Assignment 2

Richard Choi

20 August 2021

Using the data from week 2 of January 2016, construct a model that predicts the amount of a tip. Evaluate the mean squared error of this model on the data from week 4 of January 2016. Write a report that describes how you constructed the model and how accurate it is.

Submit the rmd and the knitted pdf or html.

Notes:

1. The data sets are fairly large. Each one has a couple of million records. You should still be able to run `regsubsets` and `lm` fairly quickly.
2. As the data dictionary indicates, tip information is not available for all trips
3. These data have not been cleaned; they are as they came from the data provider.
4. You will want to recode variables such as pickup and dropoff time and location into categories: they will not have linear relationships with tip amount. Some graphical exploration is likely to be helpful in addition to thinking about the problem. If you have problems drawing graphs because of the size of the data, taking a random subsample of, say, 10% of it can be useful.
5. The `total_amount` variable is the total amount paid. It includes the tip, and so can't be used to predict the tip.

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.1.3

## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.8
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1

## Warning: package 'ggplot2' was built under R version 4.1.3

## Warning: package 'tibble' was built under R version 4.1.3

## Warning: package 'tidyr' was built under R version 4.1.3

## Warning: package 'readr' was built under R version 4.1.3
```

```
## Warning: package 'purrr' was built under R version 4.1.3

## Warning: package 'dplyr' was built under R version 4.1.3

## Warning: package 'stringr' was built under R version 4.1.3

## Warning: package 'forcats' was built under R version 4.1.3

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(lubridate)
```

```
## Warning: package 'lubridate' was built under R version 4.1.3

##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(dplyr)
week2.df = read_csv("week2.csv")
```

```
## Rows: 2651287 Columns: 19

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr   (1): store_and_fwd_flag
## dbl  (16): VendorID, passenger_count, trip_distance, pickup_longitude, picku...
## dttm  (2): tpep_pickup_datetime, tpep_dropoff_datetime
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
week4.df = read_csv("week4.csv")
```

```
## Rows: 2010309 Columns: 19
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr   (1): store_and_fwd_flag
## dbl  (16): VendorID, passenger_count, trip_distance, pickup_longitude, picku...
## dttm  (2): tpep_pickup_datetime, tpep_dropoff_datetime
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
# Inspecting data

summary(week2.df)
```

```
##     VendorID     tpep_pickup_datetime          tpep_dropoff_datetime
##  Min.   :1.000   Min.   :2016-01-08 00:00:00   Min.   :2016-01-08 00:01:23
##  1st Qu.:1.000   1st Qu.:2016-01-09 18:02:05   1st Qu.:2016-01-09 18:16:14
##  Median :2.000   Median :2016-01-11 15:19:10   Median :2016-01-11 15:33:52
##  Mean   :1.532   Mean   :2016-01-11 14:11:04   Mean   :2016-01-11 14:25:25
##  3rd Qu.:2.000   3rd Qu.:2016-01-13 11:35:15   3rd Qu.:2016-01-13 11:50:25
##  Max.   :2.000   Max.   :2016-01-14 23:59:59   Max.   :2016-01-15 23:39:18
##  passenger_count trip_distance     pickup_longitude  pickup_latitude
##  Min.   :0.000   Min.   :  0.000   Min.   :-121.93   Min.   : 0.00
##  1st Qu.:1.000   1st Qu.:  1.000   1st Qu.: -73.99   1st Qu.:40.74
##  Median :1.000   Median :  1.630   Median : -73.98   Median :40.75
##  Mean   :1.659   Mean   :  2.809   Mean   : -72.84   Mean   :40.13
##  3rd Qu.:2.000   3rd Qu.:  3.000   3rd Qu.: -73.97   3rd Qu.:40.77
##  Max.   :9.000   Max.   :502.200   Max.   :   0.00   Max.   :46.31
##    RatecodeID     store_and_fwd_flag dropoff_longitude dropoff_latitude
##  Min.   : 1.000   Length:2651287     Min.   :-121.93   Min.   : 0.00
##  1st Qu.: 1.000   Class :character   1st Qu.: -73.99   1st Qu.:40.74
##  Median : 1.000   Mode  :character   Median : -73.98   Median :40.75
##  Mean   : 1.036                      Mean   : -72.91   Mean   :40.17
##  3rd Qu.: 1.000                      3rd Qu.: -73.96   3rd Qu.:40.77
##  Max.   :99.000                      Max.   :   0.00   Max.   :52.75
##   payment_type    fare_amount          extra              mta_tax
##  Min.   :1.000   Min.   :-957.60   Min.   :-35.6400   Min.   :-0.5000
##  1st Qu.:1.000   1st Qu.:   6.50   1st Qu.:  0.0000   1st Qu.: 0.5000
##  Median :1.000   Median :   9.00   Median :  0.0000   Median : 0.5000
##  Mean   :1.337   Mean   :  12.06   Mean   :  0.3235   Mean   : 0.4978
##  3rd Qu.:2.000   3rd Qu.:  13.50   3rd Qu.:  0.5000   3rd Qu.: 0.5000
##  Max.   :4.000   Max.   :3039.00   Max.   :  4.1000   Max.   :36.4400
##    tip_amount         tolls_amount      improvement_surcharge  total_amount
##  Min.   :-220.800   Min.   :-12.5000   Min.   :-0.3000       Min.   :-958.40
##  1st Qu.:   0.000   1st Qu.:  0.0000   1st Qu.: 0.3000       1st Qu.:   8.30
##  Median :   1.320   Median :  0.0000   Median : 0.3000       Median :  11.30
##  Mean   :   1.728   Mean   :  0.2746   Mean   : 0.2998       Mean   :  15.18
##  3rd Qu.:   2.260   3rd Qu.:  0.0000   3rd Qu.: 0.3000       3rd Qu.:  16.56
##  Max.   : 900.000   Max.   :811.0000   Max.   : 0.3000       Max.   :3045.34
```

```r
sum(week2.df$fare_amount < 0)
```

```
## [1] 930
```

```r
unique(week2.df$RatecodeID)
```

```
## [1]  1  5  2  4  3 99  6
```

```r
unique(week2.df$payment_type)
```

```
## [1] 1 2 4 3
```

```
unique(week2.df$store_and_fwd_flag)
```

```
## [1] "N" "Y"
```

```
sum(week2.df$extra < 0)
```

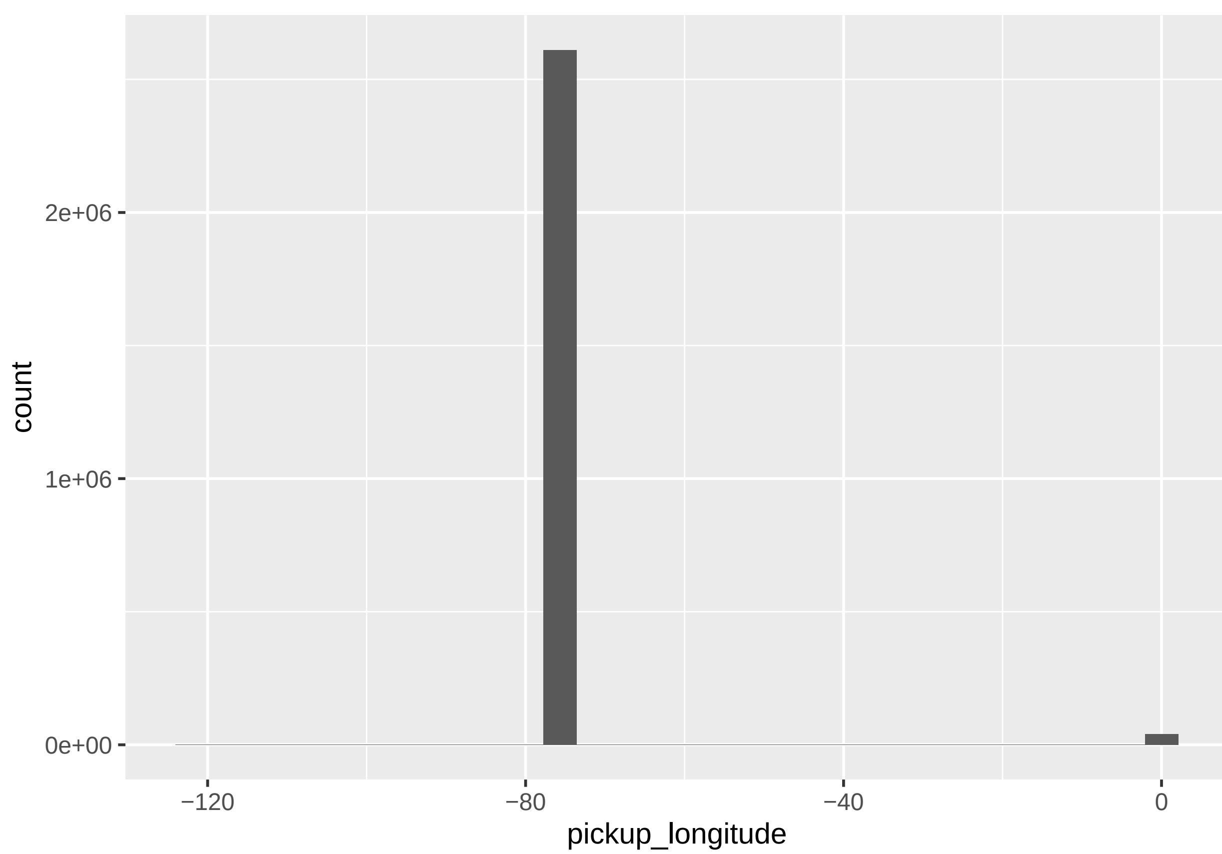
```
## [1] 448
```

