MAST30034: Applied Data Science Group Project

A Predictive Model to Maximise the Total Earnings
of
a New York City Yellow Taxi Driver

Group 7:

Geng Yuxiang
Li Shangqian
Xuanken Tay
Yin Zhou Zheng

2019
The University of Melbourne

Contents

1 Introduction

1.1 Problem Aim & Description

The ultimate aim of this project is to build a predictive model that will maximise the total earnings (fare amount + tips) of a yellow taxi driver in New York City. To facilitate the development of an accurate and practical predictive model, the project sets several restrictions which simulate realistic circumstances of a NYC yellow taxi driver. Restrictions include the input, whereby only the current date-time and location of the taxi is known, a minimum of eight hours between shifts, and others.

To evaluate the performance of a predictive model / player, the player is run through a simulation. The simulation considers a past, contiguous week's trip data (yellow and green) and awards the player with past trips based on the simulation's current date-time, and the player's current status and location. The player generating the highest average total earnings for the week - over multiple simulations - maximises (relative to other players) the total earnings of a NYC yellow taxi driver. Further details on the simulation are discussed in the next section.

The immediate goal of this project is to develop a predictive model that will maximise the average total earnings of a NYC yellow taxi driver over several simulations on a week's records of yellow and green taxi trips. Assuming the simulation design accurately mimics the real-world scenario, then an optimal predictive model for the simulation will also help maximise the total earnings of a real NYC yellow taxi driver.

1.2 Simulation Design & Provided Data-Sets

1.2.1 Game-Board

To simplify the problem, the project features a tilted rectangular grid overlay on top of the New York City map (see *Figure 1*).

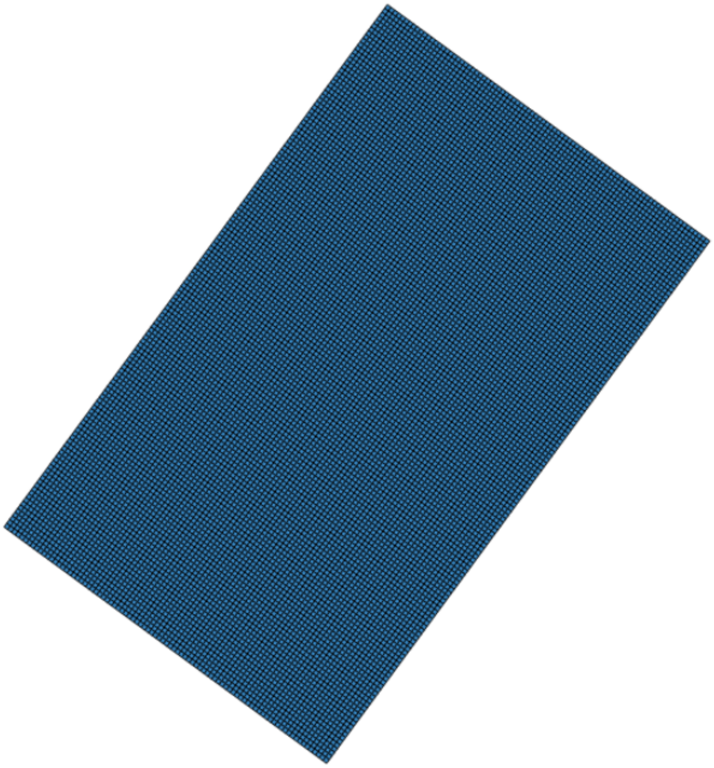

Figure 1: Tilted Rectangular Grid Covering NYC

The grid consists of equally sized square cells which cover the entirety of New York City. Each cell has a unique ID in the form of $x : y$ where x is the row number of the cell and y is the column number of the cell.

Cells with any of the following conditions are then removed:

- The cell does not contain a road.
- The cell does not feature any yellow or green taxi pick-ups within 2015 - based on the New York City Taxi and Limousine Commission (TLC) trip data.
- The cell is not connected to the rest of the cells.

The grid's remaining cells then become the game-board (see *Figure 2*).

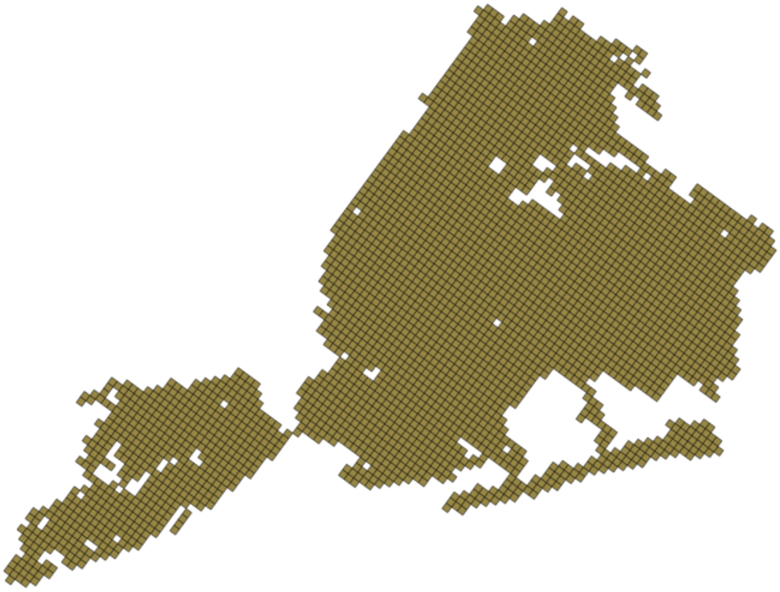

Figure 2: Game-Board

1.2.2 General Rules

The simulation is turn-based with each round representing a minute in-game. For each round, the player is provided with the current date-time (accurate to the minute) and the player's current location cell. The player must output a decision for each round. The decision determines the state of the player for the next round defined by the *Availability* and *Default Action* of the player.

The *Availability* of the player can be either "For Hire" or "Unavailable". The former allows the player to be awarded a trip if the simulation's past data contains a pick-up at the current date-time and location cell of the player. The latter ignores all pick-ups. The *Default Action* can be either to "Stay" or "Move". The former stays in the player's current cell, and the latter moves to a cell (of choice) neighbouring the player's current cell.

When a trip is awarded, the player is removed from the game until the trip's drop-off date-time. Upon return, the player is relocated to the trip's drop-off cell. Since the player can only move to one neighbouring cell per round, this restricts the player's speed to one cell per minute when the taxi is not taken.

The simulation runs on a contiguous week of past trip data. The player is restricted to a maximum of six shifts. Each shift can be a maximum of 12 hours, with a further restriction on the time hired: a maximum of 10 hours per shift. The player can choose the starting cell and date-time of each shift; provided that they all finish within the simulated week and are separated by at least eight hours.

1.2.3 Provided Data-Sets & Evaluation

Green and yellow taxi trip data for training the predictive models are provided by the project supervisors. The data-sets were originally from the TLC website. However, two additional columns have been added, providing the pick-up and drop-off location cell IDs. Besides the two additional columns, the data-sets are no different from the original TLC data-sets.

The provided data-sets are separated by month and taxi colour (yellow or green). Combined, the data-sets for model training contain all trips recorded between July 2015 and June 2017.

Each simulation runs on a random week within January 2015 to June 2015, or July 2017 to December 2017; generating several quantities of interest for each predictive model (player) to be used for model evaluation and refinement. One key quantity as a measure of player performance will be the total earnings for the week.

The simulation data is also cleaned by the project supervisors, with criterion described in **Section 3.2.1 Game Data**.

1.3 Optimisation

The fundamental heuristic that will govern our predictive model is the optimisation of the time hired. We define our player to be "hired" when our taxi is taken. Therefore, in the simulation, the time hired is equivalent to the time "out-of-the-game". Ultimately, we wish to maximise the total time hired within any simulated week.

We believe maximising the time hired will help maximise the total earnings. From previous analyses, we have seen that green taxis show a very strong linear relationship between trip duration and fare amount. Additionally, for yellow taxis paid by credit card, we observed that the vast majority of trips received tips between one and two USD.

However, previous analyses are based on a small subset of the provided data-sets; therefore, they may be biased, not reflecting the general trend for the entire July 2015 to June 2017 period. As such, we verify these trends by producing plots of data randomly sampled across the entire period.

