Predicting Horse Racing Results with Machine Learning

LYU 1703
LIU YIDE 1155062194

Supervisor:
Professor Michael R. Lyu

Outline

- Recap of last semester
- Object of this semester
- Data Preparation
- Set to sequence framework
- BN network
- Rank network

1

Recap of last semester

Recap of last semester

Methodology

- Compare different approaches
  - Horse win or lose (Binary classification)
  - Horse rank (Multi-class classification)
  - Horse finishing time (Regression)
- Use regression to predict the horse finishing time individually

Recap of last semester

Feature Engineering

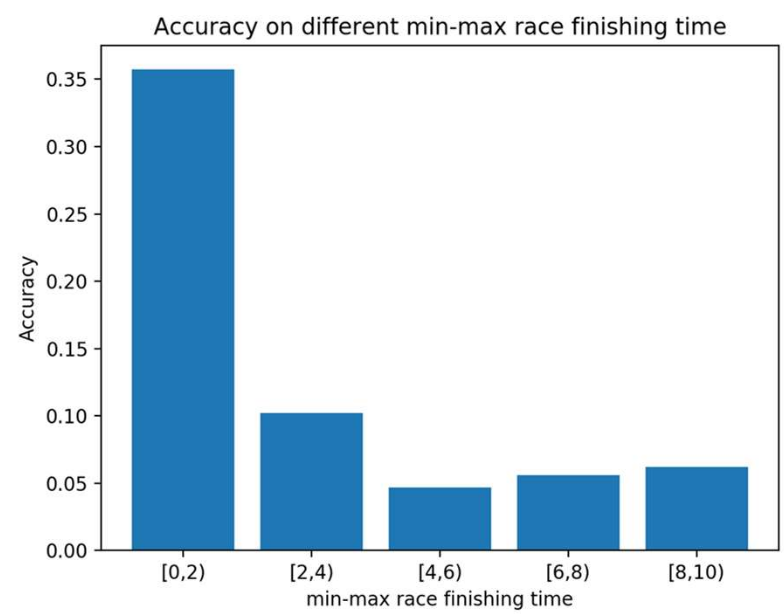
- Construct a racing record and weather database
- Evaluate a number of conditions
  - Divide and Conquer on location (‘ST’ and ‘HV’)
  - Augmented weather data
  - Win odds

<u>Result:</u> Location partition and additional features is useful in prediction

Recap of last semester

Issue

- Our model predict the horse finishing time individually
- However, horse racing concerns "group"
- Worse Performance on inconsistent race prediction



Question:
Can learning help horse racing prediction?

YES

But we need to resolves the reported issue, and bet with strategies

2 Objective

Objective

Objective

- Make use of additional data
- Predict consistent finishing time within each race
- Improve bet (WIN & PLACE) accuracy
- Explore actual bets

3 Date Preparation

Data Preparation

Additional Horse Dataset

- Last semester: useful additional information can improve the prediction results and bet accuracy
- Following this idea, we retrieve complete horse data on the HKJC website