```
any(is.na(week2.df))
```

```
## [1] FALSE
```

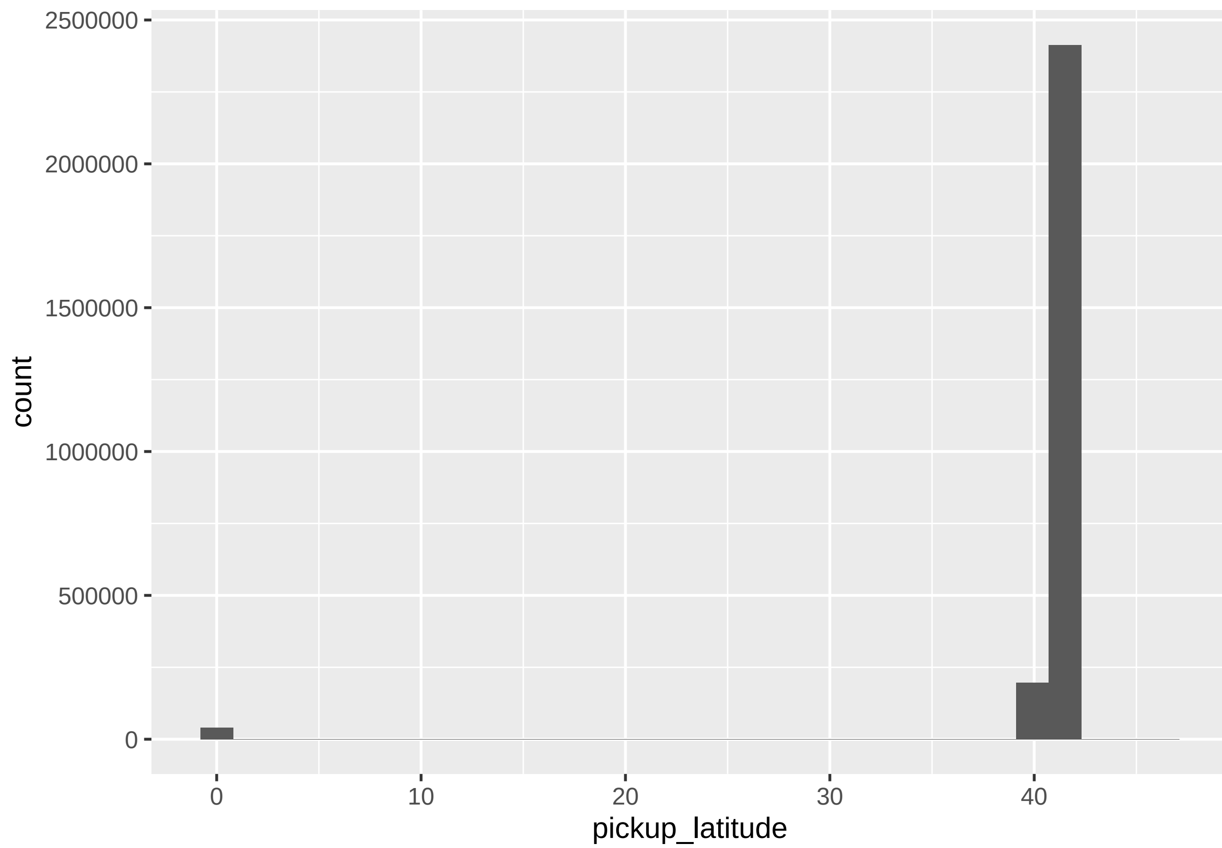
```
week2.df %>%
  ggplot(aes(x=pickup_longitude)) + geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



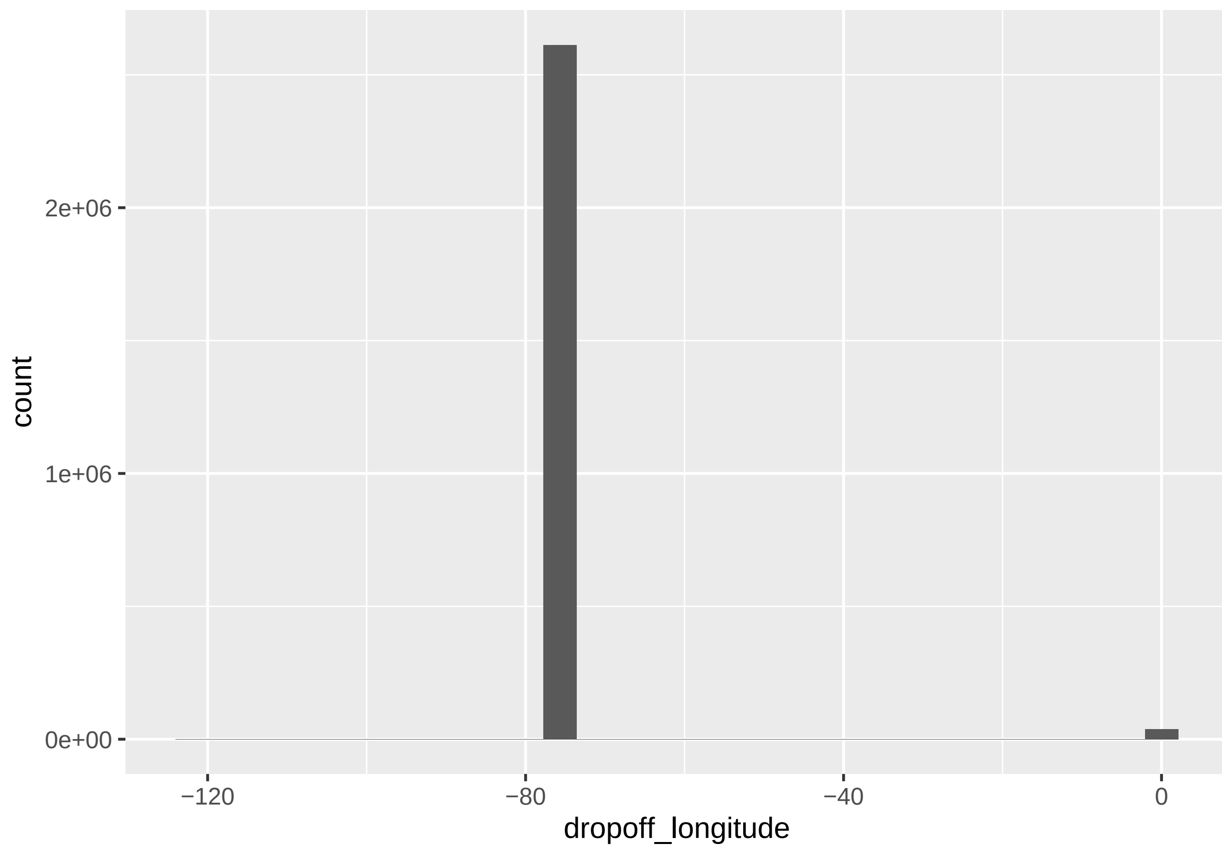

```
week2.df %>%
  ggplot(aes(x=pickup_latitude)) + geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



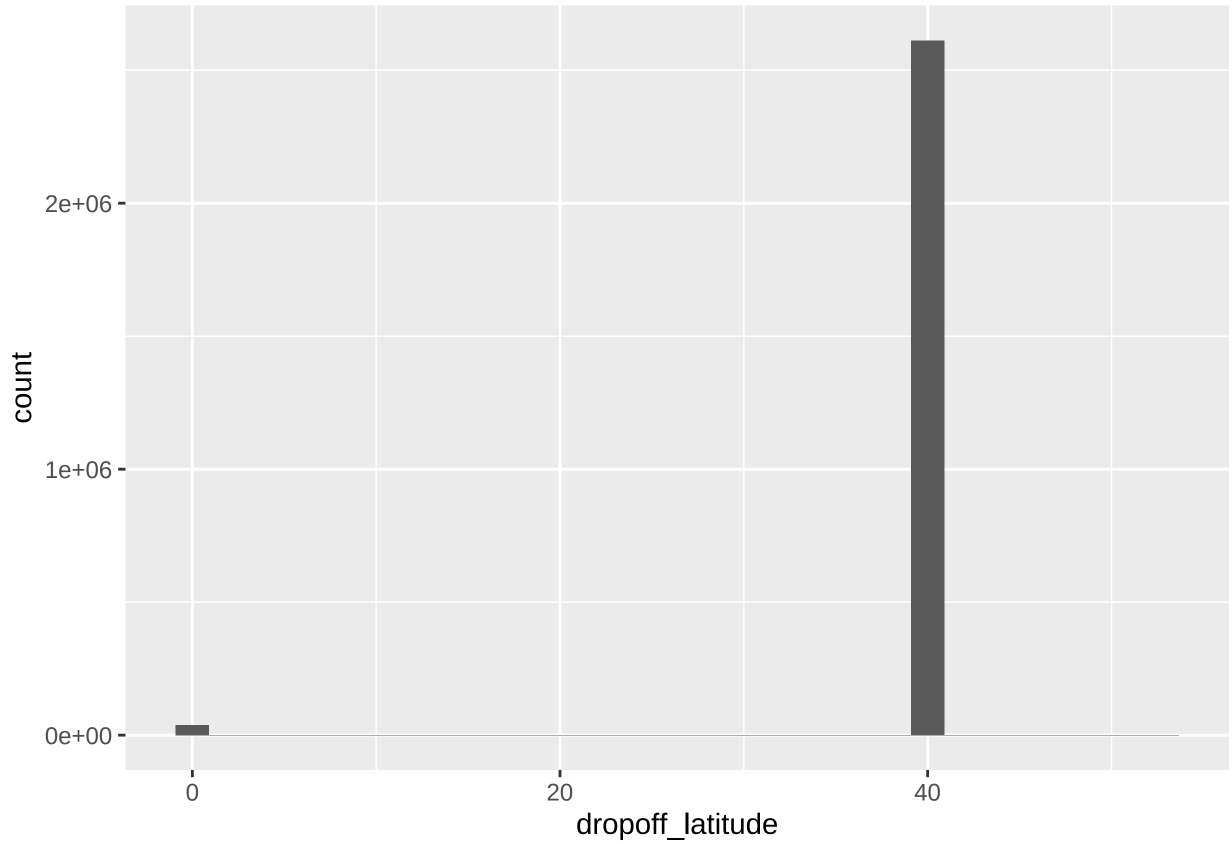

