Pricing Case Study

2022-12-16

Contents

Pricing Case Study

You're launching a ride-hailing service that matches riders with drivers for trips between the Toledo Airport and Downtown Toledo. It'll be active for only 12 months. You've been forced to charge riders $ 30 for each ride. You can pay drivers what you choose for each individual ride.

The supply pool ("drivers") is very deep. When a ride is requested, a very large pool of drivers see a notification informing them of the request. They can choose whether or not to accept it. Based on a similar ride-hailing service in the same market, you have some data on which ride requests were accepted and which were not. (The PAY column is what drivers were offered and the ACCEPTED column reflects whether any driver accepted the ride request.)

The demand pool ("riders") can be acquired at a cost of $ 30 per rider at any time during the 12 months. There are 10,000 riders in Toledo, but you can't acquire more than 1,000 in a given month. You start with 0 riders. "Acquisition" means that the rider has downloaded the app and may request rides. Requested rides may or may not be accepted by a driver. In the first month that riders are active, they request rides based on a Poisson distribution where lambda = 1. For each subsequent month, riders request rides based on a Poisson distribution where lambda is the number of rides that they found a match for in the previous month. (As an example, a rider that requests 3 rides in month 1 and finds 2 matches has a lambda of 2 going into month 2.) If a rider finds no matches in a month (which may happen either because they request no rides in the first place based on the Poisson distribution or because they request rides and find no matches), they leave the service and never return.

Submit a written document that proposes a pricing strategy to maximize the profit of the business over the 12 months. You should expect that this singular document will serve as a proposal for - A quantitative executive team that wants to know how you're thinking about the problem and what assumptions you're making but that does not know probability theory - Your data science peers so they can push on your thinking

Please submit any work you do, code or math, with your solution.

General Thoughts

The general approach that I have is to run a simulation of the case and then find out at what price point leads You can do something like linear optimization but do the the randomness in the distribution simulation would be best approach.

Methodology and Approach

To get this working, we would first need to get a simulation of potential profit for a price (Step 1) and then run that simulation for different prices (Step 2).

1. For Step 1: Simulate out profit at a particular price.

- The first step is to understand the probability of someone being able to pick up the ride. Basically, given a price that we pay to a driver, what is the probability that this person would pick it up. We can do this through a logistic regression model.
- Write out the business math and customer lifetime value from acquiring and number of times riden. So get your 30 dollars subtract your payout and multiply that by number of rides and repeat for the
- write out simulation

2. Examine Simulation to see price vs profit trade off and pick maximum point. From the above run a simulation on this but with varying rpcies and seeing where the points are.

Assumptions

- (Stated) Price that Charge Riders is \$30. Price that provide drivers ($\$30 - PricePaidToDriver$)
- (Assumed) Assuming no seasonality and also that Price Paid To Driver is a fixed amount. No dynamic pricing or price discrimination, just one price throughout. Everyone is just getting paid a flat amount.
- (Assumed and Stated) you expect to get 1000 new people every month until you have acquired all 10,000.
- Using mean prediction of logistic regression model is enough, a random variable around the SE would have been more approrpiate but also unnecessary at this stage of interview.
- (Assumption) Probability of customer renewal happens end of the month and not after every ride.
- (Assumption) That there is no limit to the number of drivers that can be fulfilled. For example, assuming that there are 10 drivers, but request for 15 that it can still be fulfilled, cause lets say throughout day. Timing is not a factor this is just done on an aggregated leve.
- Assumption we can not get customer again after losing them. No re acquiring.