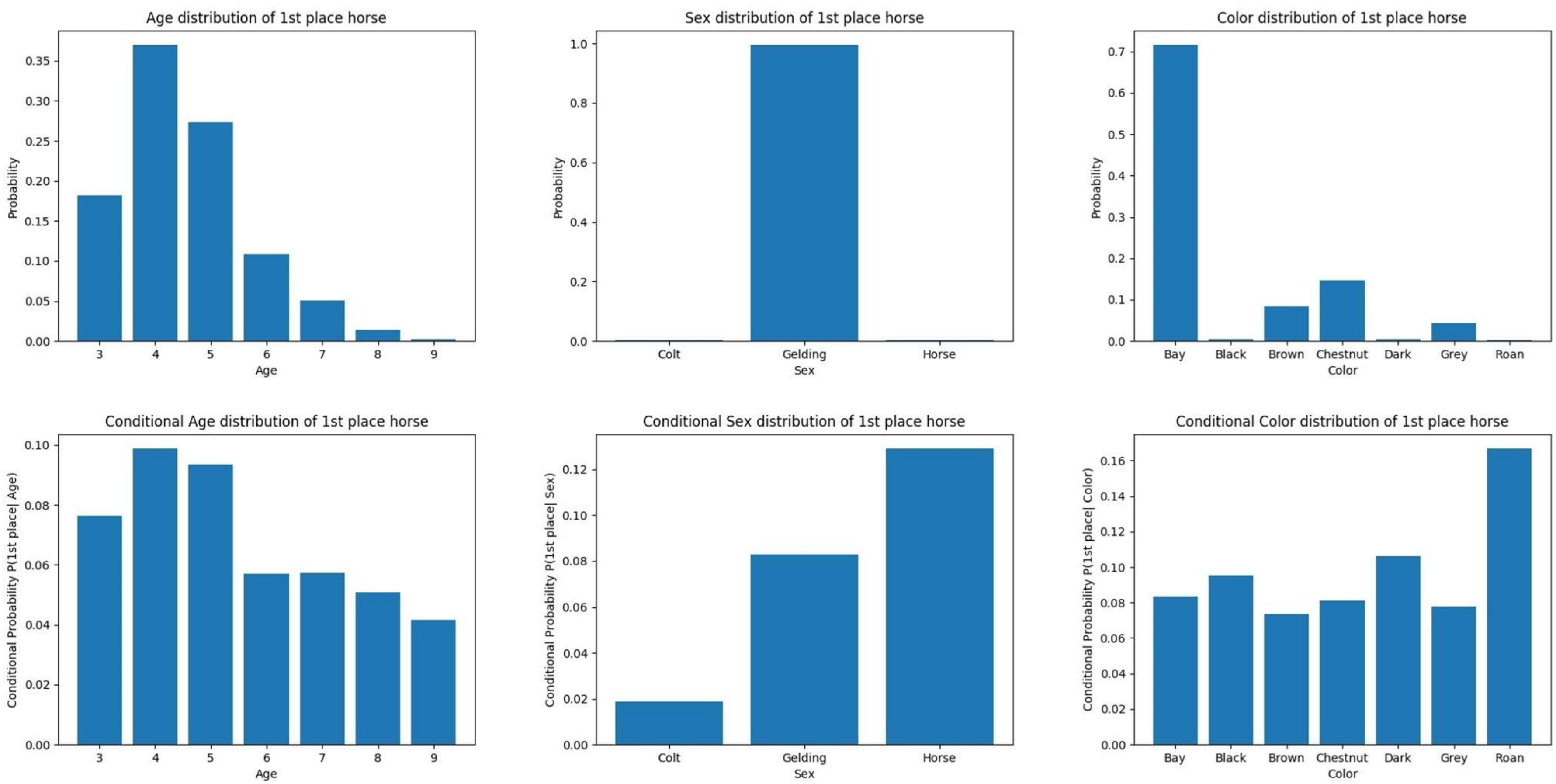
Data Preparation

Additional Horse Dataset

- The horse dataset contains 7 essential features to distinguish the horses
- This information is another golden criteria to assess the horse performance

Features	Meanings
Horseid	Unique Identifier
Origin	Place of Birth
Birth	Birth Date
Color	Fur Color
Sex	Horse Gender
Sire	Father
Dam	Mother
Dam’s Sire	Maternal Grandfather

Data Preparation



Data Preparation

Embedding network

- Learn the similarity between difference instance - cosine distance

- Mapping network f: X ->Y,
  - X: input; Y feature vector to learn

- Easy to use t-SNE to visualize the data (for future research on game selections)

