Horse Racing Prediction using Deep Probabilistic Programming with Python and PyTorch (Uber Pyro)

LYU1805
WONG Yuk 1155074616
Supervised by Professor Michael LYU
Outline

• Background
• Related Works
• Methodology
• Model Structure
• Data Preparation
• Feature Analysis
• Results
• Conclusion
• Future Work
Background – Probabilistic Programming

• Probabilistic Programming describes probabilistic models with programming
• Enables automated inference given probabilistic model
• Mainly applied for making decisions under incomplete information and uncertainty
Background - Deep Probabilistic Programming

• Deep Probabilistic Programming combines deep learning and probabilistic programming

• This project combines deep neural network with probabilistic programming

• Treat weights and biases of neural networks as random variable instead of single point values
  • Usually within a narrow range, may or may not converge to a single point
  • Range represent our uncertainty regarding individual weights and biases
  • If converge to a single point, then reduces to traditional neural network
Background – Horse Racing

- Horse racing is the sport of running horses at speed
- Many factors leading to uncertainty and incomplete information
  - Suitable for Probabilistic Programming
- Hong Kong Jockey Club hosts betting and offers different types of bet
- We focus on 2 types:
  - Win – the horse betted has won the race
  - Place – the horse betted is 1st, 2nd, or 3rd in the race
Background – Horse Class

- HKJC classify horse in classes according to its own rating
- Only horses of the same class race against each other
Related Works

• Relatively few published works
• Previous FYPs have been exploring this topic
• LYU1703
  • Predicted horse finishing time of all horses
  • Sophisticated training by Rank Network
  • Actual net gain for some specific classes (Class 1 and Class 2)
  • Formulated strategy on testing results (lack validation of strategy)
• LYU1603
  • Predicted horse finishing time of all horses
  • Actual net gain obtained for specific threshold (95%)
  • However, the number of bets made are too small
Objective

• Build a prediction model to obtain positive net gain under general circumstances
Evaluation Criteria

• Accuracy of predicting win
• Accuracy of predicting place
• Bet return of predicting win
  • return equals to win odds if correct
  • return equals to -1 if incorrect
• Bet return of predicting place
  • return equals to win odds if correct
  • return equals to -1 if incorrect
Methodology

• 3 different ways to model race results
  • Finishing time regression
  • Win/loss binary classification
  • Place multiclass classification

• Both LYU1603 and LYU1703 do regression on finishing time
  • Difficult to model the distribution of finishing time
  • Normal distribution may be a good assumption

• This project uses multiclass classification on place
  • Predict the probabilities of different places given input data of horse
  • Races are single events, how to get different place probabilities?
  • Sampling layer of Uber Pyro handles this automatically
Model Structure

1. Data preprocessing
   • Normalize numerical data
   • One-hot encode other data

2. Bayesian Neural Network
   • Outputs place probabilities

3. Sampling Layer (Training only)
   • Sample the predicted place

   • Neural Network
   • SoftMax for last layer

   • Sampling layer
   • NN output as probabilities
Data Preparation

• Data obtained from HKJC website from Jan 1 2011 to April 21 2018
• Training Data from Jan 1 2011 to Mar 29 2017 (57334 entries)
• Testing Data from Apr 2 2017 to April 21 2018 (10063 entries)
Data Preparation – Preprocessing

• Normalize real value data according to following equation:
  \[ \hat{X} = \frac{X - \text{mean}(X)}{\text{std}(X)} \]

• One-hot encode categorical data

<table>
<thead>
<tr>
<th>Color</th>
<th>Red</th>
<th>Yellow</th>
<th>Green</th>
</tr>
</thead>
<tbody>
<tr>
<td>Red</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Red</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Yellow</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Green</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Feature Analysis

• There are many data from HKJC website

• In this section, we explore the effect of different features on racing results
Feature Analysis – Excluded Features

- Year of the race
- Day of the race
- Race ID
- Race number
- Horse number

- Note that month of the race is included
Feature Analysis – Horse Origin

- Quite evenly distributed at around 8%, close to random guess of 1/12
Feature Analysis – Horse Age

• If we only bet on horses of age 2, we have ~16% accuracy
• However, this results in very few number of bets
Feature Analysis – Horse Color

• If we only bet on color Roan, we have ~16% accuracy
• However, very few number of bets due to the rarity of Roan color
Feature Analysis – Horse Sex