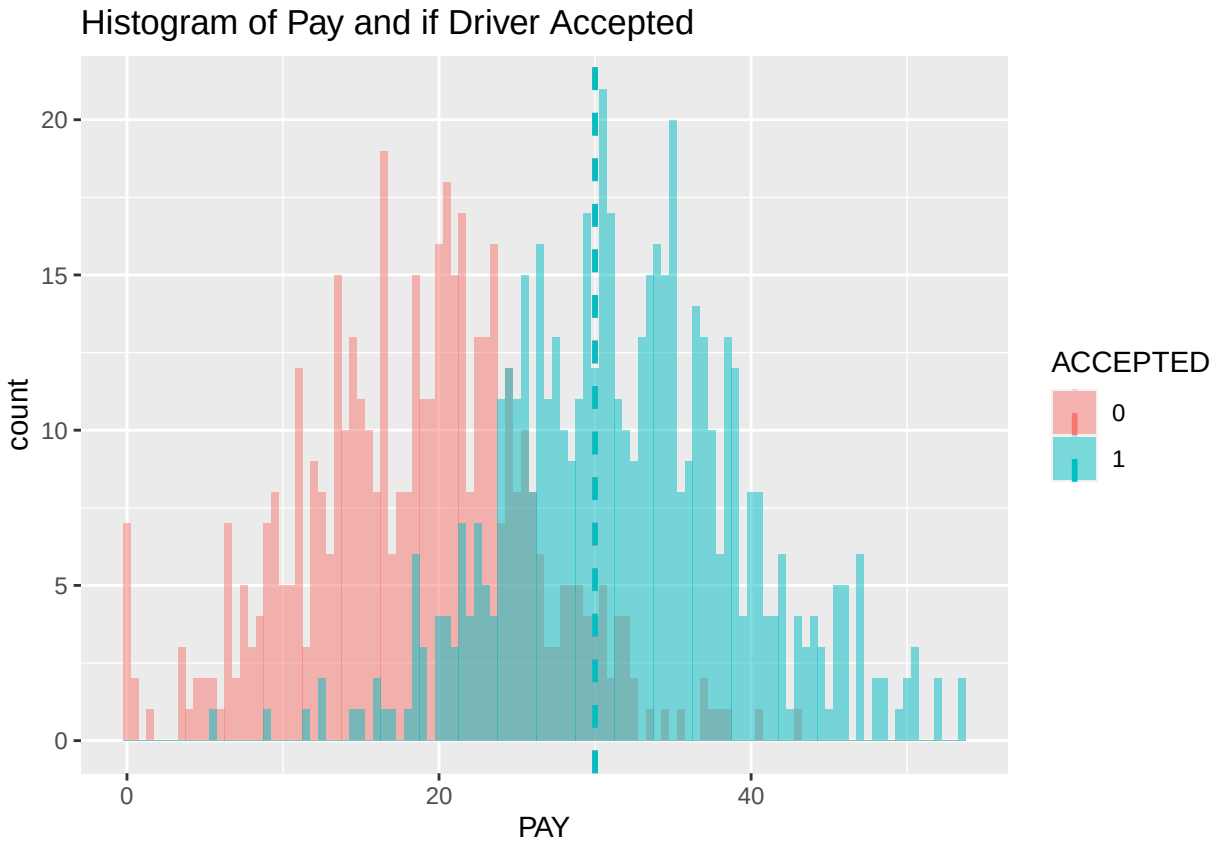
Understanding Price of Driver Acceptance

Logistic Regression Model

EDA

In this area, will explore what is the price point that a driver will need to be paid to accept a ride.

```
driverData <- read.csv("~/Downloads/driverAcceptanceData - driverAcceptanceData.csv")
driverData$Numeric_ACCEPTED <- driverData$ACCEPTED
driverData$ACCEPTED <- as.factor(driverData$ACCEPTED)

ggplot(data = driverData, aes(x = PAY, fill = ACCEPTED)) +
  geom_histogram(binwidth=.5, alpha=.5, position="identity") + ggtitle("Histogram of Pay and if Driver A
  geom_vline( aes(xintercept=30,  colour=ACCEPTED),
              linetype="dashed", size=1)
```




```
ggplot(data = driverData, aes(x = PAY, fill = ACCEPTED)) + geom_density(alpha = 0.2) + ggtitle("Distribu
  geom_vline( aes(xintercept=30,  colour=ACCEPTED),
              linetype="dashed", size=1)
```

So over here we see the differences in distirbution between the amount that Visually, it looks like after the $25 mark we start getting people more people start accepting than not.

Descriptive Statistics on the Distributions

```
describeBy(driverData[,2:3], group="ACCEPTED")
```

```
##
##  Descriptive statistics by group
## ACCEPTED: 0
##           vars   n  mean   sd median trimmed  mad min   max range  skew
## PAY          1 473 18.62 7.41  19.04   18.65 7.09   0 43.15 43.15 -0.01
## ACCEPTED*    2 473  1.00 0.00   1.00    1.00 0.00   1  1.00  0.00   NaN
##           kurtosis   se
## PAY           0.14 0.34
## ACCEPTED*      NaN 0.00
## ------------------------------------------------------------
## ACCEPTED: 1
##           vars   n  mean   sd median trimmed  mad min   max range skew kurtosis
## PAY          1 527 32.08 7.54  31.69   31.93 7.47 5.4 53.67 48.26  0.1     0.25
## ACCEPTED*    2 527  2.00 0.00   2.00    2.00 0.00 2.0  2.00  0.00  NaN      NaN
##             se
## PAY       0.33
## ACCEPTED* 0.00
```

Numeric Statistics also show the mean and median of the distirbutions and you can see that the centers are apart. Interestingly they both have close SD.