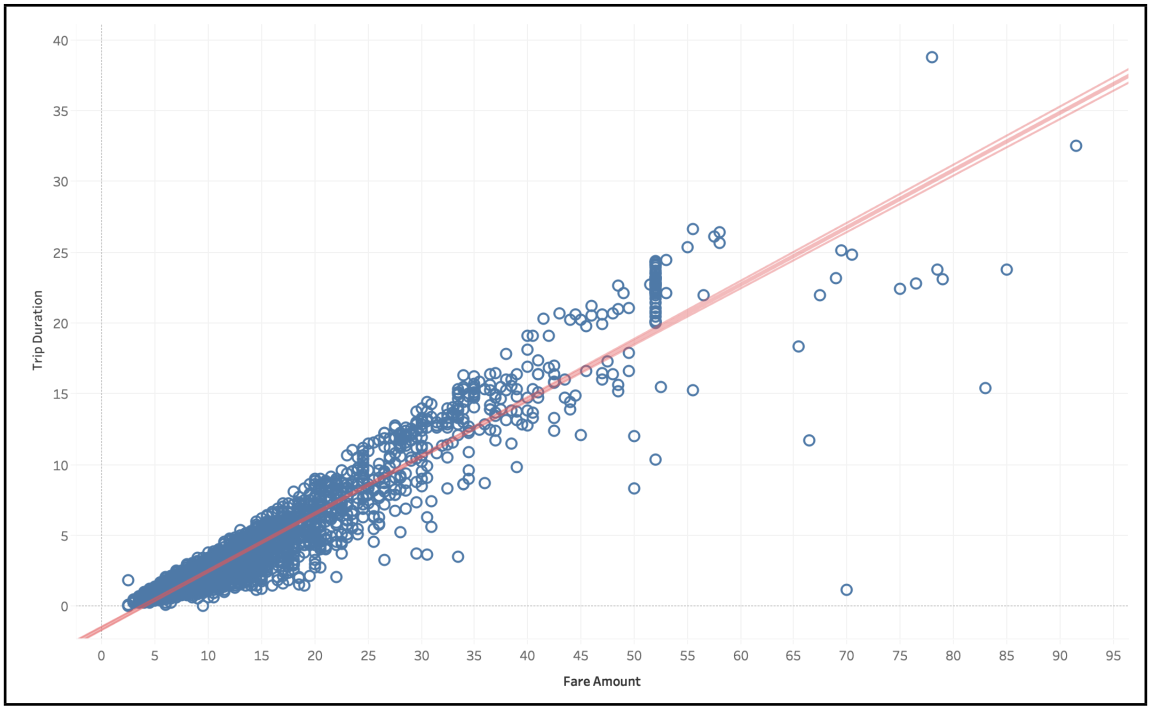

Figure 3: Trip Duration vs. Fare Amount

In *Figure 3*, we can identify a linear relationship between the trip duration and the fare amount. Additionally, in *Figure 4*, we see that the majority of tips fall in between one to two USD. This allows us to assume with more confidence that these trends are also present across the entire period (July 2015 to June 2017). Additionally, trips paid by cash are included in the simulation data; however, their tips are not recorded and consequently set to zero [1].

Combining these details, we infer that the majority of total earnings originates from the generated fare; therefore, maximising the time hired should maximise the generated fare, thereby maximising the total earnings.

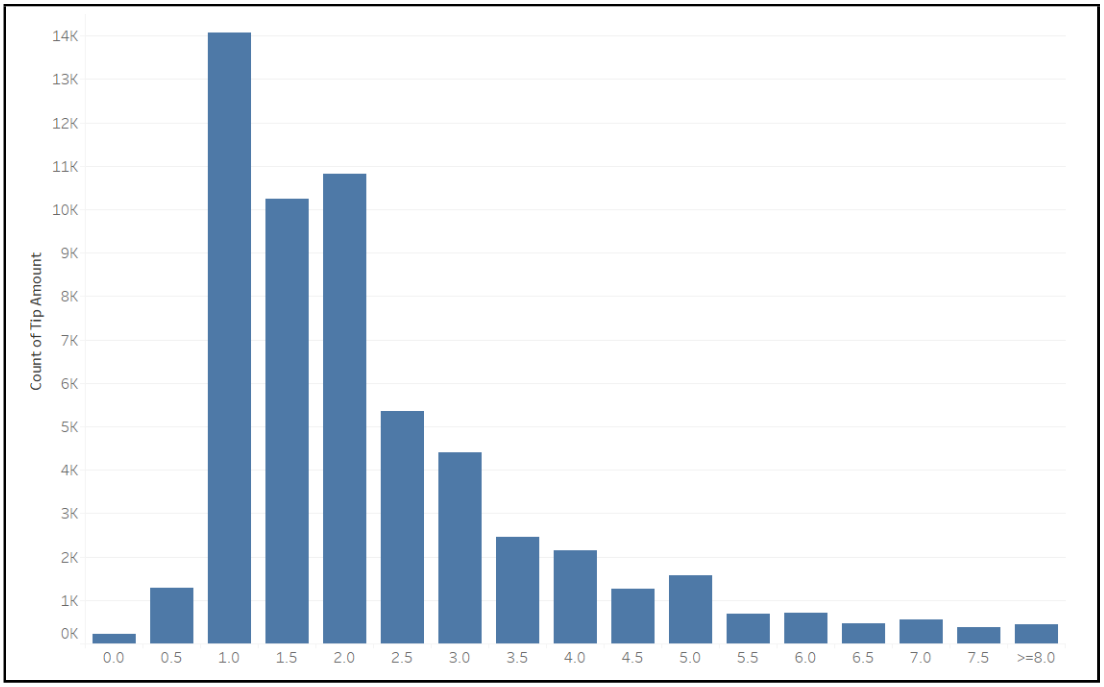

Figure 4: Histogram of Tip Amounts

2 Initial Model Design

2.1 High-Level Description

Our predictive model (player) seeks to maximise total earnings by maximising the total time hired. Therefore, our player aims to stay hired for as long as possible during each shift by picking up whatever trip it can find. Thus, the goal of our player model is to be more capable in finding trips; allowing it to stay hired for longer. In other words, our player model maximises the number of awarded trips.

As such, two immediate decisions were made. Firstly, whenever our taxi is not taken, our state will be set to "For Hire". And secondly, our player will only operate in Manhattan; re-routing back to Manhattan - using the shortest path determined by Breadth First Search - if we drop off a passenger in a different borough. The second decision is explained in **Section 2.2 Manhattan Player**.

The simulation also allows us to choose our player's shift start times, and the starting location cell of each shift. Since we wish to maximise the number of awarded trips, our shift times will be determined by analysing the most "active" hours of the week - based on past trip data. Similarly, our starting locations will be determined by the most "active" locations. Further details are provided in **Section 2.3 Shift Start Times & Locations**.

Lastly, for each round of the simulation, our player model must output the next destination cell: either to stay in the current cell, or move to a neighbouring cell. To make this decision, our player model essentially calculates the probability of being awarded a trip for **all** possible move sequences / routes in the next k rounds. It then outputs the first cell **of the route** with the highest probability of being awarded a trip within the next k rounds. This method allows our player to gain some "foresight"; weighing the implications of the next round's move. Further details are discussed in **Section 2.4 Route Model**.

To put it briefly, our initial player model is a model based in Manhattan that is always "For Hire" whenever it is available. And when our player is available, it maneuvers through Manhattan in a way that maximises its chances of being awarded a trip as soon as possible.

2.2 Manhattan Player

Our decision to base our player in Manhattan is based on two analyses: the trip distribution across NYC, and the performance of borough-based "random walkers". A consequence of our decision also results in **most of our analyses being carried out strictly within Manhattan**.

2.2.1 NYC Trip Distribution

From previous analyses on the yellow taxi trip data, we have observed that the majority of trips within New York City occur in Manhattan. However, the simulation includes both green and yellow taxi trip data; and green taxis are restricted to picking up passengers outside of Manhattan [2]. Despite the addition of green taxi trip data, the proportion of yellow taxi trips greatly outweighs green taxi trips. Therefore, the addition of green taxi trips should not have a significant effect on the distribution of trips across NYC. We verify this belief by analysing the proportion of trips in each NYC borough for the entire set of provided data.

We derive our proportions by combining all the provided data-sets together. This includes both yellow and green taxi trips - since the simulation includes both. We then count the number of trips that occur in each NYC borough; dividing each by the total number of trips across NYC to determine the proportions as follows (see *Figure 5*).

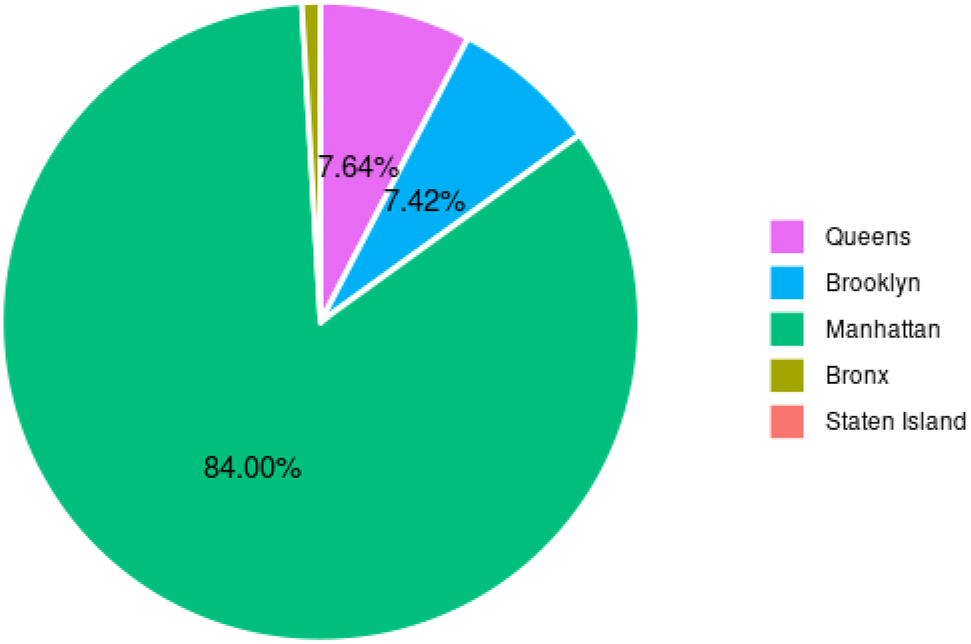

Figure 5: Proportion of Trips in Each NYC Borough for Trips (2015 July to 2017 June)

Despite the addition of green taxi trip data, we see that the majority (84%) of taxi trips still occur within Manhattan. This implies a higher availability of trips within Manhattan; supporting our decision to base our player in Manhattan. Doing so allows us to maximise the number of awarded trips; therefore maximising the total time hired and consequently, the total earnings.

