clustering visualization below was produced by the UMAP algorithm. The clustering revealed a distinct cluster of top performing jockeys. According to the HKJC website, the jockey Z. Purton has the most 1st, 2nd, and 3rd place winnings compared to all other jockeys. Additionally, the jockeys nearest to Z Purton are also in the top 10 list on the HKJC website. This alignment between cluster structure and real-world rankings further validates the quality of the jockey embeddings produced by the GloVe algorithm.

UMAP Representation of Jockey vs Jockey Relationships



The clustering visual below was also produced by the UMAP algorithm. The clustering revealed a distinct cluster of top performing horses (in red). Initially this cluster was difficult to detect because truly high performing horses are so rare. The relationship was only visually detectable after increasing the embedding training data from a couple years of data to 7 years. On average the variation between horses appears to be randomly distributed (most likely gaussian). This is an important feature of the data because it indicates that horse outcomes are evenly and randomly distributed with the exception of a small number of high performing horses. Also, a bias towards high performing horses anecdotally makes sense given the competitive nature of horse racing and financial cost of training a competitive horse.

UMAP Representation of Horse vs Horse Relationships



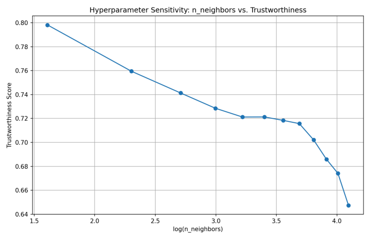
Unsupervised Approach Two | UMAP Embeddings

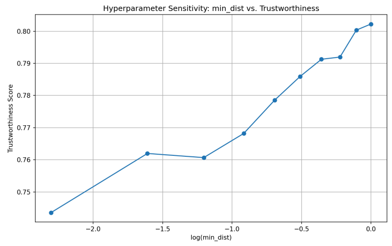
Because UMAP produced: the best visually interpretable local clustering patterns, further reduce high dimensional space, all while preserving non-linear relationships, UMAP was selected to further improve upon GloVe's ability to preserve global context by further compressing its dimensional feature space into local clustering structures within the GloVe embeddings.

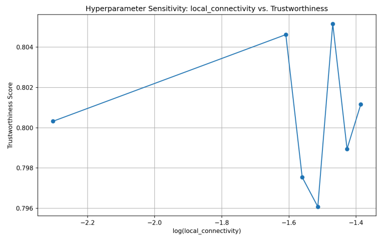
To measure the quality of clusters produced using UMAP, a sensitivity analysis was performed using the Trustworthiness metric on the following parameters: n_neighbors, min_dist, and local_connectivity. Although the trustworthiness metric is a good measure of how much of the original positional relationships were preserved after transformation, it was not the key deciding factor in choosing the final embeddings used for downstream predictive tasks. This is because 5 or less nearest neighbors were required to achieve a score better than 0.80. Decreasing the nearest neighbor parameter this low may weaken UMAP's ability to make effective cluster classifications for predictive tasks.

UMAP Trustworthiness Sensitivity Analysis (Jockey Embeddings):

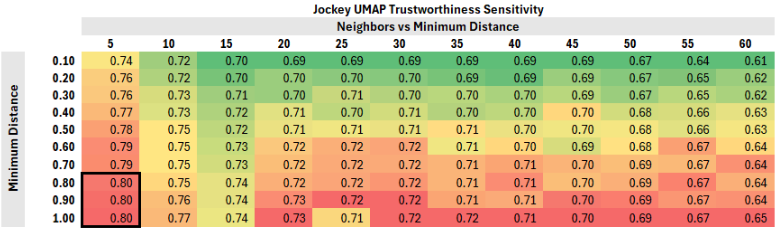
log(n_neighbors) **log(min_dist)** **log(local_connectivity)**







Model trustworthiness was most sensitive to increases in nearest neighbors. The second most sensitive parameter was Minimum Distance. The default for this value was one. Values lower than one negatively impacting trustworthiness.