• The different hormones of different gender affects racing performance
• Rig and Horse are more likely to win than others

![Graph showing conditional sex distribution of winning horses]

Colt: Young male under age 4
Filly: Young female under age 4
Gelding: Castrated male
Horse: Adult male
Mare: Adult female
Rig: Male with testicles concealed
Feature Analysis – Draw

- Smaller draw number is closer to the inside of turn
- Larger draw number is further away from the inside of turn

![Conditional Draw distribution of Winning horse](image)
Feature Analysis – Place in previous race

• Horses that win in previous race are more likely to win
• -1 denotes no previous data
Feature Analysis – Additional Features

• The previous analysis only works for features that are different for horses in the same race

• How about features that are the same?
  • Location? Shatin and Happy Valley have very different tracks
  • Race courses? Race courses of the same location can also be different
  • Horse Class? Different horse class would favor different characteristics
  • Race Distance? Longer distance requires stamina; shorter distance requires acceleration
  • Going? (Soil Condition) Softer soil may favor some horses while harder soil favor others
  • Month of the race? The weather and temperature of each month may affect horse performance

• We decided to include all of the above
Feature Analysis – Non-Identity Features

- Horse Origin
- Horse Age
- Horse Color
- Horse Sex
- Horse Draw
- Horse Old place

- Race Location
- Race Course
- Horse Class (Race Class)
- Race Distance
- Course Going (Soil Condition)
- Race Month
Feature Analysis – Identity Features

• Identity features: horseid, jockeycode, trainercode, sire, dam, dam’s sire
  • Many distinct values
  • Difficult to analyze
  • Leads to large input dimension

• Each individual horse is different

• The jockey in the race may also affect horse performance

• Different trainers results in better performance in particular tracks

• We split the input data into 3 groups
  • No identity features: input dimension of 71
  • Jockey and Trainer: input dimension of 277
  • All identity features: input dimension of 9113 (30x increase!)
Results

• Use neural network with 4 layers, each with 16 neurons
• Train models over 800,000 iterations over the training dataset
• Adam optimizer with an initial learning rate of 0.001
• Sample 100 different neural network from model and take average
Results – Betting

• A horse is bet on if it has the highest place 1 score
• For example, the following tables shows the place 1 score outputted by the neural network
• In this case, we bet on horse 3, because it has the highest score

<table>
<thead>
<tr>
<th>Horse</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
</tr>
</thead>
<tbody>
<tr>
<td>Place 1 Score</td>
<td>0.1</td>
<td>0.15</td>
<td>0.2</td>
<td>0.1</td>
<td>0.05</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Results – Identity Features

- Using Jockey and Trainer have the best performance

<table>
<thead>
<tr>
<th>Model</th>
<th>Public Odds</th>
<th>No Identity</th>
<th>Jockey Trainer</th>
<th>All Identity</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Accuracy\text{_win}</strong></td>
<td>0.2614</td>
<td>0.1840</td>
<td>0.1798</td>
<td>0.1830</td>
</tr>
<tr>
<td><strong>Accuracy\text{_place}</strong></td>
<td>0.5709</td>
<td>0.4513</td>
<td>0.4479</td>
<td>0.4551</td>
</tr>
<tr>
<td><strong>Net gain</strong></td>
<td>-224.90</td>
<td>-184.68</td>
<td>-177.45</td>
<td>-220.29</td>
</tr>
<tr>
<td><strong>Return/Bet For Win Bet</strong></td>
<td>-0.2637</td>
<td>-0.2165</td>
<td>-0.2080</td>
<td>-0.2583</td>
</tr>
</tbody>
</table>
Results – Win Odds