```
week2.df %>%
  ggplot(aes(x=dropoff_longitude)) + geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
week2.df %>%
  ggplot(aes(x=dropoff_latitude)) + geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



We notice our data needs some cleaning as some values shouldn't be negative and variables like ratecodeID are categorised by 1 to 6 yet it's out of range.

```
# Cleaning data
week2.tidy = week2.df %>%
  filter(trip_distance >0) %>%
  filter(RatecodeID <= 6) %>%
  filter(fare_amount >0) %>%
  filter(extra >=0) %>%
  filter(mta_tax >=0) %>%
  filter(tip_amount>=0) %>%
  filter(improvement_surcharge>=0) %>%
  filter(total_amount > 0) %>%
  filter(passenger_count >0 & passenger_count <=4) %>%
  filter(between(pickup_longitude, -75, -70)) %>%
  filter(between(pickup_latitude, 40, 42)) %>%
  mutate(time = case_when(hour(tpep_pickup_datetime) >= 0 & hour(tpep_pickup_datetime) <= 5 ~ "Early Mo
                          hour(tpep_pickup_datetime) >= 6 & hour(tpep_pickup_datetime) <= 11 ~ "Morning
                          hour(tpep_pickup_datetime) >= 12 & hour(tpep_pickup_datetime) <= 17 ~ "Aftern
                          hour(tpep_pickup_datetime) >= 18 & hour(tpep_pickup_datetime) <= 23 ~ "Evenin
  mutate(pickup_location = case_when(between(pickup_latitude, 40.7, 40.88) & between(pickup_longitude, 
                              between(pickup_latitude, 40.57, 40.7378) & between(pickup_longitude, -74.
                              between(pickup_latitude, 40.63, 40.739) & between(pickup_longitude, -73.9
                              TRUE~"Other")) %>%
  mutate(dropoff_location = case_when(between(dropoff_latitude, 40.7, 40.88) & between(dropoff_longitud
                              between(dropoff_latitude, 40.57, 40.7378) & between(dropoff_longitude, -7
                              between(dropoff_latitude, 40.63, 40.739) & between(dropoff_longitude, -73
```

```r
                              TRUE~"Other")) %>%
  mutate(pickup_dow = factor(weekdays(tpep_pickup_datetime), levels=c("Monday", "Tuesday", "Wednesday",
  mutate(payment_type = fct_recode(factor(payment_type), "Credit card" = "1", "cash" = "2", "No charge"
  mutate(RatecodeID = fct_recode(factor(RatecodeID), "Standard rate" = "1", "JFK" = "2", "Newark" = "3"
  mutate(pickupday_type = case_when(pickup_dow %in% c("Monday", "Tuesday", "Wednesday", "Thursday", "Fri
                             pickup_dow %in% c("Saturday", "Sunday") ~ "Weekend")) %>%
  select(tip_amount, passenger_count, trip_distance, RatecodeID, payment_type, fare_amount, extra, toll
```

```
## Warning: Unknown levels in 'f': 5, 6
```

I have decided to categorise time to (working day & weekend), (evening & early morning), factored payment type and ratecodeID to categories. Using pickup latitude and pickup longitude, the location was categorised to main boroughs in the New York City which are Manhattan, Queens, Brooklyn, and Other. Lot of tax drives should be airport trips so I have used the main boroughs. I have also dropped variables like VendorID, mta_tax, improvement surcharge, store and fwd flag as they had no relation to tips or other variables have covered their presence. For example, tip should not vary depending on the type of TPEP provider or trip distance or fare amount should account for MTA tax so MTA tax would have a small impact on tips with other variables prescenece. On the other hand, tpep pickup date time, tpep drop off date time were used to create categories of time like earlymorning and evening. The pickup longitude and latitude, were used to estimate the location of the main boroughs in the New York city.

```r
# data exploration

week2.tidy %>%
  group_by(pickup_location) %>%
  summarise(n())
```

```
## # A tibble: 4 x 2
##   pickup_location   `n()`
##   <chr>             <int>
## 1 Brooklyn         230040
## 2 Manhattan       1929872
## 3 Other            155065
## 4 Queen             49914
```

```r
week2.tidy %>%
  group_by(RatecodeID) %>%
  summarise(n())
```

```
## # A tibble: 6 x 2
##   RatecodeID              `n()`
##   <fct>                   <int>
## 1 Standard rate         2314904
## 2 JFK                     42873
## 3 Newark                   3021
## 4 Nassau or Westchester     925
## 5 Negotiated fare          3159
## 6 Group ride                  9
```

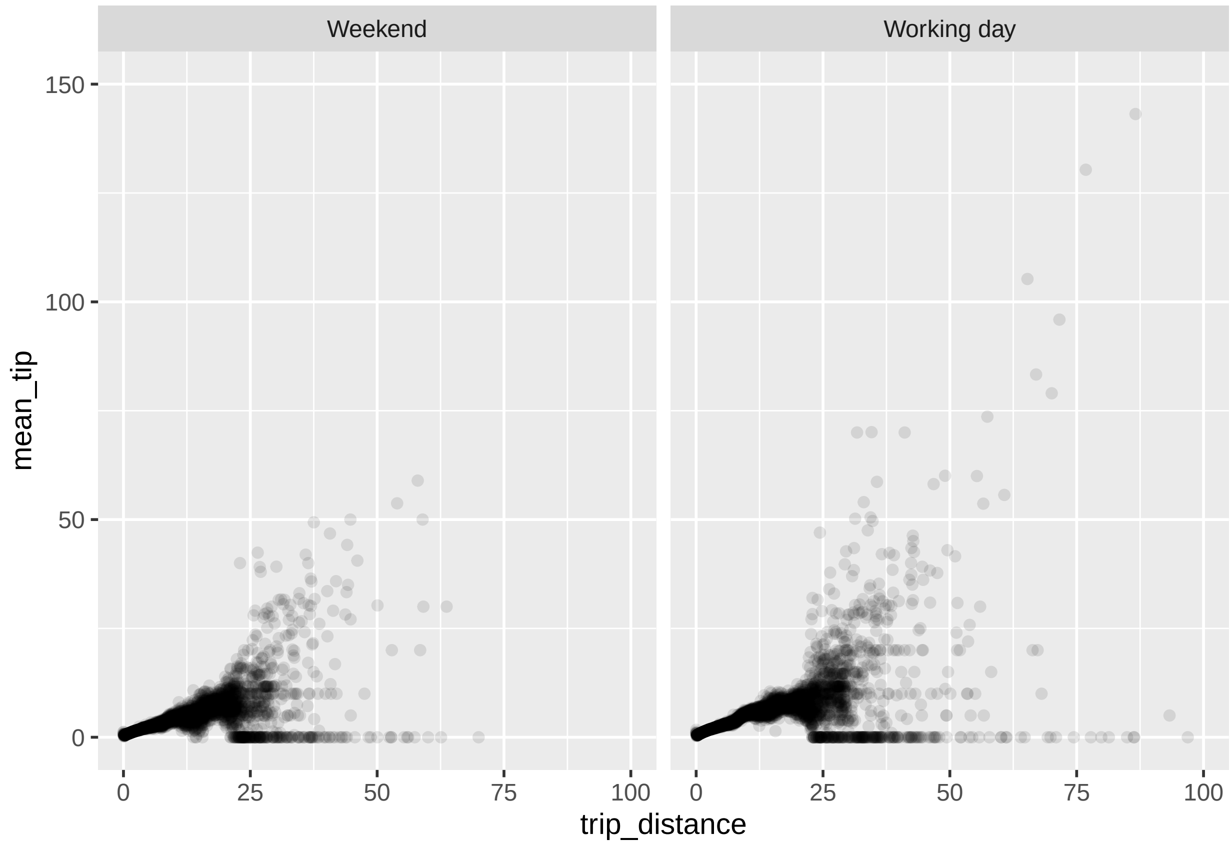
```r
week2.tidy %>%
  group_by(passenger_count) %>%
  summarise(n())
```

```
## # A tibble: 4 x 2
##   passenger_count   `n()`
##             <dbl>   <int>
## 1               1 1853649
## 2               2  363766
## 3               3  100330
## 4               4   47146
```