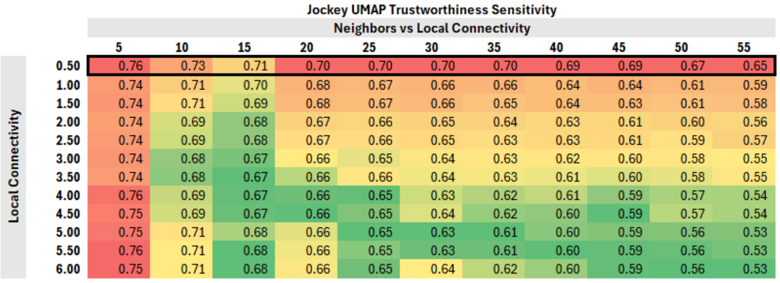


Jockey UMAP Trustworthiness Sensitivity

Neighbors vs Minimum Distance

Minimum Distance	5	10	15	20	25	30	35	40	45	50	55	60
0.10	0.74	0.72	0.70	0.69	0.69	0.69	0.69	0.69	0.69	0.67	0.64	0.61
0.20	0.76	0.72	0.70	0.70	0.70	0.70	0.69	0.69	0.69	0.67	0.65	0.62
0.30	0.76	0.73	0.71	0.70	0.71	0.70	0.70	0.70	0.69	0.67	0.65	0.62
0.40	0.77	0.73	0.72	0.71	0.70	0.71	0.70	0.70	0.70	0.68	0.66	0.63
0.50	0.78	0.75	0.72	0.71	0.71	0.71	0.71	0.70	0.70	0.68	0.66	0.63
0.60	0.79	0.75	0.73	0.72	0.72	0.72	0.71	0.70	0.69	0.68	0.67	0.64
0.70	0.79	0.75	0.73	0.72	0.72	0.72	0.71	0.71	0.70	0.69	0.67	0.64
0.80	0.80	0.75	0.74	0.72	0.72	0.72	0.71	0.71	0.70	0.69	0.67	0.64
0.90	0.80	0.76	0.74	0.73	0.72	0.72	0.71	0.71	0.70	0.69	0.67	0.64
1.00	0.80	0.77	0.74	0.73	0.71	0.72	0.72	0.71	0.70	0.69	0.67	0.65

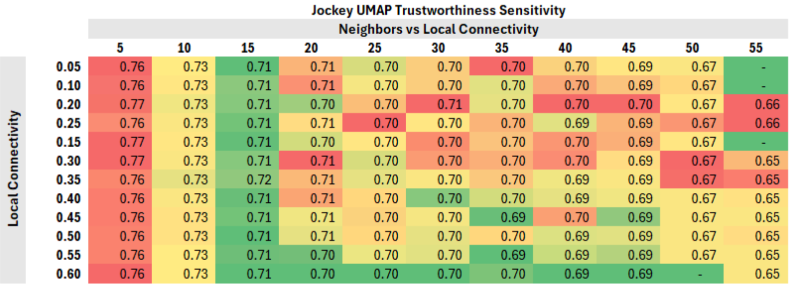
For local connectivity, values higher at one or higher produce strongly connected clusters. Values below 1, reduce connectivity amongst neighbors. Trustworthiness was highest for values between 0.05 and 1.0.



Jockey UMAP Trustworthiness Sensitivity

Neighbors vs Local Connectivity

Local Connectivity	5	10	15	20	25	30	35	40	45	50	55
0.50	0.76	0.73	0.71	0.70	0.70	0.70	0.70	0.69	0.69	0.67	0.65
1.00	0.74	0.71	0.70	0.68	0.67	0.66	0.66	0.64	0.64	0.61	0.59
1.50	0.74	0.71	0.69	0.68	0.67	0.66	0.65	0.64	0.63	0.61	0.58
2.00	0.74	0.69	0.68	0.67	0.66	0.65	0.64	0.63	0.61	0.60	0.56
2.50	0.74	0.69	0.68	0.67	0.66	0.65	0.63	0.63	0.61	0.59	0.57
3.00	0.74	0.68	0.67	0.66	0.65	0.64	0.63	0.62	0.60	0.58	0.55
3.50	0.74	0.68	0.67	0.66	0.66	0.64	0.63	0.61	0.60	0.58	0.55
4.00	0.76	0.69	0.67	0.66	0.65	0.63	0.62	0.61	0.59	0.57	0.54
4.50	0.75	0.69	0.67	0.66	0.65	0.64	0.62	0.60	0.59	0.57	0.54
5.00	0.75	0.71	0.68	0.66	0.65	0.63	0.61	0.60	0.59	0.56	0.53
5.50	0.76	0.71	0.68	0.66	0.65	0.63	0.61	0.60	0.59	0.56	0.53
6.00	0.75	0.71	0.68	0.66	0.65	0.64	0.62	0.60	0.59	0.56	0.53