4 Set to Sequence Framework

Set to Sequence Framework

Motivation

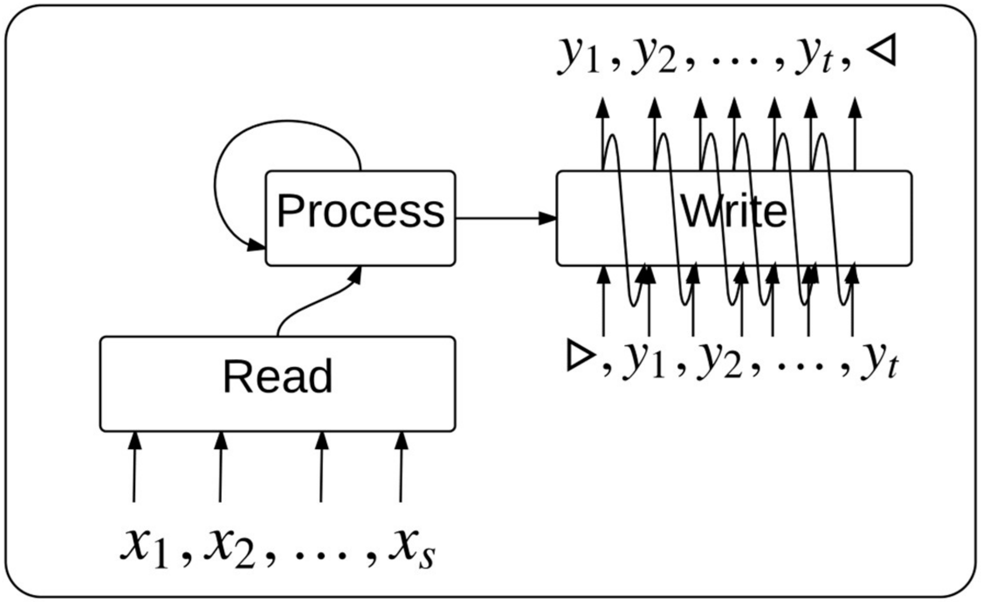
- Try use RNN to learn from horse records - set data
  - The race is grouped in training and prediction

- The set2seq framework is first developed by Vinyals (2015)

Set to Sequence Framework

Mechanism

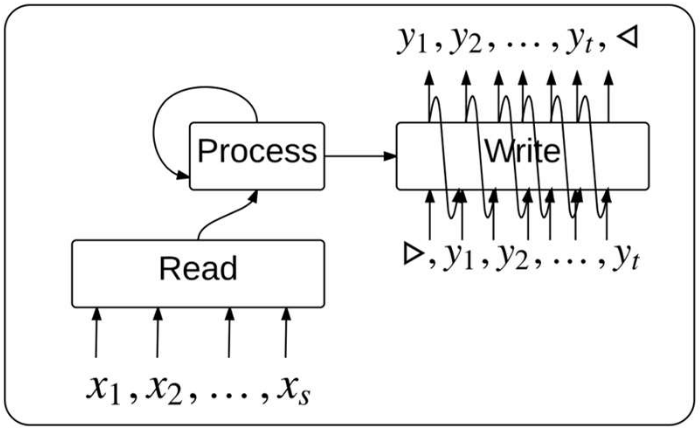
- Read: Embedding network
- Process: LSTM (A RNN unit) with attention mechanism
- Write: Pointer Network



Set to Sequence Framework

Process Module

- Attention mechanism
- Associated memory
  a. Activation function over the input m_i and output q_t.
  b. Softmax to calculate importance of each input to output
  c. Generate a context vector r_t (importance array) attached to the input.



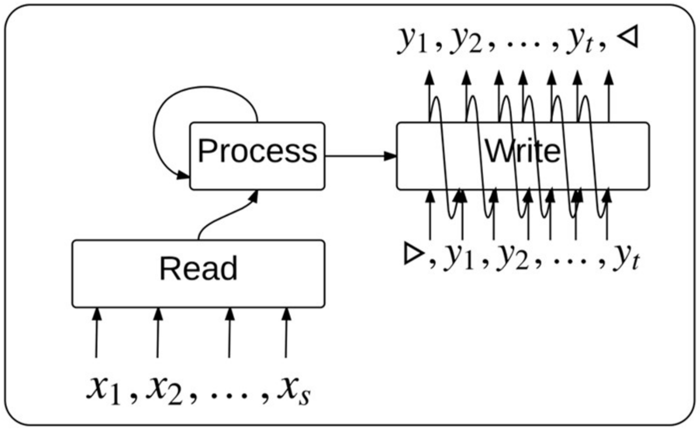
$$
\begin{aligned}
q_t &= LSTM(q^*_{t-1}) \\
e_{i,t} &= f(m_i, q_t) \\
a_{i,t} &= \frac{\exp(e_{i,t})}{\sum_j \exp(e_{j,t})} \\
r_t &= \sum_i a_{i,t} m_i \\
q^*_t &= [q_t \; r_t]
\end{aligned}
$$

Set to Sequence Framework

Write Module (decoder)