2.2.2 Random Walker Performance

The project supervisors also provided "Random Walkers": simulation players that are always "For Hire" and move about randomly each round. We modified these random walkers by restricting their operations to a single borough. For instance, a Manhattan random walker can only pick-up passengers within Manhattan. And if the walker exits Manhattan, they will be re-routed back into Manhattan before becoming available again.

We created a separate random walker for each New York City borough; running all of the walkers through 10 runs of the simulation. Each run of the simulation uses trip data from the period: 2015-06-01 00:00:00 to 2015-06-07 23:59:59 - a week from the test data period. The results can be found in the following table:

Borough	Mean Earnings (USD)	Min. Earnings	Max. Earnings	SD Earnings
Manhattan	3171.08	2790.06	3506.32	198.29
Brooklyn	954.55	437.43	1510.39	310.62
Queens	944.87	598.13	1237.71	190.05
The Bronx	427.32	81.04	757.91	214.39
Staten Island	1.1	0	11	3.3

Table 1: Borough Random Walkers' Performances

Evidently, the Manhattan random walker generates the highest earnings. By inspecting the log files (which record a taxi's actions for each round of the simulation), we could see that a key issue affecting the earnings was the availability of trips. That is, a random walker is more likely to be awarded a trip in Manhattan compared to other boroughs. Therefore, we chose to base our player in Manhattan where more trips are available; allowing us to more easily maximise the number of awarded trips.

2.3 Shift Start Times & Locations

2.3.1 Shift Start Times

Following the same goal as our Manhattan decision, we wish to schedule our shifts when more trips are available; allowing us to maximise the number of awarded trips. The simulation does not allow shifts to end early; therefore, the shift ends after 12 hours has passed, or the taxi has been hired for 10 hours. Thus, the start time of the shift essentially determines the end time of the shift.

We define 'active' hours as periods when more pick-ups/trips are made. Since our player is restricted to Manhattan, we only look at the distribution of trips within Manhattan across the hours of the week. This is done by counting the number of trips occurring in each hour of the week across the entire data-set period (July 2015 to June 2017). We then obtain the average (mean) by dividing each hour of the week's count by 104; since (52×2) weeks occur within the data-set period of two years.

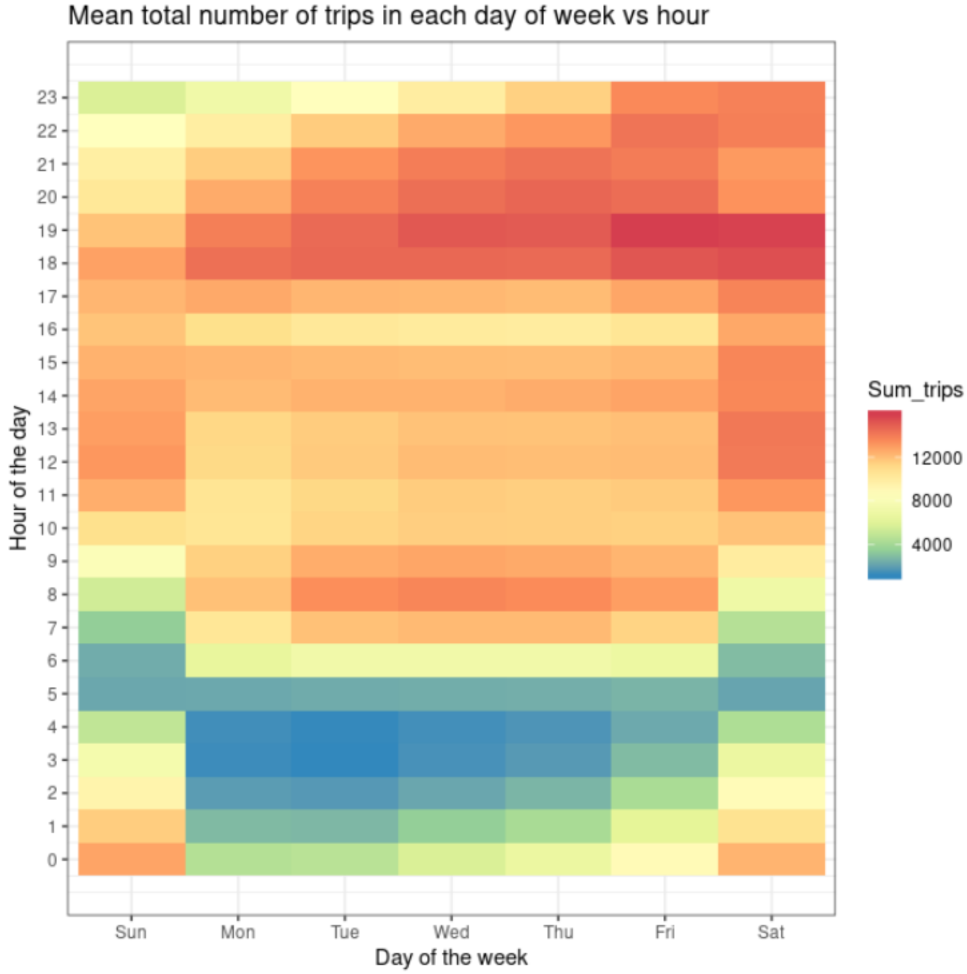

Figure 6: Average (Mean) Frequency of Trips for Each Hour of the Week

From the heat-map in *Figure 6*, we simply "eye-balled" the most active hours and set them as our shift times; hard-coding the corresponding shift start times. We looked for six contiguous 12 hour intervals that covered the most active periods (indicated by red). Our hard-coded shift times are the following:

1. *Monday 09:00* to *Monday 21:00*
2. *Tuesday 09:00* to *Tuesday 21:00*
3. *Wednesday 08:00* to *Wednesday 20:00*
4. *Thursday 11:00* to *Thursday 23:00*
5. *Friday 13:00* to *Saturday 01:00*

6. *Saturday 11:00* to *Saturday 23:00*

We made sure that each shift was separated by at least eight hours of rest. However, it is possible that the last trip of a shift is unusually long - resulting in a shift longer than 12 hours. This might result in less than eight hours of rest before the next pre-defined shift; consequently "kicking-out" our player from the game for an invalid shift time. We prevent this by delaying the next pre-defined shift to ensure eight hours of rest is provided.

It is also worth noting that our method of choosing shift times assumes that each week within the provided data-set has a similar distribution of trips across the hours of the week; and that this also extends to the simulation test data.

2.3.2 Shift Start Locations

To determine our shift start locations, we apply a similar approach as our shift start times: we look for the most active locations. We initially produce a trip frequency look-up table / data-set grouped on the 10-minute interval of the week (described in **Section 3.5 Trip Frequency Look-Up Table**). From this data-set, we can determine the most active taxi zones (highest expected trip frequencies) at the start of each shift. We then choose a random cell within the most active taxi zone to be our starting location cell.

2.4 Route Model

The route model essentially dictates our player's location cell for the next round. This involves deciding whether to stay in the current cell or move to a neighbouring cell.

Assuming our player is currently in a cell that is completely surrounded by neighbouring cells, then our player is open to nine different decisions for the next round. Either we can stay in our current cell, or move to one of the eight neighbouring cells.

To use our route model, we must initially set a positive integer k which we refer to as our "look-ahead" value. The model then calculates the probability of being awarded a trip for all possible move sequences / routes within the next k rounds. Therefore, assuming all encountered cells are completely surrounded by neighbours, then there are 9^k different routes for the next k rounds.

The model then chooses the first cell of the route with the highest probability of being awarded a trip as our location cell to move to in the next round. However, we do not continue to follow this route, but instead re-do this computation each round; again, choosing the first cell of the newly computed "best" route as our location cell for the next round. Our main intention in determining a route each round is not to set a route for the next k rounds, but to allow our player the ability to weigh the implications of the next round's move.

We explored several different weighting schemes for determining a route's likelihood of being awarded a trip - experimenting with different random functions and arbitrary threshold values. We concluded that using a probability weighting scheme was ideal since it does not involve an arbitrary choice of values, and preserves the most amount of information. Details concerning the actual computation of a route probability are explained in the following sections.

2.4.1 Probability of a Cell

A route's probability of being awarded a trip is determined by each cell's (within the route) probability of being awarded a trip. Each cell's probability is determined by its corresponding taxi zone, the specified 10-minute interval of the day, whether it is a weekday or weekend, and the expected trip frequency at the corresponding taxi zone and time interval.