Going back to earlier the designated max price that charging a rider is $30. From that you need to figure out how much you are going to pocket and then pay the driver. Remember your not going to get drivers at a low price.

So you need to create a probability that a driver would accept the ride given

$$P(AriverAcceptance) = f(PayTheyWouldRecieve)$$

This seems something that can be modelled via a logistic regression model and can be incorporated into calculation. This would be a great predictive model.

Logistic Regression

```
mylogit <- glm(ACCEPTED ~ PAY, data = driverData, family = "binomial")
summary(mylogit)
```

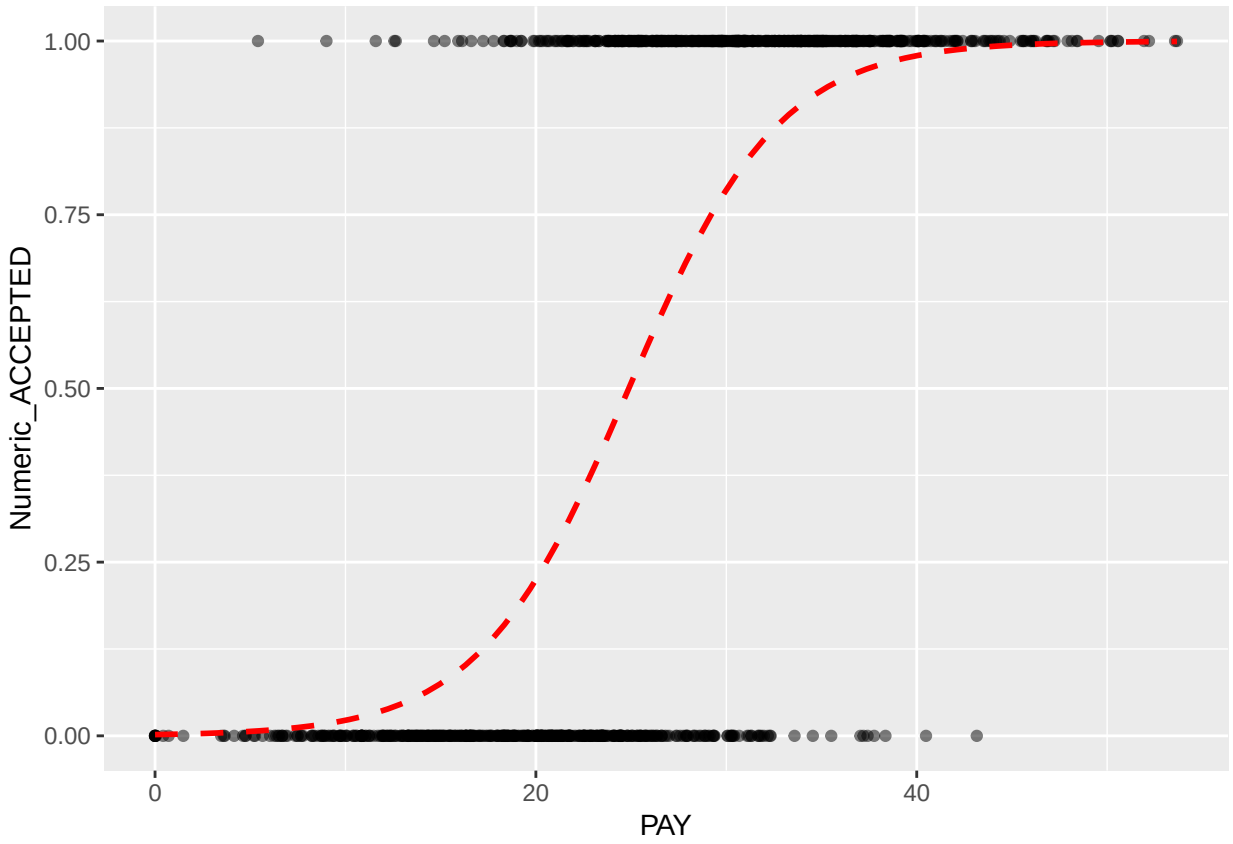
```
##
## Call:
## glm(formula = ACCEPTED ~ PAY, family = "binomial", data = driverData)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.0493  -0.6014   0.1017   0.6159   3.1468
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -6.31581    0.40859  -15.46   <2e-16 ***
## PAY          0.25386    0.01589   15.98   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1383.4  on 999  degrees of freedom
## Residual deviance:  790.0  on 998  degrees of freedom
## AIC: 794
##
## Number of Fisher Scoring iterations: 5
```

Interpretation and Visualization As the pay increases by 1 dollar, the log odds increases by a factor of .25, or put it another way. As the pay increases by 1 dollar, the odds increase by a factor of .28.