Jockey UMAP Trustworthiness Sensitivity

Neighbors vs Local Connectivity

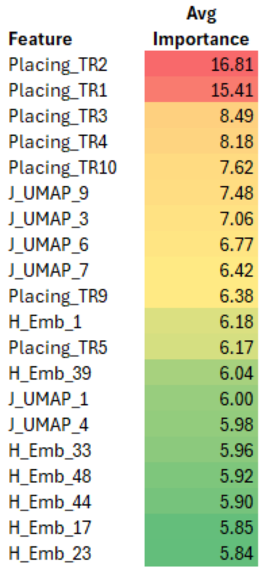
Local Connectivity	5	10	15	20	25	30	35	40	45	50	55
0.05	0.76	0.73	0.71	0.71	0.70	0.70	0.70	0.70	0.69	0.67	-
0.10	0.76	0.73	0.71	0.71	0.70	0.70	0.70	0.70	0.69	0.67	-
0.20	0.77	0.73	0.71	0.70	0.70	0.71	0.70	0.70	0.70	0.67	0.66
0.25	0.76	0.73	0.71	0.71	0.70	0.70	0.70	0.69	0.69	0.67	0.66
0.15	0.77	0.73	0.71	0.70	0.70	0.70	0.70	0.70	0.69	0.67	-
0.30	0.77	0.73	0.71	0.71	0.70	0.70	0.70	0.70	0.69	0.67	0.65
0.35	0.76	0.73	0.72	0.71	0.70	0.70	0.70	0.69	0.69	0.67	0.65
0.40	0.76	0.73	0.71	0.71	0.70	0.70	0.70	0.69	0.69	0.67	0.65
0.45	0.76	0.73	0.71	0.71	0.70	0.70	0.69	0.70	0.69	0.67	0.65
0.50	0.76	0.73	0.71	0.71	0.70	0.70	0.70	0.69	0.69	0.67	0.65
0.55	0.76	0.73	0.71	0.70	0.70	0.70	0.69	0.69	0.69	0.67	0.65
0.60	0.76	0.73	0.71	0.70	0.70	0.70	0.70	0.69	0.69	-	0.65

As a compromise between trustworthiness and clusters large enough for prediction tasks, the following parameters were chosen to create the final embeddings:

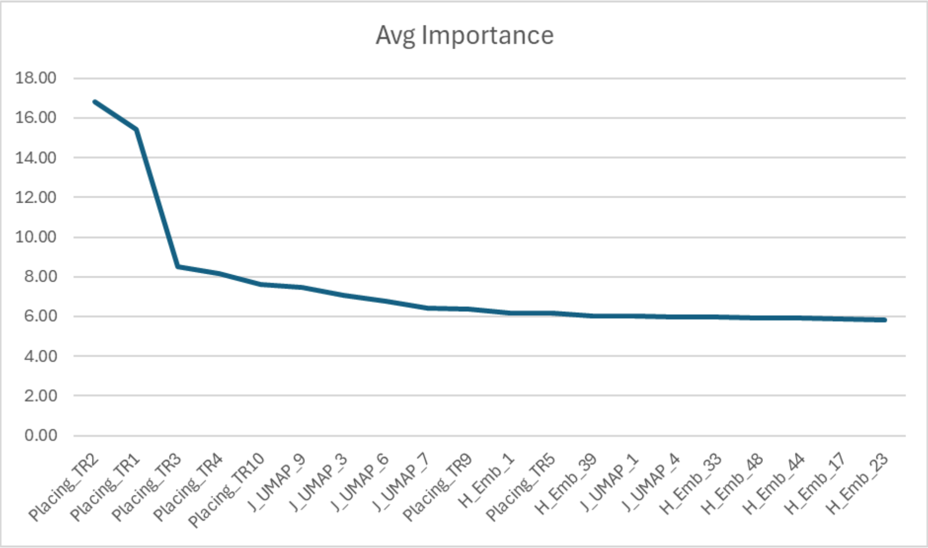
Final Horse Params: n_neighbors = 15, min_dist = 0.50, local_connectivity = 0.75

Final Jockey Params: n_neighbors = 15, min_dist = 0.10, local_connectivity = 0.90

Unsupervised Model Evaluations
Feature Importance Plot Top 20 Features



Feature	Avg Importance
Placing_TR2	16.81
Placing_TR1	15.41
Placing_TR3	8.49
Placing_TR4	8.18
Placing_TR10	7.62
J_UMAP_9	7.48
J_UMAP_3	7.06
J_UMAP_6	6.77
J_UMAP_7	6.42
Placing_TR9	6.38
H_Emb_1	6.18
Placing_TR5	6.17
H_Emb_39	6.04
J_UMAP_1	6.00
J_UMAP_4	5.98
H_Emb_33	5.96
H_Emb_48	5.92
H_Emb_44	5.90
H_Emb_17	5.85
H_Emb_23	5.84