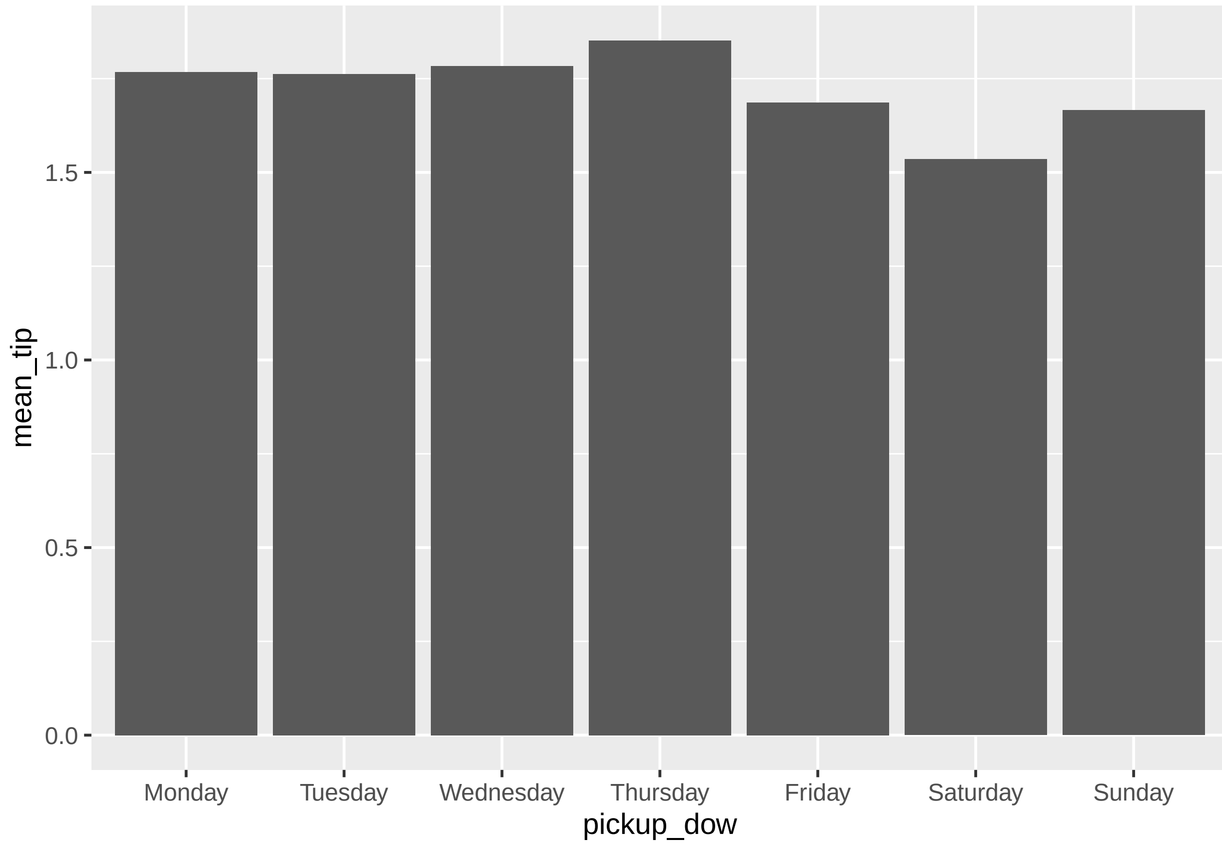
```
week2.tidy %>%
  group_by(trip_distance, pickupday_type) %>%
  summarise(mean_tip = mean(tip_amount)) %>%
    ggplot(aes(x=trip_distance, y=mean_tip)) + geom_point(alpha=0.1) + xlim(0,100) + ylim(0, 150) + fac
```

```
## `summarise()` has grouped output by 'trip_distance'. You can override using the
## `.groups` argument.
```

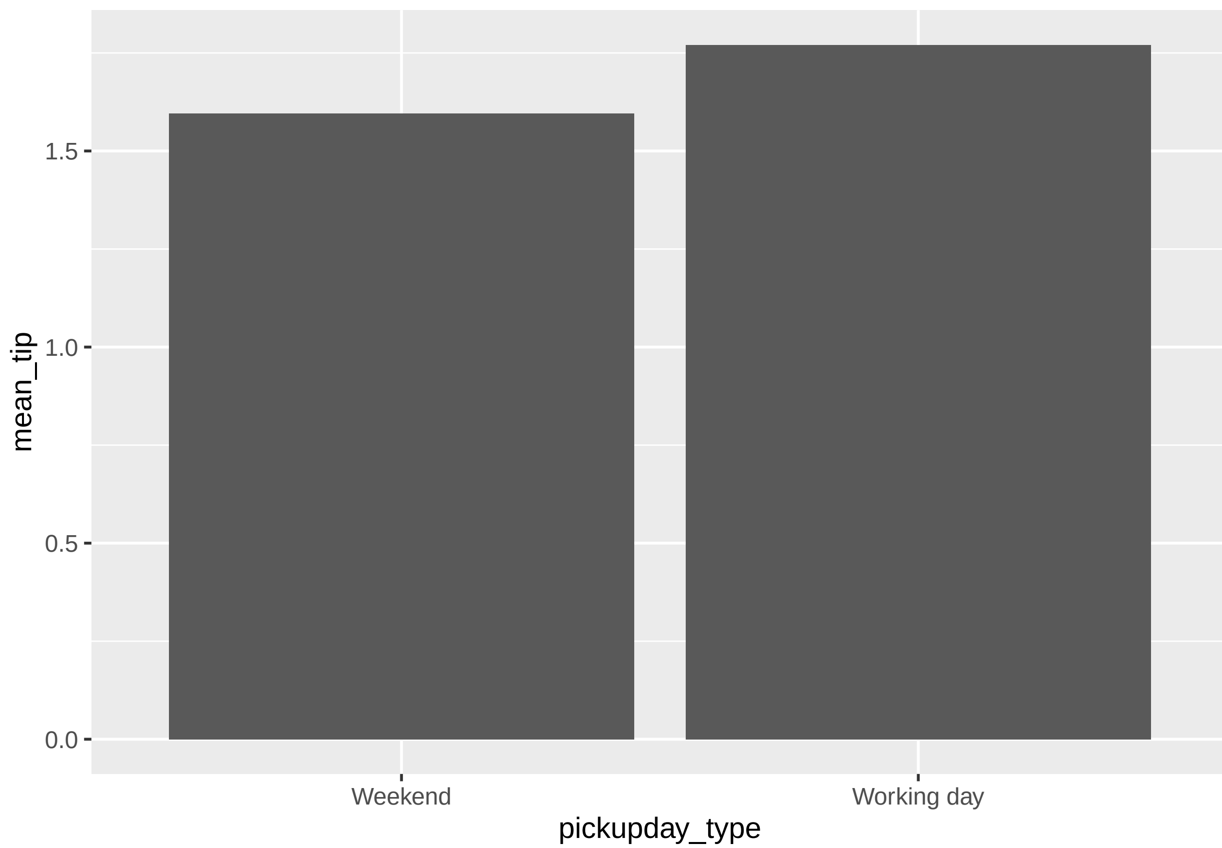
```
## Warning: Removed 17 rows containing missing values (geom_point).
```