It would be alot easier to see this visually from below.

```
ggplot(driverData, aes(x=PAY, y=Numeric_ACCEPTED)) +
  geom_point(alpha=.5) +
  stat_smooth(method="glm", se=FALSE, method.args = list(family="binomial"),
              col="red", lty=2)
```

```
## `geom_smooth()` using formula 'y ~ x'
```



You can see the nonlinearity of the curve, and how the propensity accelerates towards the middle and then flattens towards the ends.You can see that the breaking point (50% mark) is around a pay of 25.

Output of Logistic Regression and Comparison of Linear Model

Using the boot function, showcasing the probability of driver accepting at the 25 to 30 mark with increments of 1.

```
inv.logit(-6.31581+.25386*20:30)
```

```
##  [1] 0.2246780 0.2719503 0.3249995 0.3829537 0.4444379 0.5076719 0.5706614
##  [8] 0.6314425 0.6883185 0.7400311 0.7858333
```

As a baseline, i am also running a linear model here. A linear model is not appropriate because the data violates many of the assumptions of LM like linearity, but it has a nice baseline interpretation that I would like to use as a reference. The linear model is awful and incorrect, but says that a $1 of pay increases acceptance by average 3%. This model is not realistic but it is nice to have as a comparison for interpretation because we see the logistic regression model is not straightline.

```
lm_model <- lm(Numeric_ACCEPTED~PAY,data = driverData)
summary(lm_model)
```

```
##
## Call:
## lm(formula = Numeric_ACCEPTED ~ PAY, data = driverData)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.10643 -0.30045  0.01589  0.30098  1.14772
```

```
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.32724    0.03230  -10.13   <2e-16 ***
## PAY          0.03322    0.00117   28.39   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3717 on 998 degrees of freedom
## Multiple R-squared:  0.4469, Adjusted R-squared:  0.4463
## F-statistic: 806.3 on 1 and 998 DF,  p-value: < 2.2e-16
```

Why is the logistic regression model so important to have? Having a model that can predict if a person would be used in the simulation.

Simulation

I set some sanity checks at a price of 30 and 0 to mean total profit = 0. Why the above? If the max we charge is $30 and the max that gets paid is $30 then we are left with 0. While the biggest profit margin is if people riders provide rides for free, although there is some probability of that happening. I would not rely on that and I would put profit around 0 for there. In actually it would be higher but I am putting it as one of the lowest points. Just from intuition I imagine that the peak would be 23-28. It's essentially the trade off between the probability of acceptance and also profit from ride.

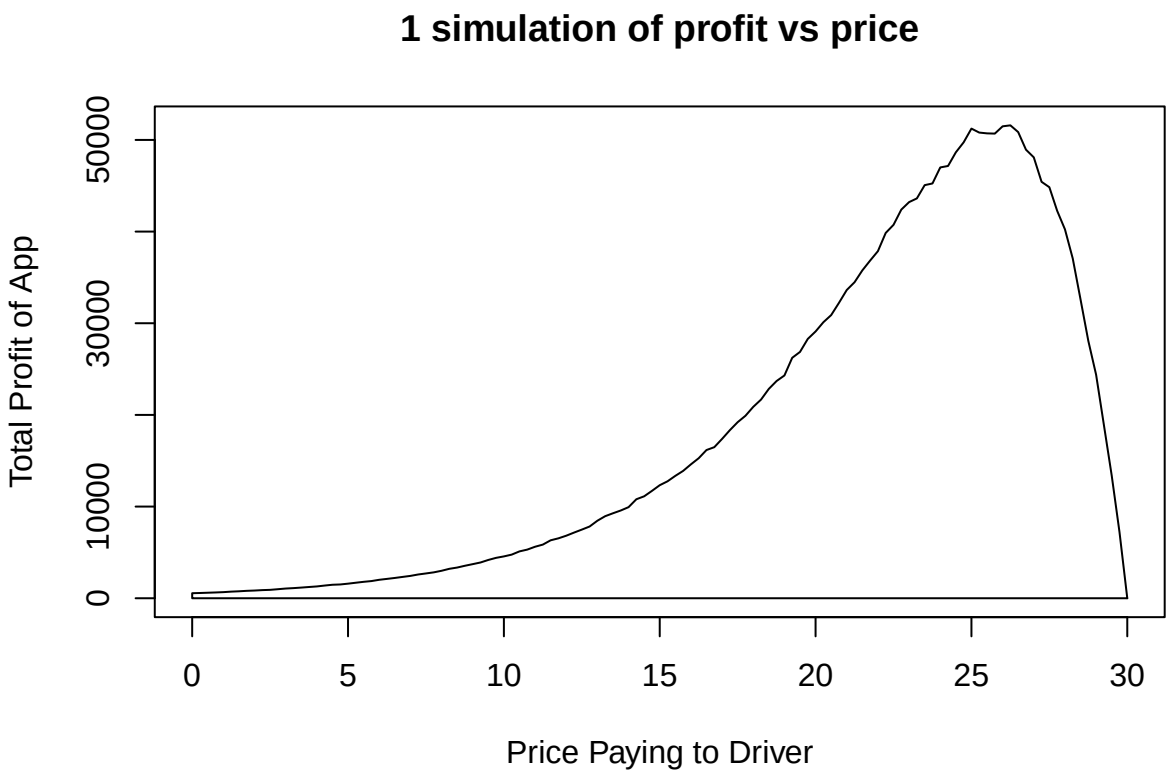
Equations to keep in mind:

- probability_of_match = f(cost_to_driver)
- Profit_to_company = 30-cost_to_driver
- quantity_to_company = probability_of_match × Number of rides
- cost_to_rider <- 30
- probability_of_renewal <- inv.logit(-6.31581+.25386*30)
- price ~ profit curve

Looks like the 25-26 is the peak, but can run some more simulations. To save time, can focus on the area that looks like the peak and if we ran simuluation 50 times what the average would look like.

```r
my_result <- data.frame(price = c(30,0), total_profit = c(0,0))
my_result_total <- list()

my_lambda <- function(x){
  # Function to return 1 random number from a different poisson distribution
  return(rpois(1,x))
}



my_result <- data.frame(price = c(30,0), total_profit = c(0,0))

for (cost_to_driver in seq(0,30,.25)){
    Profit_to_company = 30-cost_to_driver
    probability_of_getting_ride <- inv.logit(-6.31581+.25386*cost_to_driver)
    # I can apply the spectrum or I can apply the average.
    # I would want to capture the variability, but for sake of ease just do percentage


    # Month 1
    RiderRequest_Month1 <- rpois(1000, 1) #First 1000
    RiderRidesApprovedMonth1 <- probability_of_getting_ride*RiderRequest_Month1

    #Month 2
    RiderRequest_Month2 <- rpois(1000, 1) #First 1000
    NewRiderRidesApprovedMonth2 <- probability_of_getting_ride*RiderRequest_Month2
    #Applying Churn to old customer base
    TotalRiderRidesApprovedMonth2 <- c(probability_of_getting_ride*sapply(RiderRidesApprovedMonth1,my_la

    #Month3
    RiderRequest_Month3 <- rpois(1000, 1) #First 1000
    NewRiderRidesApprovedMonth3 <- probability_of_getting_ride*RiderRequest_Month3
    TotalRiderRidesApprovedMonth3 <- c(probability_of_getting_ride*sapply(TotalRiderRidesApprovedMonth2

    #Month4
    RiderRequest_Month4 <- rpois(1000, 1) #First 1000
    NewRiderRidesApprovedMonth4 <- probability_of_getting_ride*RiderRequest_Month4
    TotalRiderRidesApprovedMonth4 <- c(probability_of_getting_ride*sapply(TotalRiderRidesApprovedMonth3

    #Month5
    RiderRequest_Month5 <- rpois(1000, 1) #First 1000
    NewRiderRidesApprovedMonth5 <- probability_of_getting_ride*RiderRequest_Month5
    TotalRiderRidesApprovedMonth5 <- c(probability_of_getting_ride*sapply(TotalRiderRidesApprovedMonth4

    #Month6
    RiderRequest_Month6 <- rpois(1000, 1) #First 1000
    NewRiderRidesApprovedMonth6 <- probability_of_getting_ride*RiderRequest_Month6
    TotalRiderRidesApprovedMonth6 <- c(probability_of_getting_ride*sapply(TotalRiderRidesApprovedMonth5

    #Month7
    RiderRequest_Month7 <- rpois(1000, 1) #First 1000
    NewRiderRidesApprovedMonth7 <- probability_of_getting_ride*RiderRequest_Month7