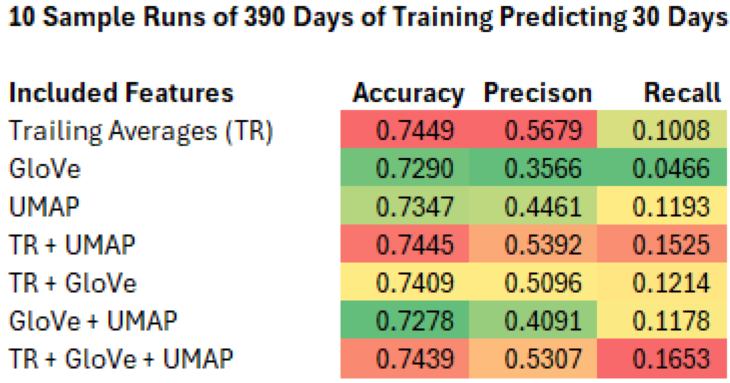


Note: Importance is measured by reduction in loss (importance_type=gain)
J_ = jockey feature, H_ = horse feature, Emb = GloVe feature, UMAP = UMAP feature

Key Findings

- Using all features, model favored trailing averages and jockey UMAP features for most runs
- Top 20 features would vary on each training set, but TR and UMAP were consistently selected more often
- Model did not appear to be overly dependent on any one feature
- No data leakage for TR features, verified in dataframe calculations only to use prior race information for each race date
- Model is trained on 390 days of trailing dates and predicts 30 days of future races (No Leakage)
- Ran a final model using the top 20 features, performance decreased in accuracy, precision, and recall by a couple of points
    - Decrease may be attributed to the model not able to reorder feature importance at for every new training set range

Feature Ablation



10 Sample Runs of 390 Days of Training Predicting 30 Days of Test

Included Features	Accuracy	Precison	Recall
Trailing Averages (TR)	0.7449	0.5679	0.1008
GloVe	0.7290	0.3566	0.0466
UMAP	0.7347	0.4461	0.1193
TR + UMAP	0.7445	0.5392	0.1525
TR + GloVe	0.7409	0.5096	0.1214
GloVe + UMAP	0.7278	0.4091	0.1178
TR + GloVe + UMAP	0.7439	0.5307	0.1653

Key Findings:

- Trailing Averages (TR) of 1 to 10 days alone produced the highest precision
- GloVe features alone performed the worst across all metrics
- UMAP clusters of GloVe embeddings outperforms Glove alone in all metrics
- TR + UMAP underperformed TR alone by 2.5 pts of precision but outperformed TR by 5 pts of recall
- TR + GloVe underperformed TR + UMAP in all metrics
- Glove + UMAP underperforms UMAP alone
- TR + GloVe + UMAP produced the highest Recall and outperformed TR alone by 6.5 pts

Optimal Strategy Recommendation
Depending on the betting strategy, if an individual is risk averse, a GBT model with TR alone is the optimal model because it has the highest precision. GBT + TR may not identifying the most winning horses (recall), but it has the highest certainty (precision) when it does identify a potential top three winner

Parameters used in Feature Importance and Feature Ablation

- 10 sets of 390 training days and 30 test days between the dates of 9/29/2023 and 7/25/2024
- All features evaluated with Gradient Boosted Trees using the following hyper parameters: objective: binary:logistic, max_depth: 5, min_child_weight: 1, gamma: .55, subsample: 0.85, colsample_bytree: 1, learning_rate:0.01, n_estimators:100
- Excluded "Win Odds" from all datasets

Discussion:

Supervised Learning

The model highlighted the effectiveness of combining feature engineering and supervised models for real world prediction tasks like horse race outcomes. A surprising result was how dependent the model was on win odds, with performance dropping significantly when they were removed.

One challenge was handling class imbalance, since only a small proportion of the horses finish in the top three. This was addressed through careful metric selection and optimizing the model focusing on precision.

Given additional time and resources, the approach could be modified to tune the weight of specific features such as recent results, to reduce the model's dependency on win odds. Or, in contrast, it could be further extended by incorporating real-time betting market dynamics, which might provide valuable insights into public sentiment and betting behaviors, potentially further improving predictive accuracy.