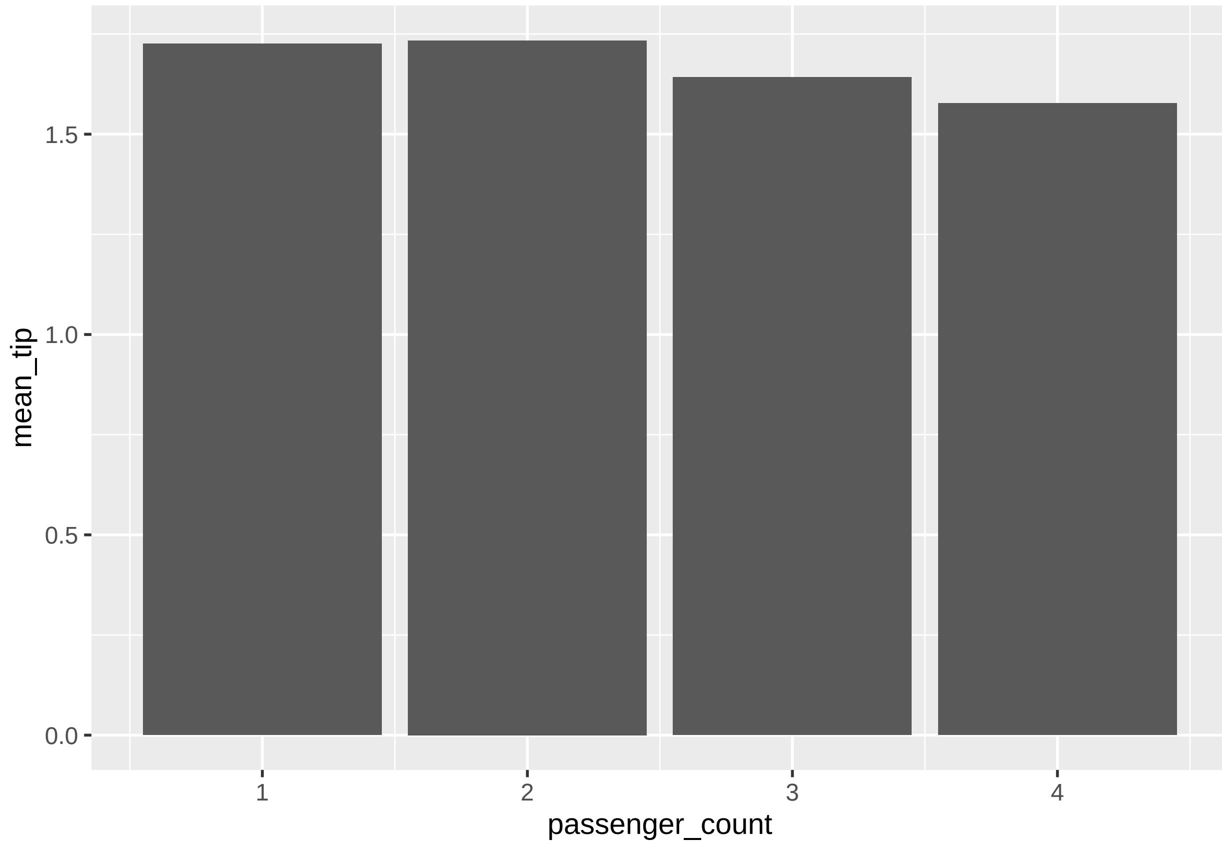

```
week2.tidy %>%
  group_by(pickup_dow) %>%
  summarise(mean_tip = mean(tip_amount)) %>%
  ggplot(aes(x=pickup_dow, y=mean_tip)) + geom_bar(stat='identity')
```




```
week2.tidy %>%
  group_by(pickupday_type) %>%
  summarise(mean_tip = mean(tip_amount)) %>%
  ggplot(aes(x=pickupday_type, y=mean_tip)) + geom_bar(stat='identity')
```

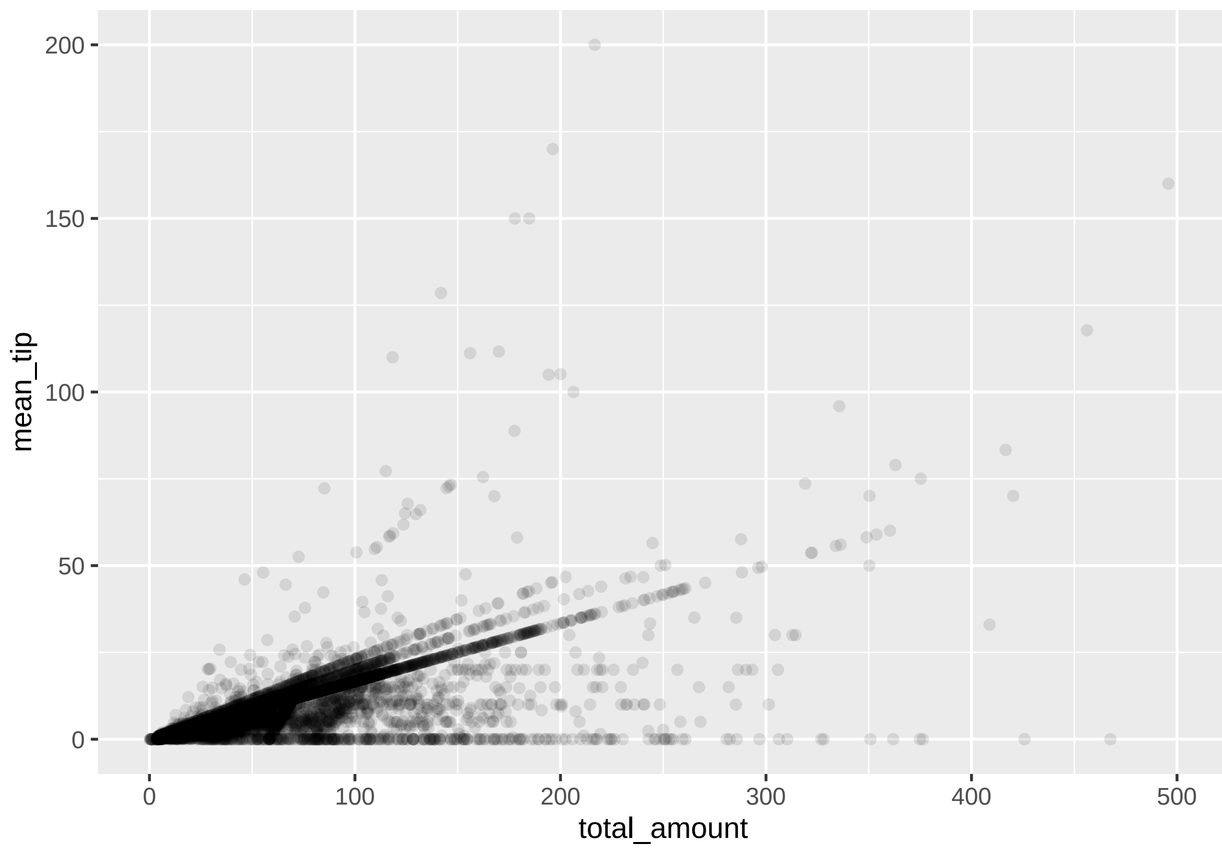



```
week2.tidy %>%
  group_by(passenger_count) %>%
  summarise(mean_tip = mean(tip_amount)) %>%
  ggplot(aes(x=passenger_count, y=mean_tip)) + geom_bar(stat='identity')
```

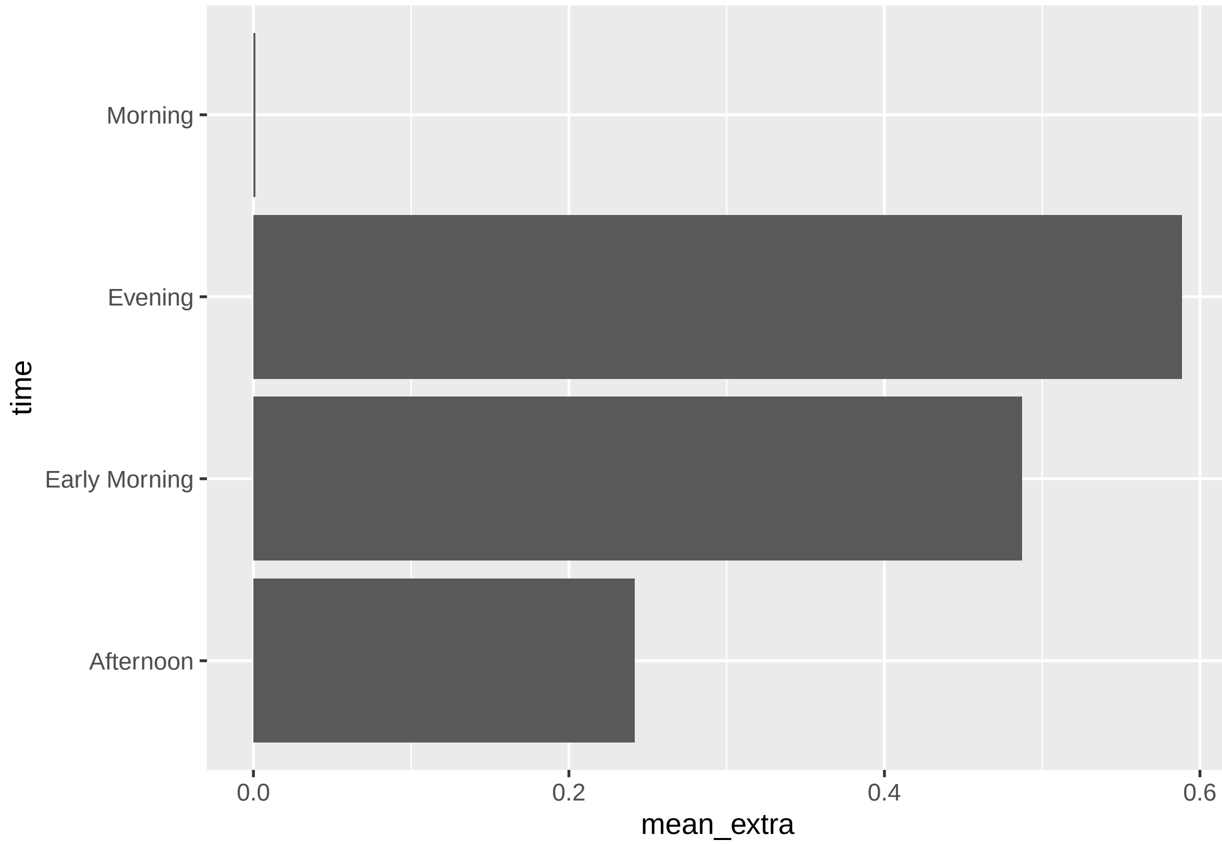



```
week2.tidy %>%
  group_by(total_amount) %>%
  summarise(mean_tip = mean(tip_amount)) %>%
  ggplot(aes(x=total_amount, y=mean_tip)) + geom_point(alpha=0.1) + xlim(0,500) + ylim(0, 200)
```