- Pointer Network
- Soft pointer pointing the most impossible horse (with maximum likelihood)
  - Softmax function over process units
  - Can have duplicated output

$$\sigma(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$$



Set to Sequence Framework

Two Experiments

- Implement the framework with Keras
- Races with 12 horses. Horses ordered by horse no.
- Goal:
    a. Point the horses in correct sequence directly
    b. Learn the finishing time race by race (Write module becomes normal LSTM)

Set to Sequence Framework

Results & Discussion

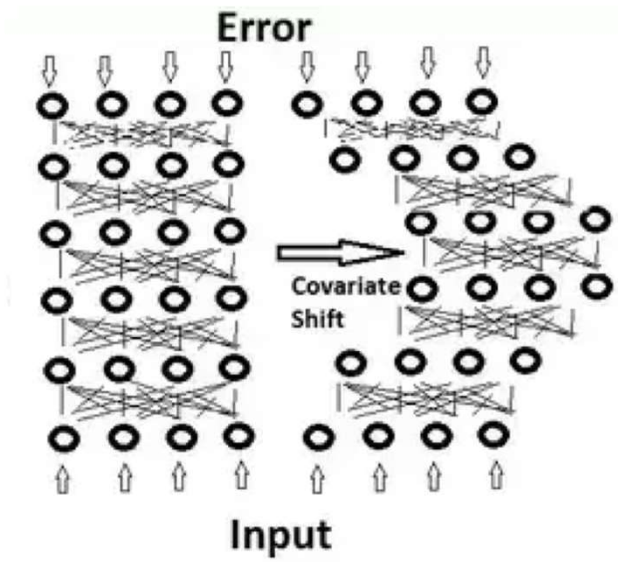
- All experiments fail
  - 1st model output duplicate horses (entries) - cannot interpret as rankings
  - 2nd model output the finishing time following the horse no.
- Although the framework works in simple cases i.e. sorting number
- We claim that the model cannot learn from complex data such as race records

5 BN Model

BN Model

Motivation - Focus on race aspects

- Covariate shift
  - cause the race finishing time inconsistency
- Though input data is normalized, layer output distribution deviates from zero mean,unit variance
- The effect gets extraneous when multiple layers stack up
- Propagation to the result



BN Model

Batch Normalization

- Idea: normalize the input of each layer (output of the former layer)
- The BN transforms with the normalized output from last layer
- In our model, We insert the layer after each dense layer

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1\ldots m}\}$;
Parameters to be learned: γ, β

Output: $\{y_i = \text{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_\mathcal{B} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma_\mathcal{B}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_\mathcal{B})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_\mathcal{B}}{\sqrt{\sigma_\mathcal{B}^2 + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma \widehat{x}_i + \beta \equiv \text{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

BN Model

Experiment & Results

- BN model predict the finishing time on the same data as last semester
- Results (and comparisons)are shown in the following:

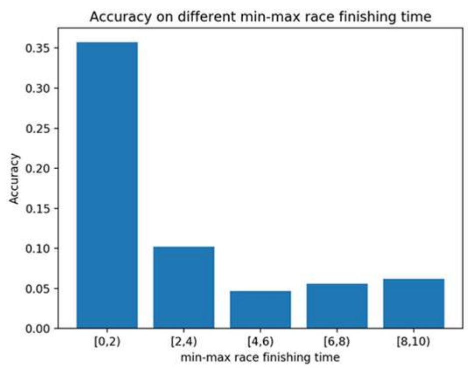
Models	Random	Odds Based	Old Model	BN Model
MSE	-	-	417.7	3.68
MAE	-	-	18.43	1.42
Accuracy_win	0.083(1/12)	0.273	0.107	0.244
Accuracy_place	0.25(3/12)	0.558	0.314	0.489
Net gain	-	-1754/-1792	-568/-1285	-1284/-1221
Return/Bet	-	-21%/-22%	-	-15%/-15%