Unsupervised Learning

The objective was to encode relational information between horses, jockeys, and their competitors to improve predictive power by providing downstream supervised models: global race context (GloVe), local race context (UMAP), and reduced dimensionality.

After model evaluation, GloVe and UMAP embeddings both outperformed the raw data features, but unexpectedly failed to outperform simple trailing averages. The local context produced by UMAP embeddings outperformed the global context of GloVe and combining the two negatively impacted all prediction metrics. As an upside, combining embeddings and trailing averages increased overall recall at the cost of some precision.

The embedding performance was less impactful than anticipated. Perhaps this is due to the embeddings being optimized based on loss and trustworthiness metrics. Given more time and resources, the embeddings should be optimized for precision and recall. The challenge is producing embedding for every historical race date (2,555 sets over 7 years). Due to compute and time constraints embeddings were calculated over longer time ranges between 7 days and 8 weeks (opposed to every historical date). For reference, A single horse GloVe embedding takes over 12 minutes to compute.

As a final thought, a possible model improvement could be the addition of co-occurrence matrices to extract features from the "Win Odds" feature. The "Win Odds" feature alone performed higher than any of the embedding features. Using embedding techniques, it may be possible to decompose the crowd intuition embedded in the "Win Odds" feature to further improve predictive power.

Ethical Considerations

Supervised Learning Ethical Considerations

1. Risk in promoting gambling: a predictor model can be mis-interpreted as a sure-win tool, encouraging excessive or irresponsible betting. The danger is particularly acute for bettors with addictive tendencies, PTSD, or constrained finances, who may rationalize betting as a "data-driven" investment and develop problem gambling. Mitigation: embed prominent responsible-gambling disclaimers in the model. Also, we may incorporate cooling-off timers to limit the usage of the model and thus the betting behavior.

2. Bias, Fairness & Secondary Use: if the model were ever repurposed for secondary use (e.g., jockey recruitment) it could unjustly favour certain gender, ethnicities or socioeconomic backgrounds if these information are used to train the model. Mitigation: prohibiting secondary uses such as talent screening without a fairness audit. Besides, the model could be retrained with a deliberate removal of such features to reduce the possibility of bias.

Unsupervised Learning Ethical Considerations

3. Interpretability & Transparency: embeddings are often opaque, making it hard for users to understand why a horse is assigned to a given cluster; this opacity can conceal data bias and render model review extremely difficult. Mitigation: document the complete model logic—including the embedding architecture, the role and purpose of key hyper-parameters, and the reasoning behind cluster formation—so that stakeholders can trace and understand the model's decisions.

4. Animal-welfare concerns: opaque clusters could be used to single out "high-potential" horses that seem able to endure harsher training schedules, thereby increasing injury risk and harming animal welfare. Mitigation: embed a welfare-impact statement in the model, making it clear that cluster insights must never override veterinary guidance. In addition, restrict access to raw horse-level outputs so trainers cannot identify specific animals to hyper-train.

Statement of Work:

Task	**Lead**	**Reviewer(s)**
Formation and Title	All	All
Draft Proposal	All	All
Data Collecting and Cleaning	Simon	Austin, Nathaniel
Data Exploration	Simon	Austin, Nathaniel
Unsupervised Learning	Austin	Simon, Nathaniel
Supervised Learning	Nathaniel	Austin, Simon
Visualizations	Nathaniel / Austin	Austin, Simon, Nathaniel
Project Write-Up	All	All

References:

Deepanshusachdeva (2023, June 23). *Implementing GloVe from scratch — Word Embedding for Transformers. Medium.* Retrieved from https://medium.com/nerd-for-tech/implementing-glove-from-scratch-word-embedding-for-transformers-95503138d65

Olamendy, J. C. (2023, October 31). *Measuring similarity between two input entities using embeddings: A deep dive. Medium*. Retrieved from https://medium.com/@juanc.olamendy/measuring-similarity-between-two-input-entities-using-embeddings-a-deep-dive-912a24296861

Frenzel, C. (2021, August 13). On the validation of UMAP embeddings: A practical example of validating UMAP embeddings in Amazon DenseClus. Medium. Retrieved from https://medium.com/data-science/on-the-validating-umap-embeddings-2c8907588175

Appendix:

Appendix 1. Data Scraped from HKJC and called from HKO API, with simple one hot/multi-hot encoding and flattening of race, horse, jockey, gear and competitor features.