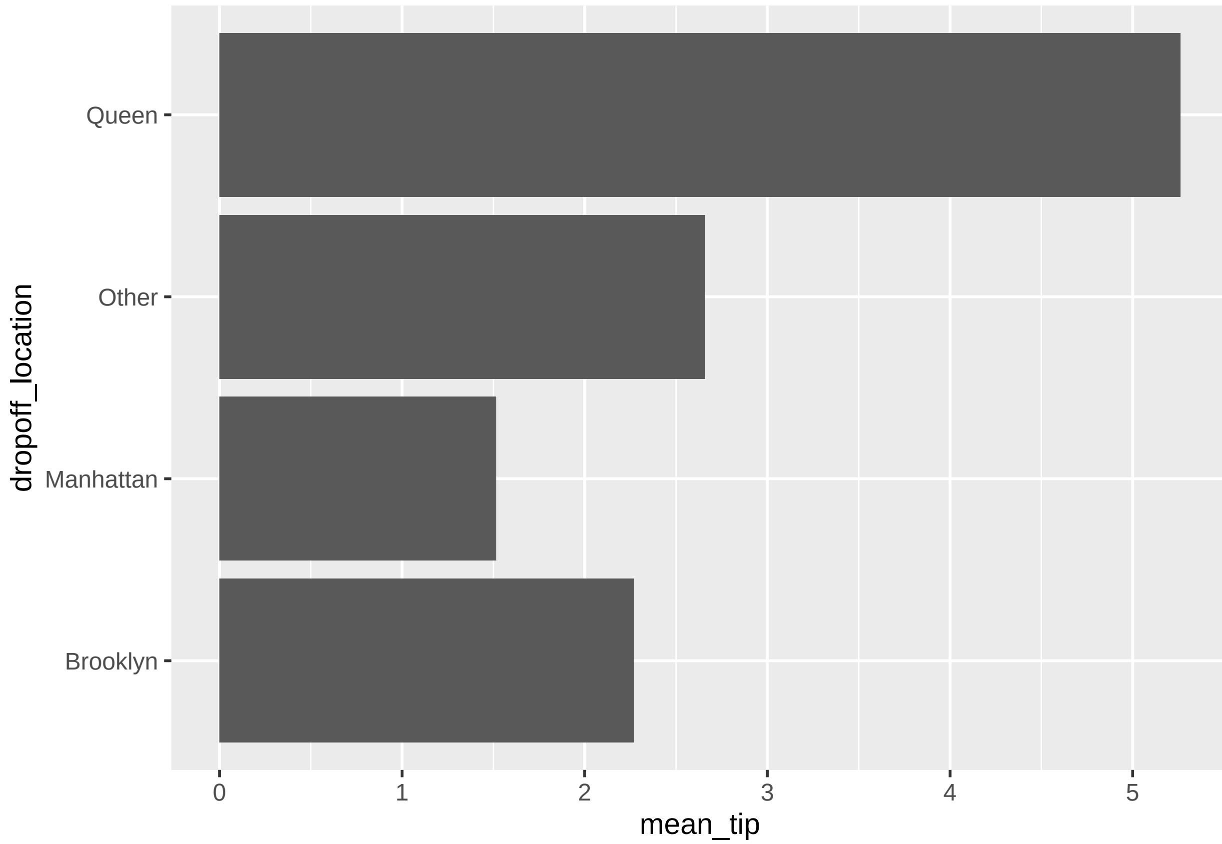
```
## Warning: Removed 14 rows containing missing values (geom_point).
```



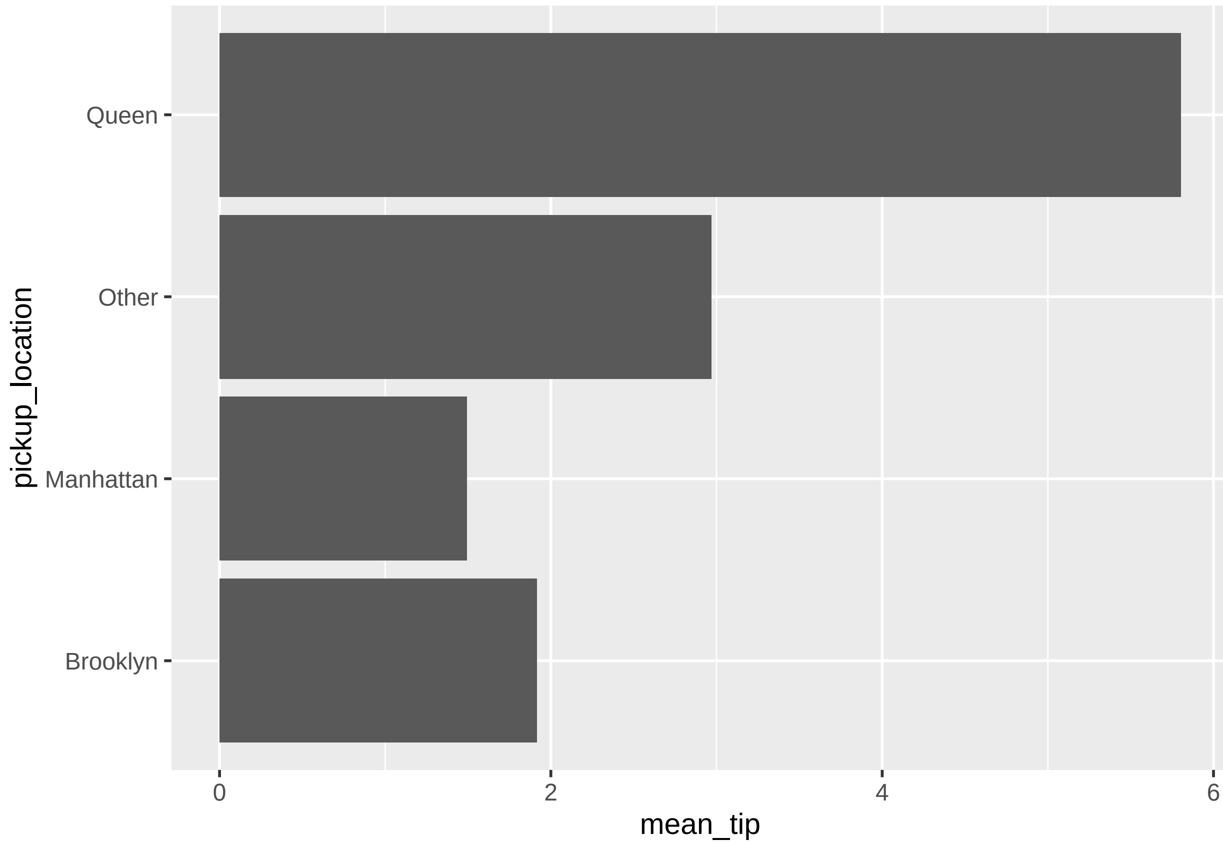

```
week2.tidy %>%
  group_by(time) %>%
  summarise(mean_extra = mean(extra)) %>%
  ggplot(aes(x=mean_extra, y=time)) + geom_bar(stat='identity')
```




```
week2.tidy %>%
  group_by(dropoff_location) %>%
  summarise(mean_tip = mean(tip_amount)) %>%
  ggplot(aes(x=mean_tip, y=dropoff_location)) + geom_bar(stat='identity')
```