The expected trip frequency values - given a taxi zone, 10-minute interval of the day, and whether it is a weekday or weekend - are stored in a pre-processed data-set we refer to as our "look-up table" (described in **Section 3.5**). These values are used to determine a cell's expected trip frequency value for a specified minute of the weekday or weekend.

We obtain the cell's expected trip frequency value - for a specified minute of the weekday or weekend - by dividing the taxi zone's expected trip frequency - for the corresponding 10-minute interval - by the number of cells it contains. We then divide this value by 10 to obtain the expected trip frequency for the cell and specified minute of the day. This assumes that a taxi zone's trips are evenly distributed among its cells and trips are evenly spread out within each 10-minute interval.

We then convert this frequency to a probability using the following formula:

$$Pr(\text{Getting a Trip}) = \frac{Frequency}{Frequency + 1} \tag{1}$$

Using the above formula, we can obtain the probability of being awarded a trip for a particular cell given the time of the day (accurate to a minute) and whether the day is a weekday or weekend. Note that the formula is the same as treating the trip frequency as *odds*, which features several desirable properties:

- A frequency of zero results in a zero probability. This indicates that it is impossible to be awarded a trip at the frequency's corresponding cell and time interval.

- A higher frequency results in a higher probability; approaching one as the frequency approaches infinite. This is desirable since a higher frequency should imply a higher chance of being awarded a trip.

- Rate of increase in the probability is not constant but diminishing with respect to frequency. This intuitively makes sense if we consider the likelihood of being awarded a trip when a competitor taxi is in the same cell at the same minute of the day. This makes $Frequency = 11$ not much better (in terms of probability) than $Frequency = 10$, while $Frequency = 2$ is greatly preferable over $Frequency = 1$ (2/3 chance compared to 1/2 chance).

- The formula maps any frequency to a valid probability where the value is contained within $[0, 1]$.

2.4.2 Probability of Route

To compute the probability of being awarded a trip for a particular route, we have the following:

Consider the route $R = C_1, C_2, ..., C_{k-1}, C_k$ where:

- C_i is the i^{th} cell in the route.

- k is the "look-ahead" value, which is the number of minutes/rounds we want to consider in advance.

With Equation (1), we have the means of calculating the probability of being awarded a trip at C_i where $i \in \{1, 2, ..., k\}$ - at anytime t. Formally,

$$\begin{aligned}
&Pr(\text{Getting a Trip at } C_i \text{ at Time } t)\\
&= p_{it}\\
&= \frac{\widehat{freq}_{it}}{\widehat{freq}_{it} + 1}
\end{aligned}$$

...where $\widehat{freq}_{it}$ is the expected trip frequency / number of trips in C_i at time t.

We then define the probability of being awarded a trip in the next k rounds following the route R as the sum of individual joint probabilities p_{it}, where $i \in \{1, 2, ..., k\}$ and $t \in \{(c+1), (c+2), ..., (c+k)\}$, and where c is the current simulation time (accurate to the minute) in this case. We then have the following equation:

$$\begin{aligned}
&Pr(\text{Getting a Trip along } R)\\
&= Pr(A_{1(c+1)}) + Pr(\overline{A_{1(c+1)}}) \times Pr(A_{2(c+2)}) + ... + \prod_{i=1}^{k-1}[Pr(\overline{A_{i(c+i)}})] \times Pr(A_{k(c+k)})\\
&= p_{1(c+1)} + (1 - p_{1(c+1)}) \times p_{2(c+2)} + ... + \prod_{i=1}^{k-1}[(1 - p_{i(c+i)})] \times p_{k(c+k)}, \qquad (2)
\end{aligned}$$

...where:

- A_{it} is the event that a trip is awarded at C_i at time t. Conversely, $\overline{A_{it}}$ is the event that **no** trip is awarded at C_i at time t.
- $Pr(A_{it})$ or p_{it} is the probability of being awarded a trip at C_i at time t (accurate to the minute). Time t is equivalent to the time of the week when our player arrives at C_i if the route is followed through.
- Similarly, $Pr(\overline{A_{it}}) = 1 - P(A_{it})$ is the probability of **not** being awarded a trip at C_i at

the time of arrival.

We note that Equation (2) assumes that the A_{it}'s are independent $\forall i, t$. Additionally, the equation suggests that to compute the probabilities of up to 9^k possible routes, we must inevitably face an undesirable exponential computation. The simulation allows up to five seconds of decision-making time for each round. Therefore, we initially set our k **look-ahead value to three** to avoid exceeding this time limit.

2.5 Model Diagram

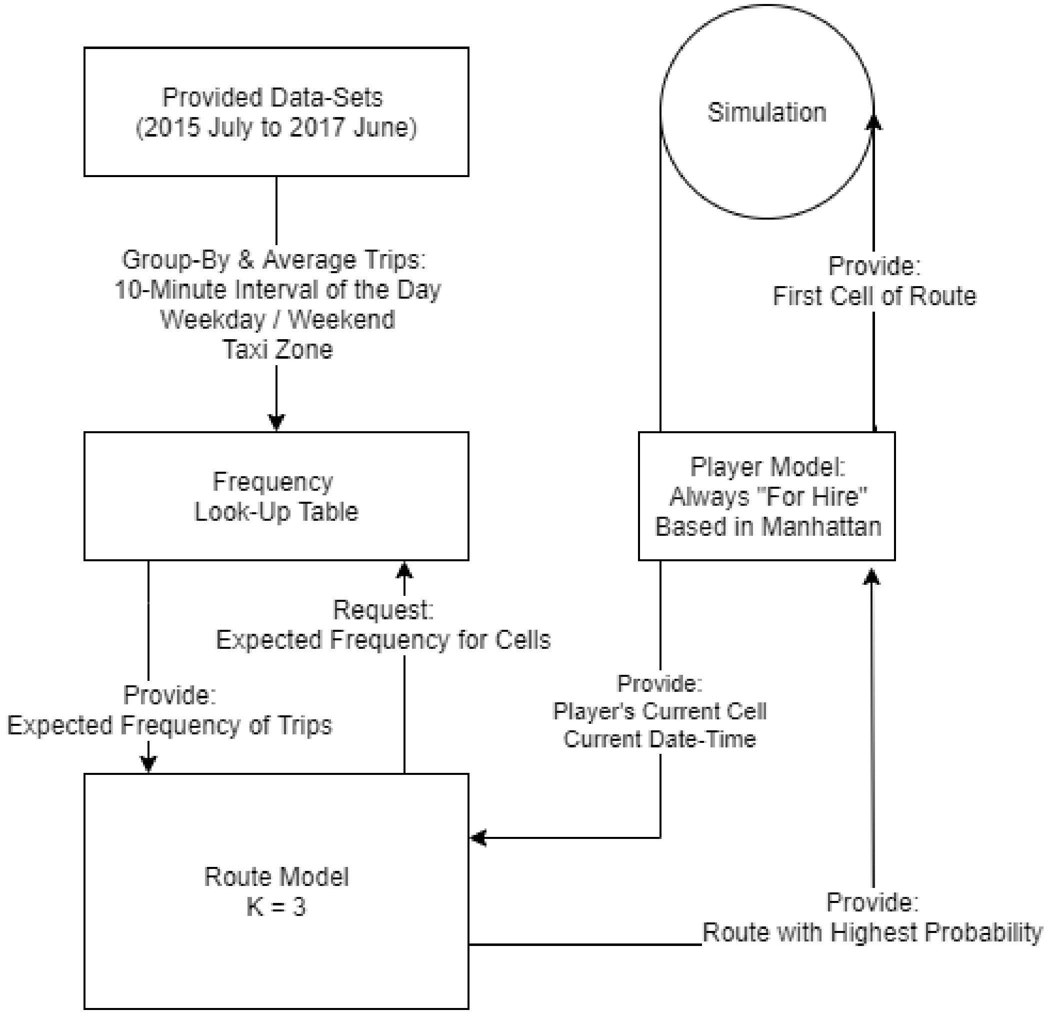

Figure 7: Flow Chart of Initial Model

3 Pre-Processing & Cleaning

3.1 Data-Set Selection

As mentioned previously, data-sets are provided for training the predictive model. Besides the addition of two columns recording the pick-up and drop-off cells, these data-sets are no different than the data-sets available on the TLC website. The data-sets are grouped by the colour of the taxi (green or yellow) and the month of the trip; covering the period from July 2015 to June 2017.

The simulation runs on a random week within January 2015 to June 2015, or July 2017 to December 2017. However, it combines the trip data of both green and yellow taxis. Therefore, we also combine the green and yellow taxi data-sets so that our predictive model is built on data that we will expect to see in the simulation (test) data.

Since the test data is run on a random week within the “combined” year, we chose to use **all two year’s worth of provided data** to build a model that generalises the patterns observed in a year. This should help make more informed decisions to maximise total earnings for any given simulation data. Combined, the “raw” provided data-sets have a total size of 37GB.

3.2 Cleaning

3.2.1 Game Data

To train a more effective predictive model, the training data should share the same characteristics as the testing data. The testing data is provided by the project supervisors. They apply several cleaning criteria which remove trips with any of the following qualities:

- The pick-up or drop-off cell is blank.
- The trip duration is less than one minute.
- The trip does not start and finish within the game time (i.e. any trip that starts before Sunday midnight and finishes after).