Feature Name	**Description**	**Type**
Placing	Finishing position of the horse in the race	Integer
Horse No.	Official number assigned to the horse	Integer
Horse	Name of the horse	String
Jockey	Name of the jockey	String
Trainer	Name of the trainer	String
Act. Wt.	Actual weight carried by the horse	Float
Declar. Horse Wt.	Declared body weight of the horse	Float
Dr.	Barrier draw (starting gate position)	Integer
LBW	Lengths behind the winner	Float
Win Odds	Official win odds for the horse	Float
Date	Date of the race	Date
Course	Racecourse code (e.g., ST, HV)	String
RaceNumber	Race number on the race date	Integer
Race type	Class or type of race	String
DistanceMeter	Race distance in meters	Integer
Score range	The score ranges of competing horses in that race	String
MinScore	Lower bound of the score range	Float

MaxScore	Upper bound of the score range	Float
Going	Track condition (e.g., GOOD)	String
Handicap	Handicap rating for the race	Binary
Course Detail	Track/course configuration	String
Time 1	Split or call time at the first section	Time
Time 2	Split time at the second section	Time
Time 3	Split time at the third section	Time
Time 4	Split time at the fourth section	Time
Time 5	Split time at the fifth section	Time
Time 6	Split time at the sixth section	Time
Sectional Time 1	Official sectional for first segment	Time
Sectional Time 2	Official sectional for second segment	Time
Sectional Time 3	Official sectional for third segment	Time
Sectional Time 4	Official sectional for fourth segment	Time
Sectional Time 5	Official sectional for fifth segment	Time
Sectional Time 6	Official sectional for sixth segment	Time
Running Position	Positions throughout the race (e.g., "4 4 4 1")	String
RunningPosition1	Position at first call	Integer
RunningPosition2	Position at second call	Integer
RunningPosition3	Position at third call	Integer
RunningPosition4	Position at fourth call	Integer
RunningPosition5	Position at fifth call	Integer

RunningPosition6	Position at sixth call	Integer
Finish Time	Final time of the race	Time
HorseIndex	Unique identifier code for the horse	String
Country	Country of origin of the horse	String
Age at scraping	Horse age recorded when data scraped	Float
Age at race	Horse age on the race date	Float
Colour	Horse coat colour	String
Sex	Horse sex (e.g., Gelding)	String
Import Type	Type of import (e.g., PP)	String
Season Stakes	Stake winnings for the current season	Float
Total Stakes	Career stake winnings	Float
No. of 1-2-3-Starts	Count of starts finishing 1st–3rd	Integer
No. of starts in past 10 race meetings	Starts within last ten meetings	Integer
Current Stable Location (Arrival Date)	Stable location with arrival date	String
Import Date	Date horse was imported	Date
Owner	Owner(s) name	String
Current Rating	Current official rating	Float
Last Rating For Retired	Last rating before retirement	Float
Start of Season Rating	Rating at season start	Float
Sire	Name of the sire	String
Dam	Name of the dam	String

Dam's Sire	Name of the dam's sire	String
Same Sire	Other horses with the same sire	String
PP Pre-import races footage	Number of pre-import races on film	Binary
Gear	Combined description of gear carried	String
Comment	Stewards’ or race comment about run	String
B	Blinker in use (1=yes, 0=no)	Binary
B_first_time	First time Blinkers worn (1=yes, 0=no)	Binary
B_replaced	Blinkers replaced (1=yes, 0=no)	Binary
B_removed	Blinkers removed (1=yes, 0=no)	Binary
BO	Blinker with one cowl only in use	Binary
BO_first_time	First time BO worn	Binary
BO_replaced	BO replaced by other gear	Binary
BO_removed	BO removed	Binary
CC	Cornell Collar in use	Binary
CC_first_time	First time Cornell Collar worn	Binary
CC_replaced	Cornell Collar replaced	Binary
CC_removed	Cornell Collar removed	Binary
CP	Sheepskin Cheek Pieces in use	Binary
CP_first_time	First time Sheepskin Cheek Pieces worn	Binary
CP_replaced	Sheepskin Cheek Pieces replaced	Binary
CP_removed	Sheepskin Cheek Pieces removed	Binary