    TotalRiderRidesApprovedMonth7 <- c(probability_of_getting_ride*sapply(TotalRiderRidesApprovedMonth6
```

```
    #Month8
    RiderRequest_Month8 <- rpois(1000, 1) #First 1000
    NewRiderRidesApprovedMonth8 <- probability_of_getting_ride*RiderRequest_Month8
    TotalRiderRidesApprovedMonth8 <- c(probability_of_getting_ride*sapply(TotalRiderRidesApprovedMonth7

    #Month9
    RiderRequest_Month9 <- rpois(1000, 1) #First 1000
    NewRiderRidesApprovedMonth9 <- probability_of_getting_ride*RiderRequest_Month9
    TotalRiderRidesApprovedMonth9 <- c(probability_of_getting_ride*sapply(TotalRiderRidesApprovedMonth8

    #Month10
    RiderRequest_Month10 <- rpois(1000, 1) #First 1000
    NewRiderRidesApprovedMonth10 <- probability_of_getting_ride*RiderRequest_Month10
    TotalRiderRidesApprovedMonth10 <- c(probability_of_getting_ride*sapply(TotalRiderRidesApprovedMonth9

    #Month11 and 12
    TotalRiderRidesApprovedMonth11 <- c(probability_of_getting_ride*sapply(TotalRiderRidesApprovedMonth
    TotalRiderRidesApprovedMonth12 <- c(probability_of_getting_ride*sapply(TotalRiderRidesApprovedMonth


    ProfitM1 <- sum(RiderRidesApprovedMonth1)*(Profit_to_company)
    ProfitM2 <- sum(TotalRiderRidesApprovedMonth2)*(Profit_to_company)
    ProfitM3 <- sum(TotalRiderRidesApprovedMonth3)*(Profit_to_company)
    ProfitM4 <- sum(TotalRiderRidesApprovedMonth4)*(Profit_to_company)
    ProfitM5 <- sum(TotalRiderRidesApprovedMonth5)*(Profit_to_company)
    ProfitM6 <- sum(TotalRiderRidesApprovedMonth6)*(Profit_to_company)
    ProfitM7 <- sum(TotalRiderRidesApprovedMonth7)*(Profit_to_company)
    ProfitM8 <- sum(TotalRiderRidesApprovedMonth8)*(Profit_to_company)
    ProfitM9 <- sum(TotalRiderRidesApprovedMonth9)*(Profit_to_company)
    ProfitM10 <- sum(TotalRiderRidesApprovedMonth10)*(Profit_to_company)
    ProfitM11 <- sum(TotalRiderRidesApprovedMonth11)*(Profit_to_company)
    ProfitM12 <- sum(TotalRiderRidesApprovedMonth12)*(Profit_to_company)

    YearlyProfit <- sum(ProfitM1,ProfitM2, ProfitM3, ProfitM4, ProfitM5,
                        ProfitM6, ProfitM7, ProfitM8, ProfitM9, ProfitM10, ProfitM11, ProfitM12 )
    my_result <- rbind(my_result, data.frame(price = cost_to_driver, total_profit = YearlyProfit))
  }


plot(my_result$price, my_result$total_profit, type = "l", xlab = "Price Paying to Driver", ylab = "Tota
```




```
kable(my_result %>% arrange(desc(total_profit)) %>% head(20))
```

price	total_profit
26.25	51589.50
26.00	51482.79
25.00	51234.25
26.50	50853.78
25.25	50801.34
25.50	50716.11
25.75	50689.13
24.75	49736.81
26.75	48934.77
24.50	48646.06
27.00	48098.87
24.25	47158.07
24.00	46999.31
27.25	45427.54
23.75	45249.38
23.50	45068.85
27.50	44841.86
23.25	43618.95
23.00	43215.18
22.75	42397.53