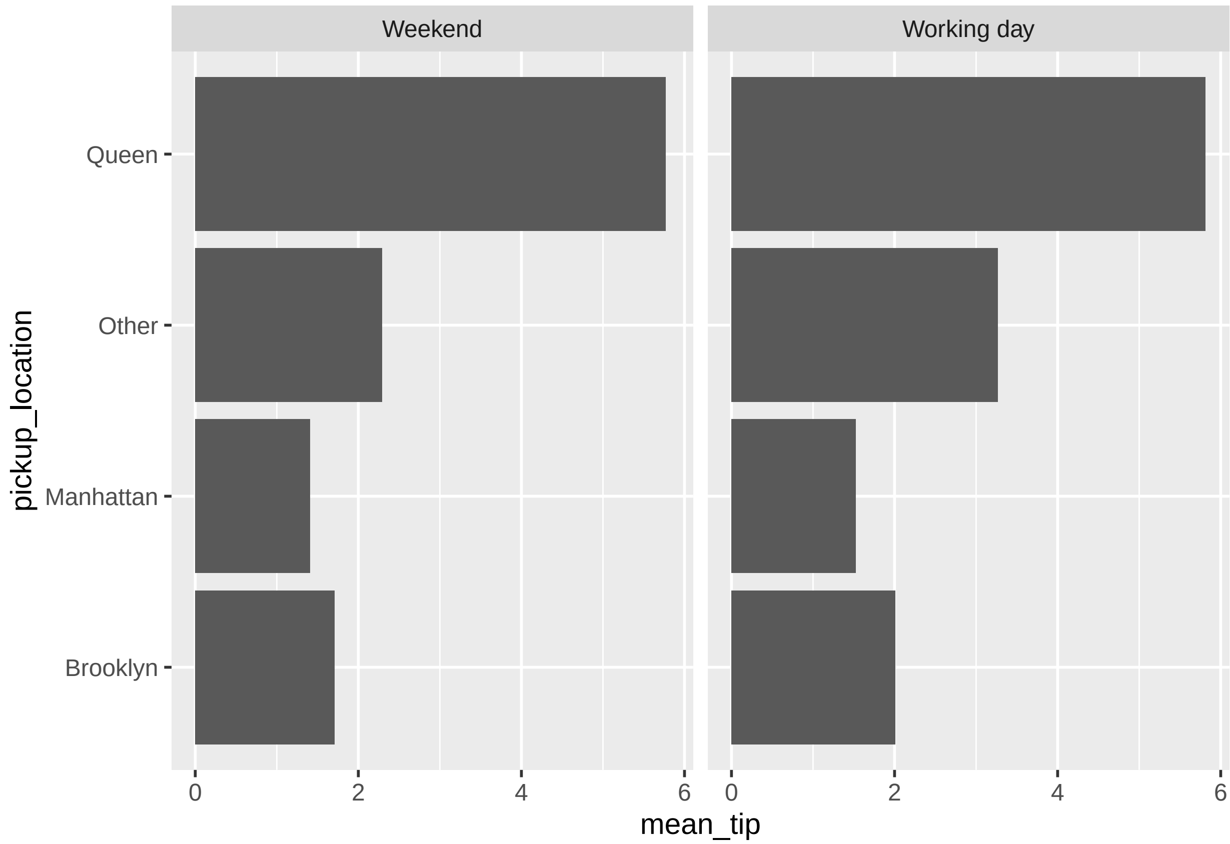



```
week2.tidy %>%
  group_by(pickup_location) %>%
  summarise(mean_tip = mean(tip_amount)) %>%
  ggplot(aes(x=mean_tip, y=pickup_location)) + geom_bar(stat='identity')
```



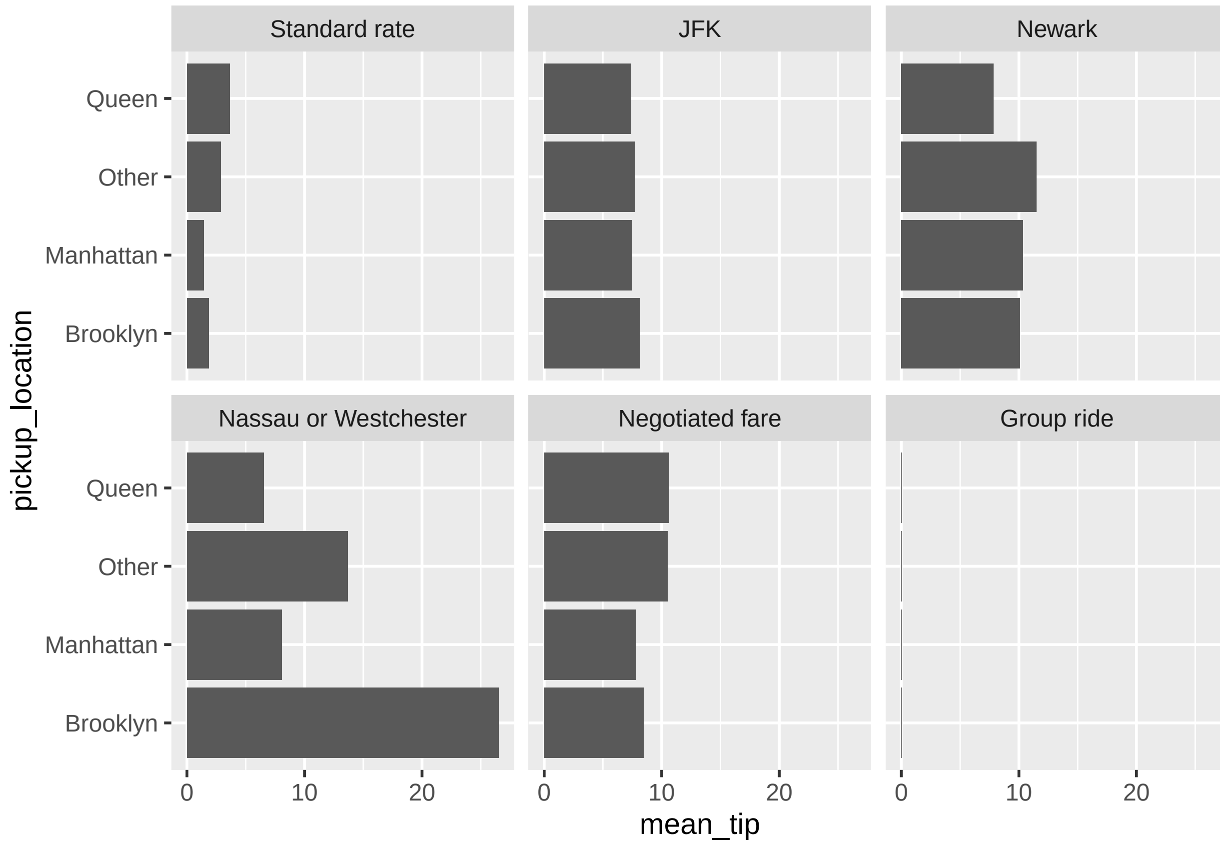

```
week2.tidy %>%
  group_by(pickup_location, pickupday_type) %>%
  summarise(mean_tip = mean(tip_amount)) %>%
  ggplot(aes(x=mean_tip, y=pickup_location)) + geom_bar(stat='identity') + facet_wrap(~pickupday_type)
```

```
## `summarise()` has grouped output by 'pickup_location'. You can override using
## the `.groups` argument.
```



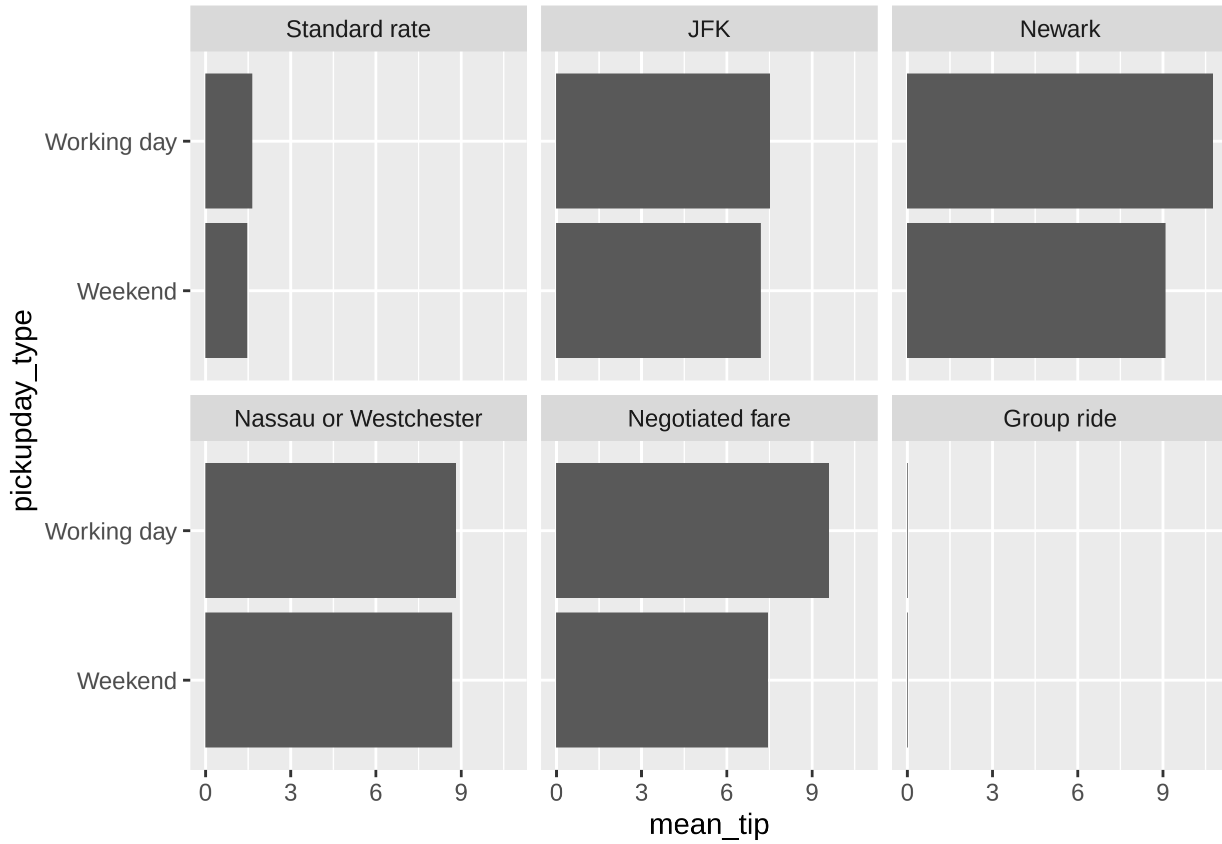

```
week2.tidy %>%
  group_by(pickup_location, RatecodeID) %>%
  summarise(mean_tip = mean(tip_amount)) %>%
  ggplot(aes(x=mean_tip, y=pickup_location)) + geom_bar(stat='identity') + facet_wrap(~RatecodeID)
```

```
## `summarise()` has grouped output by 'pickup_location'. You can override using
## the `.groups` argument.
```