- Win Odds offers the input from public intelligence
- Marginal improvement

<table>
<thead>
<tr>
<th>Model</th>
<th>Public Odds</th>
<th>No Identity</th>
<th>+Odds</th>
<th>Jockey Trainer</th>
<th>+Odds</th>
<th>All Identity</th>
<th>+Odds</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy&lt;sub&gt;win&lt;/sub&gt;</td>
<td>0.2614</td>
<td>0.1840</td>
<td>0.2576</td>
<td>0.1798</td>
<td>0.2592</td>
<td>0.1830</td>
<td>0.2634</td>
</tr>
<tr>
<td>Accuracy&lt;sub&gt;place&lt;/sub&gt;</td>
<td>0.5709</td>
<td>0.4513</td>
<td>0.5695</td>
<td>0.4479</td>
<td>0.5774</td>
<td>0.4551</td>
<td>0.5816</td>
</tr>
<tr>
<td>Net gain</td>
<td>-224.90</td>
<td>-184.68</td>
<td>-184.5</td>
<td>-177.45</td>
<td>-164.65</td>
<td>-220.29</td>
<td>-188.06</td>
</tr>
<tr>
<td>Return/Bet</td>
<td>-0.2637</td>
<td>-0.2165</td>
<td>-0.2163 (0.0002)</td>
<td>-0.2080</td>
<td>-0.2009 (0.0071)</td>
<td>-0.2583</td>
<td>-0.2205 (0.0378)</td>
</tr>
</tbody>
</table>
Results

• Currently, if we bet on all races, we are still unable to obtain a net gain
• Yet to achieve the original objective
• Can we obtain a net gain if we only bet on specific classes?
Results – Training performance by class

- Even with training, there is no net gain in classes other than Class 1 and Group 3
- No reason to bet on other classes

Training performance of model Jockey Trainer by class

<table>
<thead>
<tr>
<th></th>
<th>Class 1</th>
<th>Class 2</th>
<th>Class 3</th>
<th>Class 4</th>
<th>Class 5</th>
<th>Group 1</th>
<th>Group 2</th>
<th>Group 3</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Accuracy\textsubscript{win}</strong></td>
<td>0.2536</td>
<td>0.2522</td>
<td>0.2254</td>
<td>0.2120</td>
<td>0.2168</td>
<td>0.2558</td>
<td>0.2302</td>
<td>0.3349</td>
</tr>
<tr>
<td><strong>Accuracy\textsubscript{place}</strong></td>
<td>0.5199</td>
<td>0.5456</td>
<td>0.4987</td>
<td>0.5027</td>
<td>0.4476</td>
<td>0.5626</td>
<td>0.5721</td>
<td>0.5952</td>
</tr>
<tr>
<td><strong>Net gain</strong></td>
<td>1.69</td>
<td>-54.45</td>
<td>-254.67</td>
<td>-304.50</td>
<td>-19.43</td>
<td>-18.80</td>
<td>-8.32</td>
<td>10.86</td>
</tr>
<tr>
<td><strong>Return/Bet</strong></td>
<td>0.0123</td>
<td>-0.1224</td>
<td>-0.1677</td>
<td>-0.1762</td>
<td>-0.0278</td>
<td>-0.2089</td>
<td>-0.1935</td>
<td>0.1448</td>
</tr>
</tbody>
</table>
Results – Testing performance by class

- Betting only on Class 1 and Group3, we can achieve the following performance:

<table>
<thead>
<tr>
<th>Class</th>
<th>Class 1</th>
<th>Group 3</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy&lt;sub&gt;win&lt;/sub&gt;</td>
<td>0.2771</td>
<td>0.2825</td>
<td>0.2796</td>
</tr>
<tr>
<td>Accuracy&lt;sub&gt;place&lt;/sub&gt;</td>
<td>0.4979</td>
<td>0.6117</td>
<td>0.5504</td>
</tr>
<tr>
<td>Net gain</td>
<td>2.45</td>
<td>7.89</td>
<td>10.34</td>
</tr>
<tr>
<td>Return/Bet</td>
<td>0.1753</td>
<td>0.6573</td>
<td>0.3977</td>
</tr>
</tbody>
</table>

Testing performance of model Jockey Trainer by class
Conclusion

• Currently, if we bet on all races, we are still unable to obtain a net gain
• We have yet to achieve the original objective
• We can obtain a net gain of ~40% if we only bet on specific classes
Future work

• Current model has 4 layers with 16 neurons per layer
• May be too small to fully utilize the large input size
• Explore different hyper parameters

• Current model will be bet on a horse if it has the highest score
• Even if the score is low (as low as 0.1, i.e., 10% chance to win)
  • If winodds is smaller than 10, then there is an expected loss
• Adjust to betting only if probability > (1/winodds)
Future work

• Current model takes entry input one by one
• Fails to take performance of other horses into consideration
• Build a model which takes all horses in the same races for input

• Current model give equal importance to all training data
• However, correctly predicting horses of high win odds is more important because this results in higher return
• Duplicate training data according to win odds
• Reinforcement learning
Thank you!