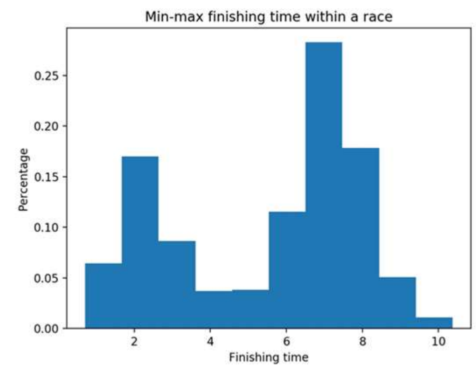
Models	Random	Odds Based	Old Model	BN Model
MSE	-	-	417.7	3.68
MAE	-	-	18.43	1.42
Accuracy_win	0.083(1/12)	0.273	0.107	0.244
Accuracy_place	0.25(3/12)	0.558	0.314	0.489
Net gain	-	-1754/-1792	-568/-1285	-1284/-1221
Return/Bet	-	-21%/-22%	-	-15%/-15%

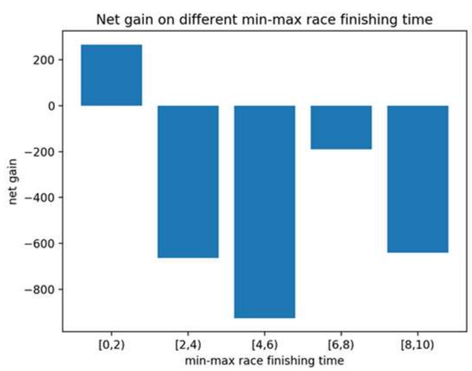
- BN model predict the finishing time in a race aspect automatically
- Large increase in WIN/PLACE accuracy
- Net gain better than public intelligence (note[1])
- Claim: consistent time distribution boost the performance of the model

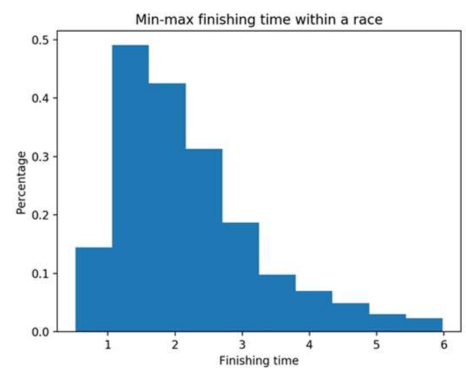
BN Model

Analysis on our Claim









6 Rank model

Rank Model

Motivation

- BN model learns the consistent finishing time distribution of each model.
- Yet the regression methodology under BN model has a major drawback:
  - Learn the finishing time distribution
  - Cannot inference the relative rankings between horses (ultimate goal)

Rank Model

Rank model

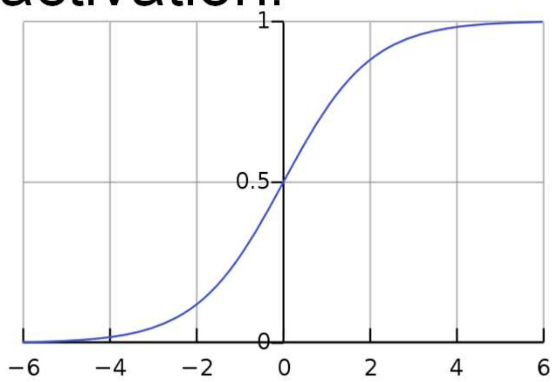
- To learn the ultimate goal, we propose to learn the ranking by pair:
- Consider the BN model as a nonlinear network function f: $R^d \rightarrow R$, which map the input features to the finishing time.
- Ranking is defined using f(x).
- $f(x_i)<f(x_j)$: x_i is faster than x_j, denoted as $x_i < x_j$

Rank Model

Rank model

- In this task, we aim to learn $P(x_i < x_j)$, which can be approximate by ε_{ij}, the difference of horse finishing time
- To learn $P(x_i < x_j)$, we establish a trainable bound using the sigmoid activation.