- The payment is not cash or credit card.

To share the same characteristics, we apply the same criteria to our training data.

3.2.2 Erroneous Records & Outlier Removal

In addition to the removal of records that will not appear in the simulation/game data; we apply several other cleaning criteria to remove erroneous records and outliers.

We choose to remove outliers since we wish to produce a player model that performs consistently. Outliers represent unusual behaviour, departing from the consistent and expected behaviour that we wish to achieve. By having a consistent player, we can be more confident in estimating its expected total earnings for a week.

We apply the following cleaning criteria:

- The passenger count is non-negative.

  A negative passenger count would be impossible and is likely an input error.

- The max trip duration is 120 minutes.

  From previous projects' analyses, 120 minutes has been shown to be the approximate 95% percentile for trip duration whereby 5% of trips had a duration greater than 120 minutes. Since we wish to train a consistent player, we treat trips longer than 120 minutes as outliers and remove them.

- The total earnings (fare amount + tips) is within $(0, 100)$.

  A negative total earnings should be impossible; while total earnings greater than 100 USD are considered outliers (in the same manner as trips longer than 120 minutes).

3.2.3 Summary

The cleaning process is carried out on each of the provided data-sets. With the exception of 'trip duration' and 'total earnings', the cleaning can be carried out by directly referencing the attributes available in the "raw" provided data-sets.

In summary, we apply all the following cleaning processes (where justifications have been provided above):

- *Remove trips with a negative 'passenger_count'*:

  The data-set attribute 'passenger_count' records the number of passengers on the trip. We note that a value of zero is still a valid trip where an item is being delivered.

- *Remove trips with durations less than one minute or greater than 120 minutes*:

  We initially derive the trip duration in minutes by taking the time difference between the pick-up and drop-off date-times recorded in attributes 'tpep_pickup_datetime' & 'tpep_dropoff_datetime' respectively. We then proceed to remove the appropriate trips.

- *Remove trips with missing 'PickupCell' or 'DropoffCell'*:

  These attributes record the trip's pick-up cell and drop-off cell respectively [1].

- *Keep trips where 'payment_type' is equivalent to one or two*:

  This attribute records the trip's method of payment. One and two correspond to credit card and cash respectively.

- *Remove trips with total earnings less than zero or more than 100 USD*:

  We initially derive the total earnings by adding the 'fare_amount' and 'tip_amount' attributes which record the trip's fare amount and tip amount in USD respectively. We then remove unwanted trips.

- *Remove any trip that starts before a Sunday midnight and finishes after the same midnight*:

  We do this by inspecting the 'tpep_pickup_datetime' & 'tpep_dropoff_datetime' attributes which record a trip's pick-up and drop-off times respectively.

3.3 Feature Engineering

After cleaning the entire set of provided data, we start extracting the trip records from each data-set; assigning them into new data-sets, each storing trips occurring in the following half-year periods:

- *2015 July* to *2015 December*

- *2016 January* to *2016 June*

- *2016 July* to *2016 December*
- *2017 January* to *2017 June*

However, we only extract the following attributes for each trip:

1. '*Season*': The season of the year when the pick-up occurred.
2. '*Pickup_month*': The month of the year when the pick-up occurred.
3. '*Pickup_week*': The week of the year when the pick-up occurred.
4. '*Pickup_wday*': The day of the week when the pick-up occurred.
5. '*Weekend*": Boolean indicating whether the pick-up occurred on a weekend or not.
6. '*Pickup_hour*': The hour of the day when the pick-up occurred.
7. '*Time*': If pick-up occurred during daytime (06:00 - 18:00) or nighttime (18:00 - 06:00).
8. '*Pickup_day*': The day of the month when the pick-up occurred.
9. '*Pickup_minute*': The minute of the hour when the pick-up occurred.
10. '*Pickup_cell*': The game-board cell ID of the pick-up.
11. '*Dropoff_cell*': The game-board cell ID of the drop-off.
12. '*Trip_distance*': The distance of the trip in miles.
13. '*Trip_duration*': The duration of the trip in minutes.
14. '*Total_earnings*': The trip's total earnings (fare amount + tips).

Attributes $\{1, 2, 3, 4, 5, 6, 7, 8, 9\}$ are derived from the trip's *'tpep_pickup_datetime'*: an attribute providing the date-time of a trip's pick-up. We can use these attributes to perform further pre-processing, where we *aggregate* trips into particular groups. For instance, we can group all the trips by each unique combination of the pick-up weekday, hour, and cell. The new data-set will consist of rows for each unique combination of these attributes, possibly including the average of trip distances, durations or earnings for each group.

Attributes $\{10, 11, 12\}$ are directly copied from the original cleaned data-sets. And attributes $\{13, 14\}$ require some simple manipulations that are briefly described in the summary of cleaning techniques listed in **Section 3.2.3**. We choose to keep the trip distance, duration and earnings since we assume that they are good indicators of trip profitability; therefore, they can potentially help improve our model. Although we do not end up using these attributes within this project, we do believe that they can be helpful in improving the performance of our final model.

After completing the above cleaning and pre-processing, we end up with four data-sets containing trips for each half-year. Each data-set contains roughly 55 million trips and takes up around 3.5 GB of disk space.

3.4 Aggregation

3.4.1 Grouping

The structure of our four data-sets allows for further pre-processing by grouping a particular set of time-related attributes. However, any choice of grouping is accompanied by an assumption. For instance, if we group trips by the month of the year, aggregating the trips' total earnings by taking the median; then we are assuming that the month *may* have an effect on the median total earnings, and that the year of the month has no effect.

3.4.2 Aggregation Methods

When we take a particular grouping, we often encounter the situation where many trips belong to the same group. While some of the trip's attributes can be discarded; we prefer to keep others (e.g. total earnings). However, since each row of the new data-set represents a different group, we must somehow aggregate and summarise the quantity of interest for trips within each group.

We choose to use the median for aggregating each quantity of interest (trip duration, distance and total earnings). Therefore, for trips belonging to a single group, the median of their quantity of interest is taken to represent their corresponding group.

Our aim is to create a *consistent* player. Unlike the mean, the median is not affected by outliers. Therefore, it is a better measure for the *expected* result. Consequently, models trained from data-sets aggregated with the median should result in more consistent players.

3.5 Trip Frequency Look-Up Table

By applying a particular grouping to our four half-year data-sets, we can create a "look-up" table that stores the expected frequency of trips for a given time interval. This "look-up" table is crucial for the route model since it provides the expected frequency values for each taxi zone in Manhattan.

Over the course of the project, we pre-processed three different frequency look-up tables - each using a different grouping. Each new look-up table helped improve our player model's performance. The groupings of the three look-up tables are explained in more detail below.

3.5.1 Initial Model

The look-up table used in our initial model grouped trips by the ten-minute interval of the day, whether the trip was on a weekday or weekend, and the pick-up taxi zone of the trip. This involved inspecting the following attributes (in the four half-year data-sets) for each trip:

- *'Pickup_minute'* & *'Pickup_hour'*:

  These were both required to determine the trip's 10-minute interval of the day. For instance, if the pick-up hour is 21 and the pick-up minute is 32; then the pick-up time is 21:32 or 9:32PM. We **round down** to the nearest 10-minute interval; therefore, 21:32 maps to 21:30, which represents the 10-minute interval of the day from 21:30 to 21:40.

- *'Weekend"*:

  We did not distinguish the trip's day of the week; instead, we only consider if the trip occurred on a weekday or a weekend.

- *'Pickup_cell'*:

We map each cell to a Manhattan taxi zone. Therefore, we loosen the assumption that each cell has an effect on the frequency of trips. Instead, we assume that each taxi zone has an effect on the trip frequency.

When mapping the cells to zones; there were several cells that intersected with more than one taxi zone. For such a case, we looked at the trip data containing longitude and latitude coordinates, mapping the cell to the taxi zone where most of its trips occurred (according to the longitude and latitude coordinates).

We group on these attributes (where the pick-up cell has been transformed to the pick-up taxi zone) using all four half-year data-sets. We also count the total number of trips that occur in each unique group. The new table contains rows for each unique set of:

- 10-Minute Interval of the Day
- Weekday or Weekend
- Taxi Zone

We then divide each row's total number of trips by 104 since 52×2 weeks occur within the four half-year data-sets. We also divide the trip frequency for rows representing weekdays by five so that we obtain the expected number of trips for a specified 10-minute interval on **any** weekday. Similarly, we divide the trip frequency for rows representing weekends by two.

For any missing groups, we set their expected frequencies to zero. This intuitively makes sense: if we did not count any trips for a particular group throughout the entire data-set period, then we have reason to believe that no trips will occur for that group. We also note that the method for deriving each of the expected frequencies assumes that the distribution of trips is the same across weeks, weekdays, and weekends. This is the structure of the frequency look-up table used in our initial model.

However, our route model requires the expected trip frequency for a **cell's** specified minute of the day. Therefore, our route model first identifies the expected frequency of the cell's corresponding taxi zone, dividing the value by the number of cells within that taxi zone to obtain the cell's expected trip frequency. It then further divides the value by 10 to obtain the expected trip frequency for the specified minute. Thus, we are also assuming that the expected frequency of a taxi zone is divided evenly among its cells; and that trips are evenly spread out within each 10-minute interval.