CO	Sheepskin Cheek Piece on one side	Binary
CO_first_time	First time CO worn	Binary
CO_replaced	CO replaced	Binary
CO_removed	CO removed	Binary
E	Ear Plugs in use	Binary
E_first_time	First time Ear Plugs worn	Binary
E_replaced	Ear Plugs replaced	Binary
E_removed	Ear Plugs removed	Binary
H	Hood in use	Binary
H_first_time	First time Hood worn	Binary
H_replaced	Hood replaced	Binary
H_removed	Hood removed	Binary
P	Pacifier in use	Binary
P_first_time	First time Pacifier worn	Binary
P_replaced	Pacifier replaced	Binary
P_removed	Pacifier removed	Binary
PC	Pacifier with cowls in use	Binary
PC_first_time	First time PC worn	Binary
PC_replaced	PC replaced	Binary
PC_removed	PC removed	Binary
PS	Pacifier with one cowl in use	Binary
PS_first_time	First time PS worn	Binary

PS_replaced	PS replaced	Binary
PS_removed	PS removed	Binary
SB	Sheepskin Browband in use	Binary
SB_first_time	First time SB worn	Binary
SB_replaced	SB replaced	Binary
SB_removed	SB removed	Binary
SR	Shadow Roll in use	Binary
SR_first_time	First time SR worn	Binary
SR_replaced	SR replaced	Binary
SR_removed	SR removed	Binary
TT	Tongue Tie in use	Binary
TT_first_time	First time Tongue Tie worn	Binary
TT_replaced	Tongue Tie replaced	Binary
TT_removed	Tongue Tie removed	Binary
V	Visor in use	Binary
V_first_time	First time Visor worn	Binary
V_replaced	Visor replaced	Binary
V_removed	Visor removed	Binary
VO	Visor with one cowl in use	Binary
VO_first_time	First time VO worn	Binary
VO_replaced	VO replaced	Binary
VO_removed	VO removed	Binary

XB	Crossed Nose Band in use	Binary
XB_first_time	First time XB worn	Binary
XB_replaced	XB replaced	Binary
XB_removed	XB removed	Binary
HorseCompetitor1	Index of first competing horse	String
HorseCompetitor2	Index of second competing horse	String
HorseCompetitor3	Index of third competing horse	String
HorseCompetitor4	Index of fourth competing horse	String
HorseCompetitor5	Index of fifth competing horse	String
HorseCompetitor6	Index of sixth competing horse	String
HorseCompetitor7	Index of seventh competing horse	String
HorseCompetitor8	Index of eighth competing horse	String
HorseCompetitor9	Index of ninth competing horse	String
HorseCompetitor10	Index of tenth competing horse	String
HorseCompetitor11	Index of eleventh competing horse	String
HorseCompetitor12	Index of twelfth competing horse	String
HorseCompetitor13	Index of thirteenth competing horse	String
JockeyCompetitor1	Name of first competing jockey	String
JockeyCompetitor2	Name of second competing jockey	String
JockeyCompetitor3	Name of third competing jockey	String
JockeyCompetitor4	Name of fourth competing jockey	String
JockeyCompetitor5	Name of fifth competing jockey	String

JockeyCompetitor6	Name of sixth competing jockey	String
JockeyCompetitor7	Name of seventh competing jockey	String
JockeyCompetitor8	Name of eighth competing jockey	String
JockeyCompetitor9	Name of ninth competing jockey	String
JockeyCompetitor10	Name of tenth competing jockey	String
JockeyCompetitor11	Name of eleventh competing jockey	String
JockeyCompetitor12	Name of twelfth competing jockey	String
JockeyCompetitor13	Name of thirteenth competing jockey	String
CourseName	Racecourse name (ShaTin or HappyValley)	String
MeanTemperature	Mean air temperature on race day	Float
MaxTemperature	Maximum temperature recorded	Float
MinTemperature	Minimum temperature recorded	Float
MaximumRelativeHumidity	Highest relative humidity	Float
MinimumRelativeHumidity	Lowest relative humidity	Float
GrassMinimumTemperature	Minimum temperature on turf	Float
Rainfall(mm)	Daily rainfall in millimetres	Float
MeanUVIndex	Mean UV index	Float
MaxUVIndex	Maximum UV index	Float

Appendix 2. Data Schema of features on the final dataset

382 Features evaluated

Feature	**Description**
Effective Date	The training set evaluation date
top_three	Binary top three placing flag
Placing	The finishing place of horse
Horse No.	Horse number in race
Horse	Horse Name
Jockey	Jockey Name
Trainer	Trainer name
Act. Wt.	Jockey weight
Declar. Horse Wt.	Horse weight
Dr.	The gate the horse starts the race at
Win Odds	The pay odds for each horse in race
Date	Date of race
Course	Course of race
RaceNumber	Race number of race
Race type	Type of race
DistanceMeter	Distance of race in meters
Score range	Score range of race
MinScore	Minimum score of race
MaxScore	Maximum score of race