Rerunning simulation 50 times

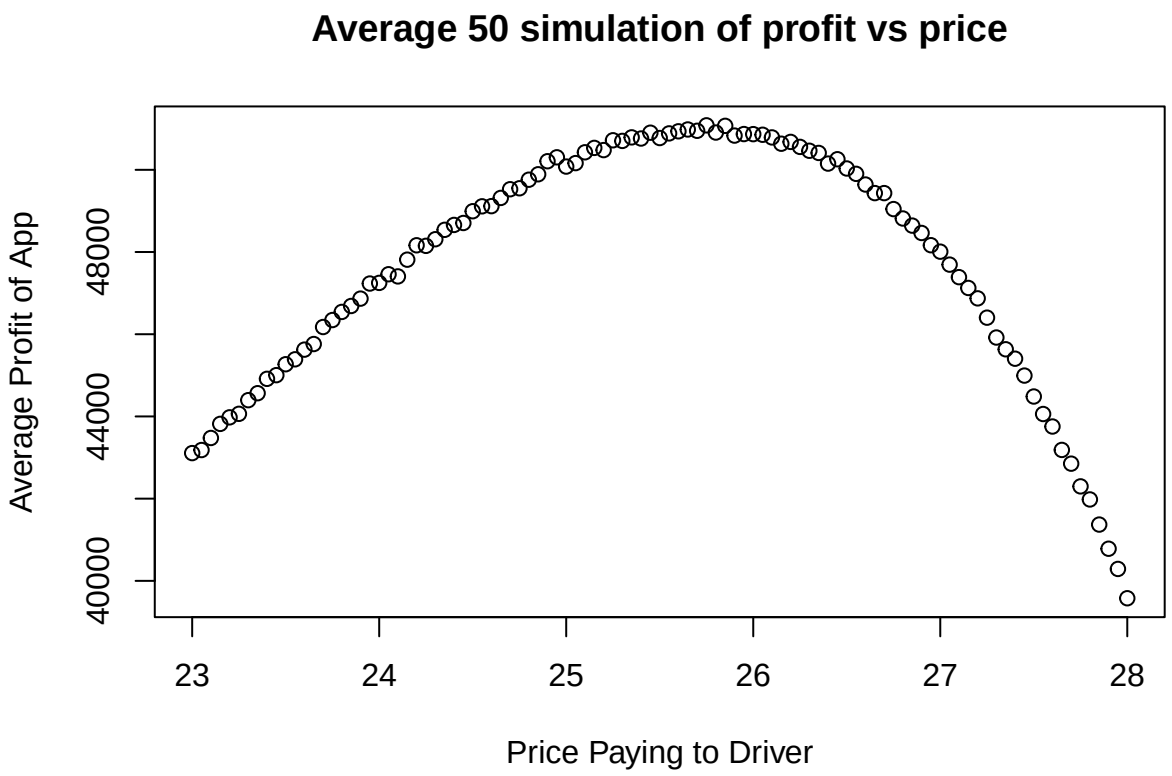
Ran for range 23 to 28 (Save on computational time)

```
my_result2 <- data.frame(price = c(30), total_profit = c(0))
my_lambda <- function(x){
  # Function to return 1 random number from a different poisson distribution
  return(rpois(1,x))
}
for (i in 1:50){
  for (cost_to_driver in seq(23,28,.05)){
      Profit_to_company = 30-cost_to_driver
      probability_of_getting_ride <- inv.logit(-6.31581+.25386*cost_to_driver)
      # I can apply the spectrum or I can apply the average.
      # I would want to capture the variability, but for sake of ease just do percentage


      # Month 1
      RiderRequest_Month1 <- rpois(1000, 1) #First 1000
      RiderRidesApprovedMonth1 <- probability_of_getting_ride*RiderRequest_Month1

      #Month 2
      RiderRequest_Month2 <- rpois(1000, 1) #First 1000
      NewRiderRidesApprovedMonth2 <- probability_of_getting_ride*RiderRequest_Month2
      #Applying Churn to old customer base
      TotalRiderRidesApprovedMonth2 <- c(probability_of_getting_ride*sapply(RiderRidesApprovedMonth1,my

      #Month3
      RiderRequest_Month3 <- rpois(1000, 1) #First 1000
      NewRiderRidesApprovedMonth3 <- probability_of_getting_ride*RiderRequest_Month3
      TotalRiderRidesApprovedMonth3 <- c(probability_of_getting_ride*sapply(TotalRiderRidesApprovedMonth

      #Month4
      RiderRequest_Month4 <- rpois(1000, 1) #First 1000
      NewRiderRidesApprovedMonth4 <- probability_of_getting_ride*RiderRequest_Month4
      TotalRiderRidesApprovedMonth4 <- c(probability_of_getting_ride*sapply(TotalRiderRidesApprovedMonth

      #Month5
      RiderRequest_Month5 <- rpois(1000, 1) #First 1000
      NewRiderRidesApprovedMonth5 <- probability_of_getting_ride*RiderRequest_Month5
      TotalRiderRidesApprovedMonth5 <- c(probability_of_getting_ride*sapply(TotalRiderRidesApprovedMonth

      #Month6
      RiderRequest_Month6 <- rpois(1000, 1) #First 1000
      NewRiderRidesApprovedMonth6 <- probability_of_getting_ride*RiderRequest_Month6
      TotalRiderRidesApprovedMonth6 <- c(probability_of_getting_ride*sapply(TotalRiderRidesApprovedMonth

      #Month7
      RiderRequest_Month7 <- rpois(1000, 1) #First 1000
      NewRiderRidesApprovedMonth7 <- probability_of_getting_ride*RiderRequest_Month7
      TotalRiderRidesApprovedMonth7 <- c(probability_of_getting_ride*sapply(TotalRiderRidesApprovedMonth

      #Month8
      RiderRequest_Month8 <- rpois(1000, 1) #First 1000
      NewRiderRidesApprovedMonth8 <- probability_of_getting_ride*RiderRequest_Month8
      TotalRiderRidesApprovedMonth8 <- c(probability_of_getting_ride*sapply(TotalRiderRidesApprovedMonth