3.5.2 Day of the Week

In the model refinement stage, we make a slight adjustment to the groupings of our frequency look-up table. Instead of grouping the trip on whether it occurs on a weekday or weekend, we group the trip on the day of the week it occurred. Therefore, we retract our previous assumption that the distribution of trips is the same across weekdays and across weekends. Instead, we assume that the day of the week has an effect on the distribution.

This decision was mainly motivated by 'eyeballing' the heat-map representing the trip distribution across hours of the week in *Figure 6.* It is evident that the distribution is not the exact same across weekdays and across weekends (by observing the changes in colour horizontally across weekdays or weekends); therefore, we have reason to believe that the day of the week affects the distribution of trips across the day.

By using this new grouping, we believe that the expected frequencies for a specified 10-minute interval of the week will be more accurate. Therefore, our model will be able to make more informed decisions.

3.5.3 Five-Minute Intervals

The last adjustment we made to the frequency look-up table was changing the groupings of ten-minute intervals to five-minute intervals. For instance, a trip that occurs at 12:56 will **round down** to the nearest five-minute interval; therefore, 12:56 maps to 12:55, which represents the five-minute interval of the day from 12:55 to 13:00.

A smaller interval gives us more accurate expected frequencies since we are generalising the expected frequency of trips over a smaller interval. However, this is with respect to the training data-set. Therefore, reducing the interval can lead to over-fitting. In this case, the change to five-minute intervals lead to consistent, albeit small improvements in model performance (in terms of total earnings) when compared to the ten-minute intervals. Thus, this is a good indication that we did not over-fit on our training data.

4 Initial Model Evaluation

4.1 Evaluation Metrics

The main heuristic governing our player model is the maximisation of the number of trips awarded. We stated earlier that this should allow us to maximise the total time hired and consequently maximise the total earnings.

As such, the three key evaluation metrics that we look at include:

- Total Earnings: Our player's total earnings (in USD) after a week's simulation of trips. The main objective of this project is to maximise this metric since it is the strongest indicator of a player's profitability.

- Total Time Hired: Our player's total time hired (in minutes) after a week's simulation of trips.

  Ideally, we want to be achieving the maximum limit of approximately 60 hours (3600 minutes) - where a maximum of 10 hours can be spent moving passengers within each of the six shifts.

- Total Trips Awarded: The total number of trips awarded to our player during the simulation run.

  This metric allows us to gain some insight about the performance of our model with respect to our underlying heuristic - which aims to maximise the number of trips awarded.

Again, to emphasise, although our model aims to maximise the number of trips awarded; it is more accurate to say that our model aims to stay hired for as long as possible. Our model simply maximises the likelihood of being awarded a trip; such that it will be hired as soon as possible after it becomes available. Therefore, the maximisation of the number of awarded trips does not imply that our model looks for shorter trips; in fact, our model does not distinguish between types of trips.

4.2 Initial Model Performance

We now evaluate the performance of our initial model against a baseline model. Our initial model is described in **Section 2 Initial Model Design**. The baseline model is a random walker based in Manhattan that is always set to “For Hire”. Like our initial player model, its shifts are set for the most active hours of the week.

We run both players through a simulation of 10 games over the test period from 2015-06-01 00:00:00 to 2015-06-07 23:59:59. The evaluation metrics are averaged (using the mean) over the 10 games and displayed in the following table:

Player Type	Earnings (USD)	Time Hired (MIN)	Trips
Random Walker (Baseline)	3237.05	2926.1	228.33
Initial Model	3730.98	3447.01	267.32

Table 2: Initial Model Performances

Evidently, compared to our baseline model, our initial model experiences a large improvement in its performance. Our initial model generates an additional 500 USD, is hired approximately 8.5 hours longer than the baseline model, and is awarded with 40 more trips.

The improvement in the total time hired and the number of awarded trips suggests that our initial model’s routing algorithm is effective. That is, our non-random maneuvering around Manhattan is helping us obtain trips and staying hired.

However, we can still see that our initial model’s total time hired is about 2.5 hours less than the limit of 60 hours. This suggests that there is still room for improvement in terms of maximising the total time hired.

5 Model Refinement

Over the course of this project, we made several adjustments to our initial model. These adjustments all aim to improve either the model performance or model efficiency. The model refinements are described in the following sections. Lastly, in **Section 5.7 Comparison of Models**, we compare the performances of each model version by running a simulation; summarising their results in a table.

5.1 Increasing the Look-Ahead Value ‘k’

In the initial model design, we state that our route model involves an undesirable exponential computation: requiring the computation of up to 9^k possible routes’ probabilities for the next k rounds. After careful examination of the code, we identified a method allowing us to linearise the computation. Consequently, we could set a much higher k value without having to worry about exceeding each round’s five second decision-making time limit.

5.1.1 Linear Computation

We will not go into the specifics of the code; however, we will still summarise our general approach to linearising the computation. By breaking down the code, we identified many redundant and repetitive computations that could otherwise be avoided. We vectorised computations where we could and utilised a dynamic programming approach to remove the redundancies. This allowed us to ultimately achieve a linear time complexity for the route model computation.

5.1.2 Setting k = 10

Although a higher k may seem desirable, we find that its benefits drop-out fairly quickly as k grows. A higher k involves the addition of probabilities that are products of many probabilities. Multiplying any value by a probability below one will inevitably reduce the original value. Therefore, the product of more probabilities will approach zero. As such, the probabilities being added at the end of $Pr(\text{Getting a Trip along } R)$ will be negligible for higher k’s. Thus, these probabilities will have a minimal effect on the overall value of the cumulative probability for each route.

To determine our ‘optimal’ k-value, we ran several simulations with our player model using different k values; comparing the total earnings of each model. We found that there were consistent improvements in total earnings for higher k values when $k \in \{1, 2, ..., 10\}$. However, beyond $k = 10$, and the total earnings essentially stabilises. This is quite possibly where the ‘negligible’ probabilities (mentioned above) start to occur.

Therefore, we chose to set our k look-ahead value to 10. The total earnings essentially stabilise for $k = 10$ and beyond. By taking $k = 10$, we have a good trade-off between the computational cost and the resultant performance / total earnings. We also note that $k = 10$ allows each round's decision to be made well within one second.

5.2 Frequency Poisson Model

For our route model, we use a look-up table to determine the expected frequency of a cell for a specified time-interval. However, the look-up table is based entirely on past data and takes up 600KB of RAM whenever we need to make a decision for each round. Therefore, the look-up table is undesirable in the sense that it does not generalise and it uses a bit of RAM.

In an effort to generalise the past trip data and save RAM, we used the look-up table as the training data-set for a Poisson Regression model. We chose this particular model since the look-up table supplies expected trip frequencies; in other words, 'count' data, which is very suitable for Poisson Regression. We note that since we take the average frequencies, we often encounter decimal values; therefore, we initially round the frequencies to the closest integers so that our frequencies are all non-negative integers.

Before training the model, we use one hot encoding on our categorical predictors:

- 10-Minute Interval of the Day
- Weekday or Weekend
- Taxi Zone

In a similar manner to the look-up table, the trained Poisson Regression model can predict the expected frequency of trips for a particular cell at a specified minute of the weekday or weekend. It divides the predicted trip frequency for the corresponding taxi zone and 10-minute interval by the number of cells in the taxi zone and 10 - to predict the cell's expected trip frequency for the minute.

We observe in the table of results from **Section 5.7** that using a Poisson Regression model to predict the expected frequencies results in similar total earnings when compared to using the look-up table directly. This is a good sign that the Poisson Regression model is generalising well. Additionally, we have improved the efficiency of the model by reducing the required RAM down to 2KB - capturing the look-up table's information in the Poisson Regression model's parameters.

5.3 Look-Up Table. Grouping by Day

From the initial model design's look-up table, we make an adjustment in the grouping of trips. Instead of grouping on the following attributes:

- 10-Minute Interval of the Day
- Weekday or Weekend
- Taxi Zone

...we switch the weekday/weekend attribute with the day of the week. The reasoning is provided in **Section 3.5.2 Frequency Look-Up Table: Day of the Week**. Put briefly, we found evidence to suggest that the day of the week affected the trip frequency distribution across the week. After pre-processing this new look-up table, we re-train our Poisson Regression model using the new look-up table as the training data.

5.4 Re-Scheduling Shifts

At this stage in the model refinement phase, our model was already performing quite well, consistently generating total earnings of around 3900 USD. We could see that our model was effectively performing its underlying purpose: maximising the time hired and the number of awarded trips. We were consistently nearing the limit of 60 hours for the total time hired for each simulation run.

However, we soon identified an issue when our player model was compared against other groups in the nightly games run by the project supervisors. In conjunction to the accidental discovery that our model still achieves a total time hired of around 60 hours with a different set of shift times; this motivated us to re-schedule our shifts during periods of the week when trips are expected to be more profitable.

5.4.1 Nightly Games

A simulation is run each night with all the groups' player models. After each simulation, a table of results is supplied comparing each group's player's performance. Although the total time hired is not supplied in the results, both the mean total earnings and mean number of trips are provided.