Going	Track condition of race
Handicap	Handicap information of race
Course Detail	Details of racecourse
Finish Time	Finish time of horse
HorseIndex	Index of horse
Country	Country of horse
Colour	Colour of horse
Sex	Sex of horse
Import Type	Type of horse import
Import Date	Import date of horse
Owner	Owner of horse
Sire	Sire of horse, i.e., mother
Dam	Dam of horse, i.e., father
Dam’s Sire	Sire of horse’s dam
Same Sire	Indicator of same sire
PP Pre-import races footage	Pre-import race footage
Gear	Gear used by horse (e.g., blinders, shoes)
[Gear]_first_time/replaced/removed	First time, replaced, or removed specific gear
HorseCompetitor[1-13]	Competitor horse identifiers in the same race
JockeyCompetitor[1-13]	Competitor jockey identifiers in the same race
DistanceMeterAsStr	Distance of race as string

[Score range] Placing Value TR[1-5]	Placing values within specific score ranges
[DistanceMeterAsStr] Placing Value TR[1-5]	Placing values within specific distances
Placing_TR[1-10]	Historical placing values
H_Emb_[0-49]	Glove Horse Embedding Features 1-50
H_UMAP_[0-9]	UMAP Horse Embedding Features 1-10
J_Emb_[0-49]	Glove Jockey Embedding Features 1-50
J_UMAP_[0-9]	UMAP Jockey Embedding Features 1-10

Appendix 3. Explanation of files used to produce report

382 Features evaluated

Explanation of Files

File	**Description**	**Section**
race-trailing-stats.py	Creates trailing average racing features	Unsupervised
win_odds_model.py	Used to measure accuracy, preciion, and recall for an "Odds" only model	Unsupervised
co_occurrence_embeddings.py	Create co-occurrence matrices for final training sets, also creates values for hyper parameter sensitivity testing.	Unsupervised
flatten_embedding_data.py	Flattens co-occurrence matrices into one embedding per race date. Files are 12Gbs plus. This step is required to join embeddings into the final training sets without causing an out of memory error due to cross joining	Unsupervised
umap_embeddings.py	Creates UMAP embeddings from GloVe embeddings, supporting functions for sensitivity charts and tables	Unsupervised
merge_umap_embeddings.py	Merges UMAP embeddings into final training sets	Unsupervised
query_repository.py	Repository of queries used to build final training sets	Unsupervised
merge_all_data_for_model.py	Use the results from race-trailing-stats.py, co_occurrence_embeddings.py, flatten_embedding_data.py, merge_umap_embeddings.py, and query_repository.py to create final training sets	Unsupervised
xgboost_GloVe_UMAP_test.py	Used to produce feature importance and ablation analysis for report	Unsupervised
visuals and tables used in presentation.xlsx	Used to create some of the visualtions in the unsupervised portion of the project	Unsupervised
failure_analysis	- identifies false negatives and false positives	Supervised
learning_curve	- generates learning curve for training data	Supervised
logistic_notrandemb	- logistic regression model with no trailing stats or embeddings just race features	Supervised
logistic_withtrandemb	- logistic regression model with trailing stats, embeddings, and race features	Supervised
naive_models	- random choice model and dummy variable	Supervised
param_search	- finding hyperparameters that optimize the xgboost model for precision	Supervised
sensitivity_analysis	- graphing the sensitivity of the xgboost model to varying hyperparameters	Supervised
xg_boost_everything	- xgboost model with trailing stats, embeddings, and race features	Supervised
xg_boost_noemb_notr	- xgboost model with only race features	Supervised
xg_boost_noemb_yestr	- xgboost model with trailing stats and race features and no embeddings	Supervised
xg_boost_precision_optimized	- xgboost model with hyperparameters optimized for precision	Supervised
xg_boost_yesemb_notr	- xgboost model with embeddings and race features and no trailing stats	Supervised
1_race_data_scraping	Scrape race data from website and store in csv	Data Extraction
2_horse_data_scraping	Scrape race data from website and store in csv	Data Extraction
3_race_comment_scraping	Scrape race data from website and store in csv	Data Extraction
4_dataset_cleaning	Scrape race data from website and store in csv	Data Extraction
5_race_start_time	Scrape race data from website and store in csv	Data Extraction
6_weather_data	Scrape race data from website and store in csv	Data Extraction
7_weather_API	Use weather api to collect weather start per race date and store in csv	Data Extraction
jockey-data-scraping	Scrape aggregate jockey stats and store in csv	Data Extraction