$$S(t) = \frac{1}{1+e^{-t}}.$$



Rank Model

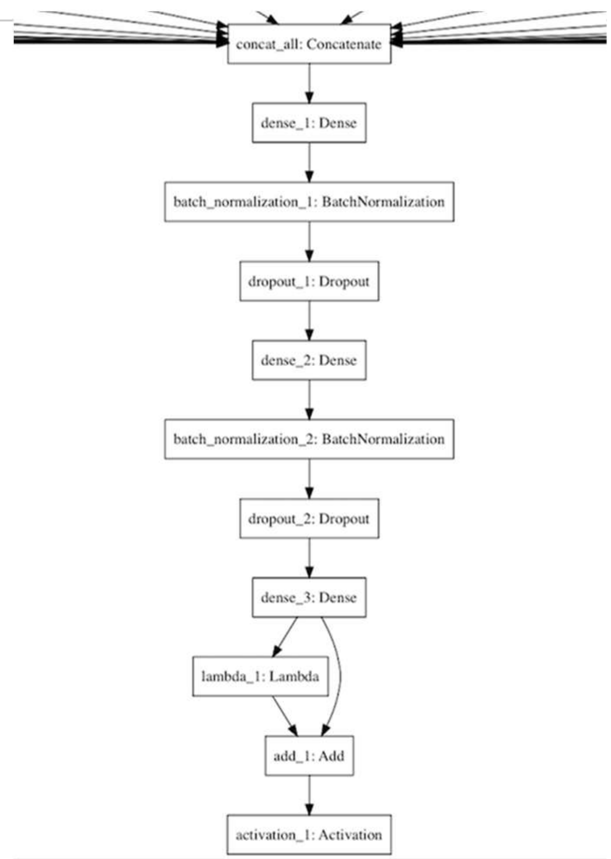
Rank model

- For simplicity, we define the $z_{ij} \equiv f(x_i)-f(x_j)$ to be the difference in finishing time of horse x_i, x_j
- Our model learns the following:

$$P_{ij} = \frac{e^{z}_{ij}}{1+e^{z}_{ij}}$$

- Then the loss (cross entropy) be $\Delta_{ij} \equiv \varepsilon_{ij} * -log(sigmoid(P_{ij})) + (1-\varepsilon_{ij}) * -log(1-sigmoid(P_{ij}))$

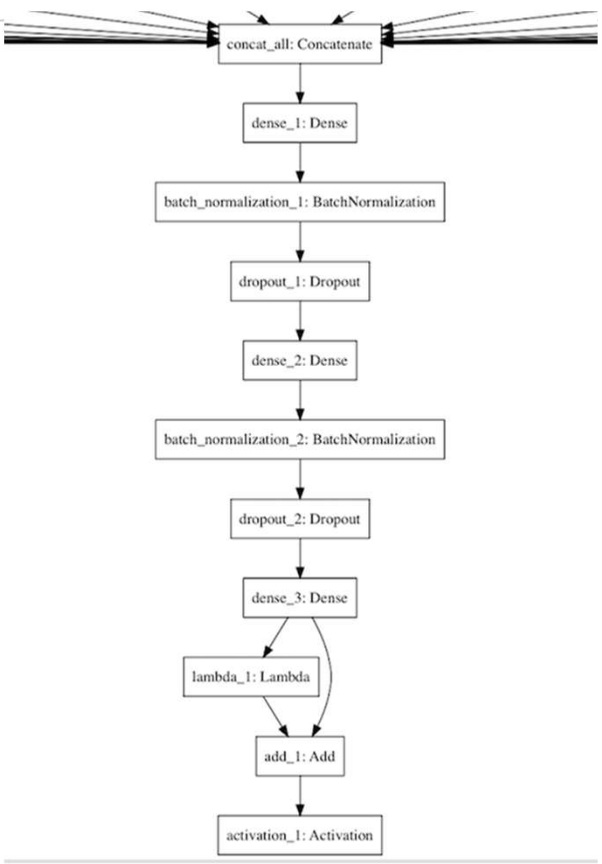
where ε_{ij} is also after activation



Rank Model

Rank model

- During training, we freeze the bottom layers and only train on the topest layer
  - Maintain the finishing time distribution
  - Learn the minor difference
- In prediction, we only extract the output of the BN model part (finishing time)



Rank Model

Experiments & Results`

Models	Odds Based	BN Model	Rank Model[1]
Accuracy_win	0.273	0.244	0.305
Accuracy_place	0.558	0.489	0.521
Net gain	-1754/-1792	-1284/-1221	181.5/-124.5
Return/Bet	-21%/-22%	-0.15%/-0.15%	17%/-12%