In a nightly simulation run on the night of 2019-10-17, our group submitted our player model featuring all the refinements described so far. The simulation used the test data period from 2017-07-10 00:00:00 to 2017-07-16 23:59:59, obtaining the evaluation metrics by averaging over 25 game runs. We observe the results for our group and other competitors - with higher total earnings - in *Table 3*:

Group	Mean Earnings (USD)	Mean Trips
Competitor	4088.73	279.24
Competitor	4069.94	276.76
Our Group	3922.83	305.44

Table 3: Comparison with Competitors

Evidently, despite our player's ability to consistently obtain more trips, this does not translate into a higher average earnings. This suggests that the trips we are awarded with are generally less profitable than the trips awarded to competitors. Thus, we should consider the profitability of a trip and factor this into our model.

5.4.2 Profitability of Trips

We can define the profitability of a trip by its total earnings per minute. By using the total earnings **rate**, we have a normalised value that better informs us of the profitability of a trip. Using the trip's total earnings alone is not a great indicator of profitability since a higher total earnings by itself might merely be implying that the trip duration was longer.

Our current player model does not consider how profitable a trip will be. It simply aims to be awarded a trip as soon as possible whenever it is available, regardless of the trip quality. Therefore, addressing this non-discriminatory behaviour is likely to cause a major change to the underlying heuristic of our model.

Thus, we instead take a simpler approach, working around this issue by changing our shifts to occur when we expect to see the most profitable trips.

5.4.3 Most Profitable Hours of the Week

We identify the most profitable hours of the week in the same manner as identifying the most "active" hours of the week (described in **Section 2.3**). Again, we only look at the distribution of trip total earning rates (TER) within Manhattan across the hours of the week. This is done by averaging the trips' TERs for each hour of the week across the entire data-set period (July 2015 to June 2017). We then visualise these values on a heat-map in *Figure 8*.

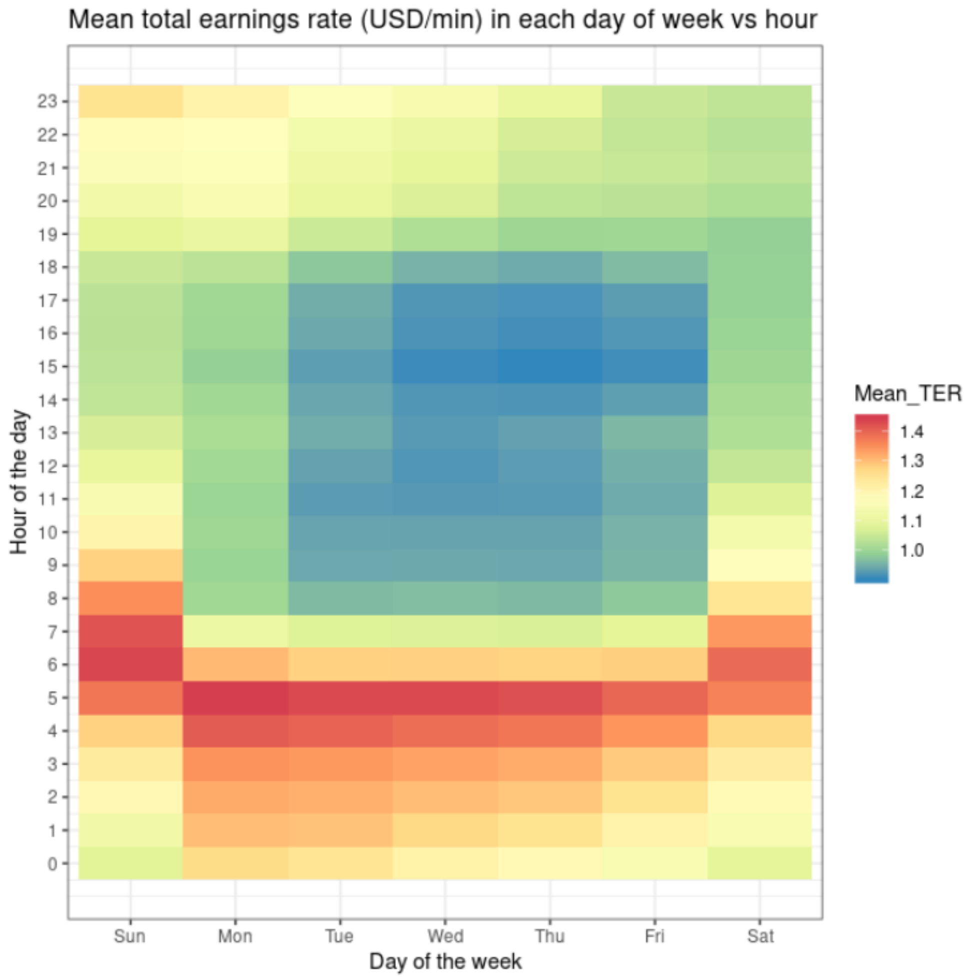

Figure 8: Average (Mean) Total Earnings Rate (USD/MIN) for Each Hour of the Week

We assume that the distribution of trip TERs across the hours of the week are similarly distributed across all weeks within the provided data-set period; and that this also extends to the simulation test data.

From the heat-map in *Figure 8*, we simply "eye-balled" the hours with the most profitable trips and set them as our shift times; hard-coding the corresponding shift start times. We looked for six contiguous 12 hour intervals that covered these profitable periods (indicated by red). Our hard-coded shift times are the following:

1. *Monday 00:00* to *Monday 12:00*
2. *Monday 20:00* to *Tuesday 08:00*
3. *Tuesday 19:00* to *Wednesday 07:00*
4. *Wednesday 19:00* to *Thursday 07:00*
5. *Saturday 00:00* to *Saturday 12:00*
6. *Sunday 00:00* to *Sunday 12:00*

After hard-coding these shift times, we ran several local simulations, and consistently earned at least 100 USD more than our previous model. Additionally, our total time hired was still very close to the maximum of 60 hours despite not working during the most active hours. This suggests that the shift time has a negligible effect on our model's ability to stay hired.

5.5 Look-Up Table. 5-Minute Intervals

Currently, our model's Poisson Regression model for predicting the expected frequency is trained on the look-up table with the following grouping:

- 10-Minute Interval of the Day
- Day of the Week
- Taxi Zone

We further refine the 10-minute intervals into five-minute intervals. An example of a five-minute interval is provided in **Section 3.5.3 Frequency Look-Up Table: Five-Minute Intervals**. A new look-up table is generated from the four half-year data-sets using this new grouping. We then re-train our Poisson Regression model using this new look-up table.

The main motivation behind the 5-minute interval is to allow our Poisson Regression model to predict more accurate expected frequencies for our route model. This will allow our route model to make more informed decisions, which should help enhance the performance of our player model. We also consider the possibility of over-fitting in **Section 3.5.3**. However, the improvement in performance indicates that we are not over-fitting on our training data when we reduce the interval size to five minutes.

5.6 Final Model

5.6.1 Feature Description

With the above improvements, we arrive at our final model. To summarise, our final model has the following features:

- Based in Manhattan; re-routing back into Manhattan after dropping off a passenger in a different borough.
- Always "For-Hire".
- Route model (described in **Section 2.4**) to determine the next round's move with a look-ahead k value of 10.
- Poisson Regression model - built on a look-up table that groups trips across the 2015 July to 2017 June period on the trip's five-minute interval of the week, day of the week, and taxi zone - that predicts and provides the requested expected frequencies to the route model.

5.6.2 Model Diagram

A flow chart briefly summarising the final model can be found at *Figure 9*.