```

```
      #Month9
      RiderRequest_Month9 <- rpois(1000, 1) #First 1000
      NewRiderRidesApprovedMonth9 <- probability_of_getting_ride*RiderRequest_Month9
      TotalRiderRidesApprovedMonth9 <- c(probability_of_getting_ride*sapply(TotalRiderRidesApprovedMonth

      #Month10
      RiderRequest_Month10 <- rpois(1000, 1) #First 1000
      NewRiderRidesApprovedMonth10 <- probability_of_getting_ride*RiderRequest_Month10
      TotalRiderRidesApprovedMonth10 <- c(probability_of_getting_ride*sapply(TotalRiderRidesApprovedMon

      #Month11 and 12
      TotalRiderRidesApprovedMonth11 <- c(probability_of_getting_ride*sapply(TotalRiderRidesApprovedMon
      TotalRiderRidesApprovedMonth12 <- c(probability_of_getting_ride*sapply(TotalRiderRidesApprovedMon


      ProfitM1 <- sum(RiderRidesApprovedMonth1)*(Profit_to_company)
      ProfitM2 <- sum(TotalRiderRidesApprovedMonth2)*(Profit_to_company)
      ProfitM3 <- sum(TotalRiderRidesApprovedMonth3)*(Profit_to_company)
      ProfitM4 <- sum(TotalRiderRidesApprovedMonth4)*(Profit_to_company)
      ProfitM5 <- sum(TotalRiderRidesApprovedMonth5)*(Profit_to_company)
      ProfitM6 <- sum(TotalRiderRidesApprovedMonth6)*(Profit_to_company)
      ProfitM7 <- sum(TotalRiderRidesApprovedMonth7)*(Profit_to_company)
      ProfitM8 <- sum(TotalRiderRidesApprovedMonth8)*(Profit_to_company)
      ProfitM9 <- sum(TotalRiderRidesApprovedMonth9)*(Profit_to_company)
      ProfitM10 <- sum(TotalRiderRidesApprovedMonth10)*(Profit_to_company)
      ProfitM11 <- sum(TotalRiderRidesApprovedMonth11)*(Profit_to_company)
      ProfitM12 <- sum(TotalRiderRidesApprovedMonth12)*(Profit_to_company)

      YearlyProfit <- sum(ProfitM1,ProfitM2, ProfitM3, ProfitM4, ProfitM5,
                          ProfitM6, ProfitM7, ProfitM8, ProfitM9, ProfitM10, ProfitM11, ProfitM12 )
      my_result2 <- rbind(my_result2, data.frame(price = cost_to_driver, total_profit = YearlyProfit))
  }
}
```

Final Result

```
library(dplyr)

final_result <- my_result2 %>% group_by(price) %>% summarise(
  avg_profit = mean(total_profit)
) %>% arrange(desc(avg_profit))
plot(final_result$price, final_result$avg_profit, xlab = "Price Paying to Driver", ylab = "Average Profi
```




```
kable(head(final_result, 20))
```

price	avg_profit
25.75	51081.66
25.85	51069.70
25.65	50981.08
25.70	50949.59
25.60	50936.69
25.80	50906.78
25.45	50903.12
25.55	50886.07
25.95	50869.48
26.00	50867.80
26.05	50852.74
25.90	50832.79
25.35	50789.39
26.10	50788.19
25.50	50773.28
25.40	50761.29
25.25	50718.48
25.30	50703.18
26.20	50678.80
26.15	50635.03

Running 50 simulations it looks like the top if the best price, but high $25 looks to be best. In this case 25.75