- Rank model outperforms BN model
- Positive net gain on WIN bet

Rank Model

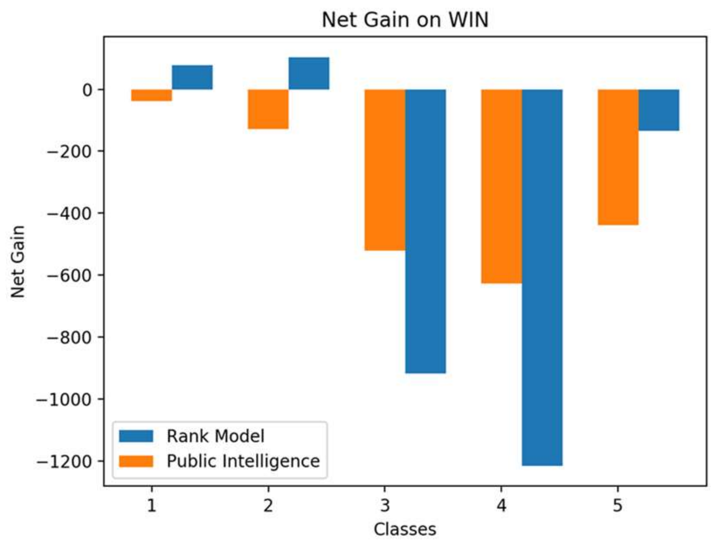
A finding

Class	1	2	3	4	5
Accuracy_win	0.5625	0.409	0.2355	0.1904	0.2457
Accuracy_place	0.625	0.6363	0.5181	0.492	0.5084
Net gain (WIN)	78	103.5	-918.5	-1216	-133.5
Return/Bet (WIN)	0.43	0.12	-0.33	-0.38	-0.11
Net gain (PLACE)	-27.9	-96.6	-706.3	-829.2	-187.5
Return/Bet (PLACE)	-0.17	-0.11	-0.25	-0.26	-0.15

Model works better on higher class (class 1 and 2)

Rank Model

A finding

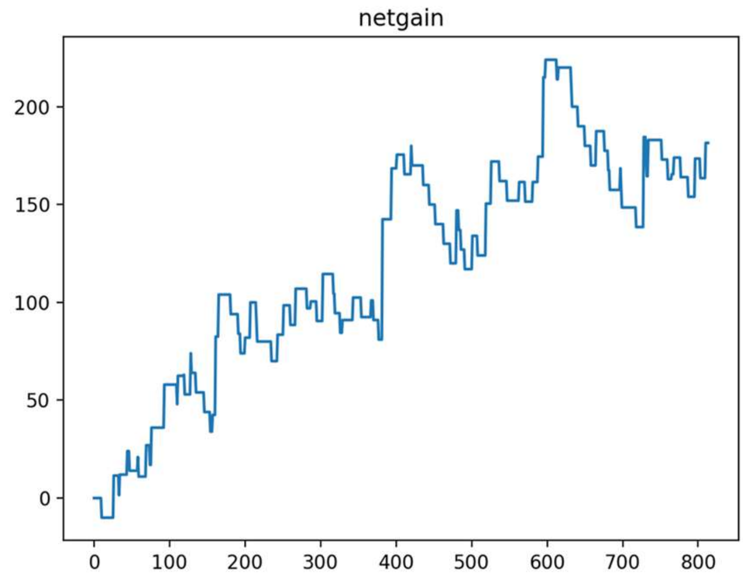


Model works better on higher class (class 1 and 2)

Rank Model

Final result

- We present our results using a set of rank models to establish confidence. Combined with our claim, we have:



Real-time bets

- We step forward and predict the future race (for fun)
- Here we show a race where we predict the 1st correctly

raceid	class	place	finishtime	rankmodel	rankmodel place
2018040207	Class 2	1	81.14	80.116	2
2018040207	Class 2	2	81.51	80.031	1
2018040207	Class 2	3	81.71	80.272	5
2018040207	Class 2	4	81.84	80.248	4
2018040207	Class 2	5	81.87	80.134	3
2018040207	Class 2	6	82.14	80.478	7
2018040207	Class 2	7	82.19	80.445	6

http://racing.hkjc.com/racing/Info/meeting/Results/English/Local/20180402/ST/7

6 Conclusion

Conclusion

- Additional horse data
- Review 3 models
  - Set2seq
  - BN model
  - Rank model
- Achieve promising results
- Try actual bet

Q&A