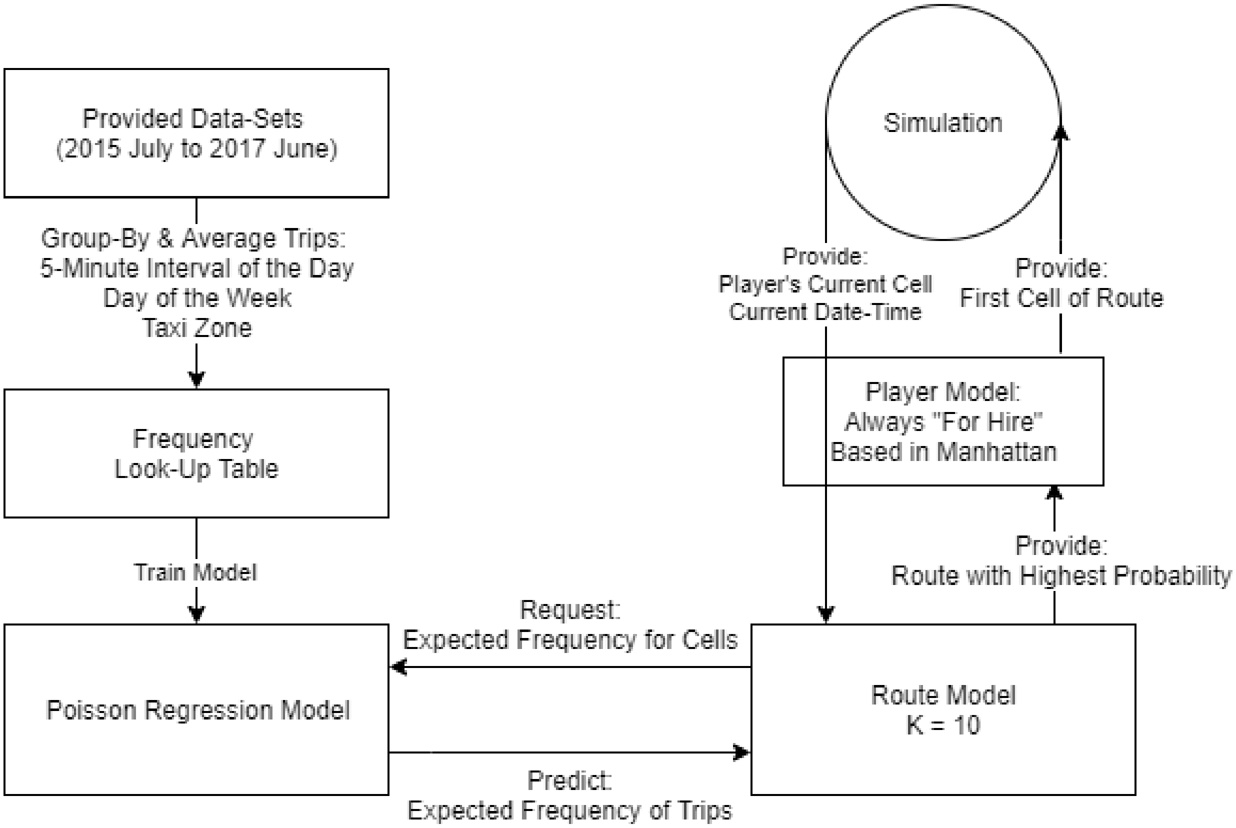

Figure 9: Flow Chart of Final Model

5.7 Comparison of Models

Simulations were run throughout the project for each successive model refinement to assess changes in performance. However, to make the comparison more convenient, we run all versions of our model on a single simulation. Each version corresponds to the refinements discussed above. We run the following model versions - named by their corresponding refinement:

- *Baseline*: The Manhattan random walker.
- *Initial*: The initial model as described in **Section 2 Initial Model Design**.
- *k=10*: Changing the look-ahead value from 3 to 10.
- *Poisson*: Using a Poisson Regression model to predict the expected frequencies that are supplied to the route model.

- *Day-Week*: Changing the grouping of the look-up table. From weekday/weekend to the trip's day of the week. Then re-training our Poisson Regression model on this new look-up table.

- *Shift-Change*: Re-scheduling shifts to the most profitable hours of the week.

- *Final*: Changing the grouping of the look-up table. From 10-minute intervals to five-minute intervals. Then re-training our Poisson Regression model on this new look-up table. This is the final model.

We run each model version through a simulation of 10 games over the test period from 2015-06-01 00:00:00 to 2015-06-07 23:59:59. The evaluation metrics are averaged (using the mean) over the 10 games and displayed in the following table:

Model Version	Earnings (USD)	Time Hired (MIN)	Trips
Baseline	3123.43	2844.54	225.36
Initial	3730.98	3447.01	267.32
k=10	3847.59	3464.31	303.83
Poisson	3899.39	3493.51	303.97
Day-Week	3938.38	3584.31	306.33
Shift-Change	4109.09	3593.12	291.67
Final	4142.12	3595.25	317.67

Table 4: Performance Comparison on Different Model Versions

Evidently, we see a gradual increase in the total earnings for each successive model refinement. We observe a large jump of 1.5 hours in the total time hired when we changed the grouping of our look-up table to the trip's day of the week. And from there, a gradual increase, approaching the limit of 60 hours (3600 minutes). Our final model is just short of the limit by five minutes; doing a great job in maximising the total time hired.

We note the drop in the number of awarded trips when we change our shifts from the most active hours to the most profitable hours. Despite this drop in trips, the total time hired is still higher for the more profitable shifts. This might suggest that active hours feature shorter trips than those occurring during more profitable hours.

6 Conclusion

In the last nightly game simulation, we managed to secure the top place in terms of total earnings. The results for the top five groups can be found in the following table:

Group	Mean Earnings (USD)	Mean Trips
Our Group	4070.84	305.52
Competitor	3999.95	253.28
Competitor	3881.73	269.08
Competitor	3811.53	262.24
Competitor	3798.46	250.28

Table 5: Comparison with Competitors

Our group's player model leads with a mean earnings of approximately 70 USD. We note that our final model is consistently awarded with a significantly higher number of trips: around 35 more trips than the second highest (269.08). Although we do not know the specifics of each trip, a higher number trips suggests that our model is performing as intended. That is, it is effectively finding trips to maximise the total time hired.

Over the course of the project, we explored many different approaches with the ultimate aim of maximising the total earnings (fare amount + tips) of a yellow taxi driver in New York City. We started with a simple initial model and persistently explored potential areas of improvement. By consistently making small refinements to our model throughout the project, we arrived at a final player model with one of the highest total earnings among our competitors.

7 Limitations & Future Improvements

7.1 Realistic Applications

Although the project's aim is to build a predictive model that will maximise the total earnings of a yellow taxi driver in New York City; there are several limitations that prevent our player model from being applied in a realistic scenario. In other words, limitations that suggest our model will not perform so well for today's New York City yellow taxi driver.

7.1.1 Testing Data-Set Period

To evaluate the performance of our player, we essentially run it through a simulation of a week's worth of past trip data. This past trip data is either from the period covering January 2015 to June 2015, or July 2017 to December 2017. Therefore, the test data itself is either four or two years old.

The main concern about testing on past trip data is that it assumes similar trends in trips occurring today. However, this is unlikely the case, especially with the rising popularity of competing "for-hire" services such as Uber. It is quite possible that there are significantly less yellow and green taxi trips in today's NYC compared to 2015 or 2017.

Therefore, a more "realistic" evaluation of a player model can be attained by simulating on more recent data (e.g. 2019). However, this also suggests that the models should be trained on more recent data as well - assuming trends have changed over the years.

7.1.2 Simulation Player Count

Each nightly game simulation runs a single player from each group. With 13 groups, this implies that 13 players will be placed in each simulation. However, it is unlikely to expect that only 13 yellow taxis operate within the entirety of New York City.

In a realistic scenario, competition among NYC taxis is likely to be intense. We are more likely to see taxis congregate at locations that generally feature more pick-ups. However, by simulating with only 13 players, we remove a lot of competition.

Therefore, the total earnings for each simulation is likely overestimated due to the lack of competition. A possible work-around is to incorporate enough players to make the simulation more realistic; however, this scenario is likely infeasible, considering the extra strain for simulating each additional player.

7.1.3 Unrecorded Cash Tips

We note that the data provided by the TLC states that tips for taxi trips paid by cash are not recorded - being set to zero. Since the test data contains trips paid by cash; it is likely that the total earnings underestimates the actual total earnings - since cash tips are not recorded.

7.2 Model Inadequacies

7.2.1 Distinguishing Trips

Based on past data, it is possible to predict what “type” of trips occur at certain cells for particular date-times. For instance, there may be a location cell where taxi’s often pick-up passengers that wish to be transported from the airport to the city. Thus, based on past trip data, a player model can choose to prioritise such trips.

However, our player model does not distinguish trips in such a manner. Instead, it only tries to find a trip as soon as possible once it becomes available - with the intention of staying hired for as long as possible. In **Section 5.4**, we identify an issue where we are picking up more trips than competitors, but generating a lower total earnings. This suggested that our player could be improved by prioritising more profitable trips; in other words, choosing to distinguish between trips.

We do not end up implementing a means of prioritising more profitable trips in our final model. Instead, we merely change our shift times to include the more profitable hours of the week. Therefore, our model still aims to find a trip as soon as possible, without considering the quality of the trip.

We did some initial exploration in the idea of distinguishing trips. The main method in implementing such a feature would be to modify our route model; particularly the probability of a route. Instead of a probability, we could consider a scoring function which also considers the expected profitability of a cell in addition to the expected frequency of the cell.

Due to time limitations, we were not able to explore this feature in more depth. However, this is definitely a possible area to consider for improving model performance.

7.2.2 Taxi Zones

The look-up tables are grouped on the taxi zones rather than the individual cells. As such, we make the assumption that a cell’s taxi zone represents the cell’s expected frequency of trips for a specified time interval. However, despite assuming this relationship for all our models, we were generally skeptical about it.

Instead, we were considering using clusters instead of taxi zones. That is, identifying clusters of cells within Manhattan that share similar expected trip frequencies for a specified time interval. We would then group our look-up table using the clusters instead of the taxi zones; since the clusters are generated to represent their cells' expected trip frequencies.

Although we were not able to explore this area in more detail, we are confident that using clusters instead of taxi zones can help improve the performance of our model - by supplying more accurate expected trip frequencies.

8 Bibliography

[1] New York City Taxi and Limousine Commission. (2018). Data Dictionary – Yellow Taxi Trip Records. Retrieved from
`https://www1.nyc.gov/assets/tlc/downloads/pdf/data_dictionary_trip_records_yellow.pdf`

[2] New York City Taxi and Limousine Commission. (2019). TLC Trip Records User Guide. Retrieved from
`https://www1.nyc.gov/assets/tlc/downloads/pdf/trip_record_user_guide.pdf`