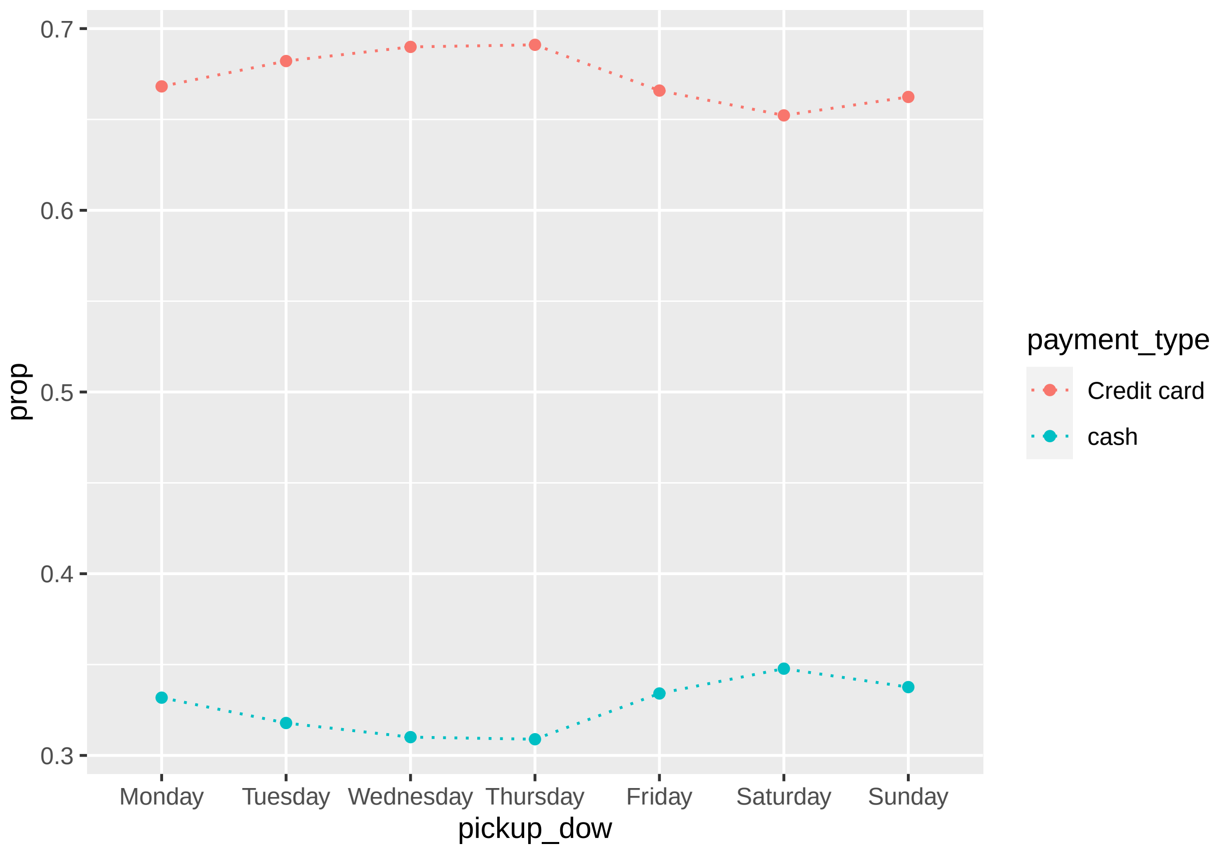

```
week2.tidy %>%
  group_by(pickupday_type, RatecodeID) %>%
  summarise(mean_tip = mean(tip_amount)) %>%
  ggplot(aes(x=mean_tip, y=pickupday_type)) + geom_bar(stat='identity') + facet_wrap(~RatecodeID)
```

```
## `summarise()` has grouped output by 'pickupday_type'. You can override using
## the `.groups` argument.
```

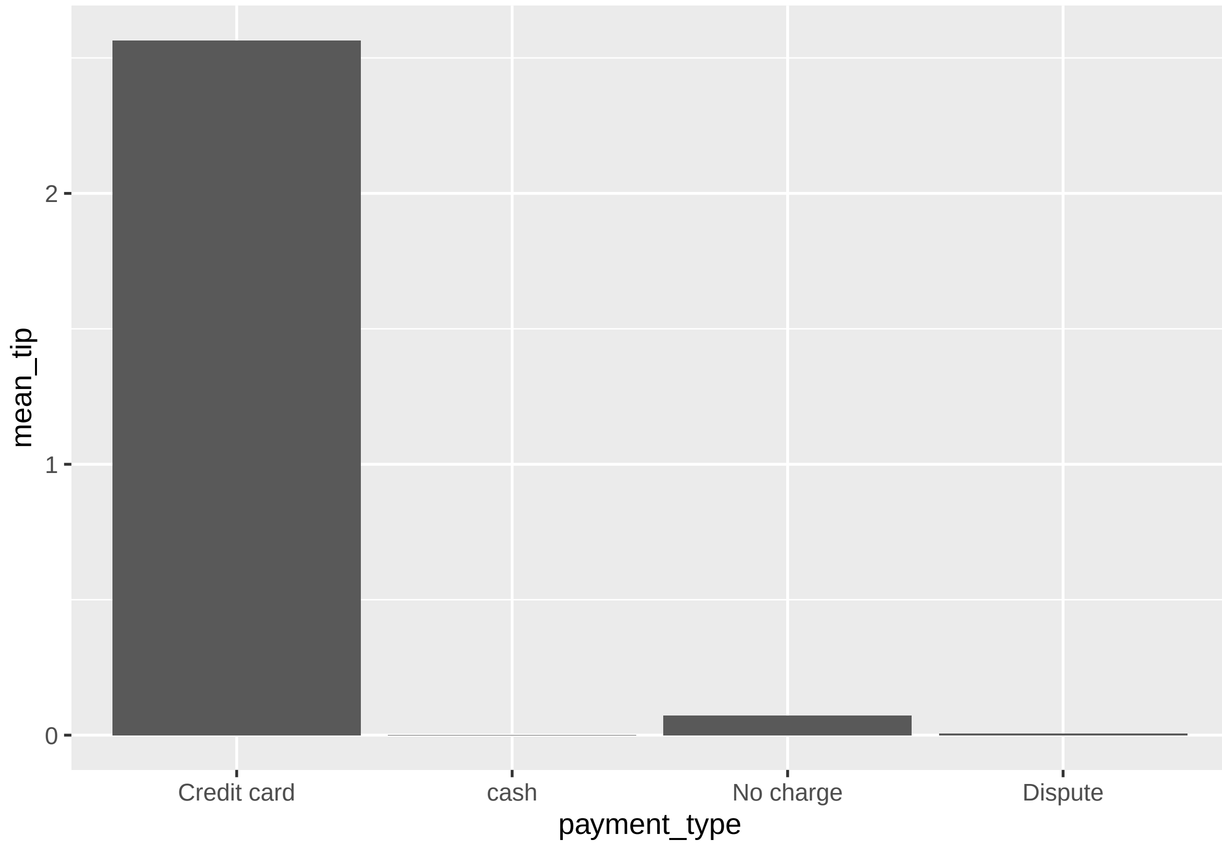



```
week2.tidy %>%
  filter(payment_type %in% c('Credit card','cash')) %>%
  group_by(pickup_dow,payment_type) %>%
  summarise(n = n()) %>%
  mutate(sum = sum(n), prop = n/sum) %>%
  ggplot(aes(x=pickup_dow, y=prop,color=payment_type, group=payment_type)) + geom_point() + geom_line(li
```

```
## `summarise()` has grouped output by 'pickup_dow'. You can override using the
## `.groups` argument.
```

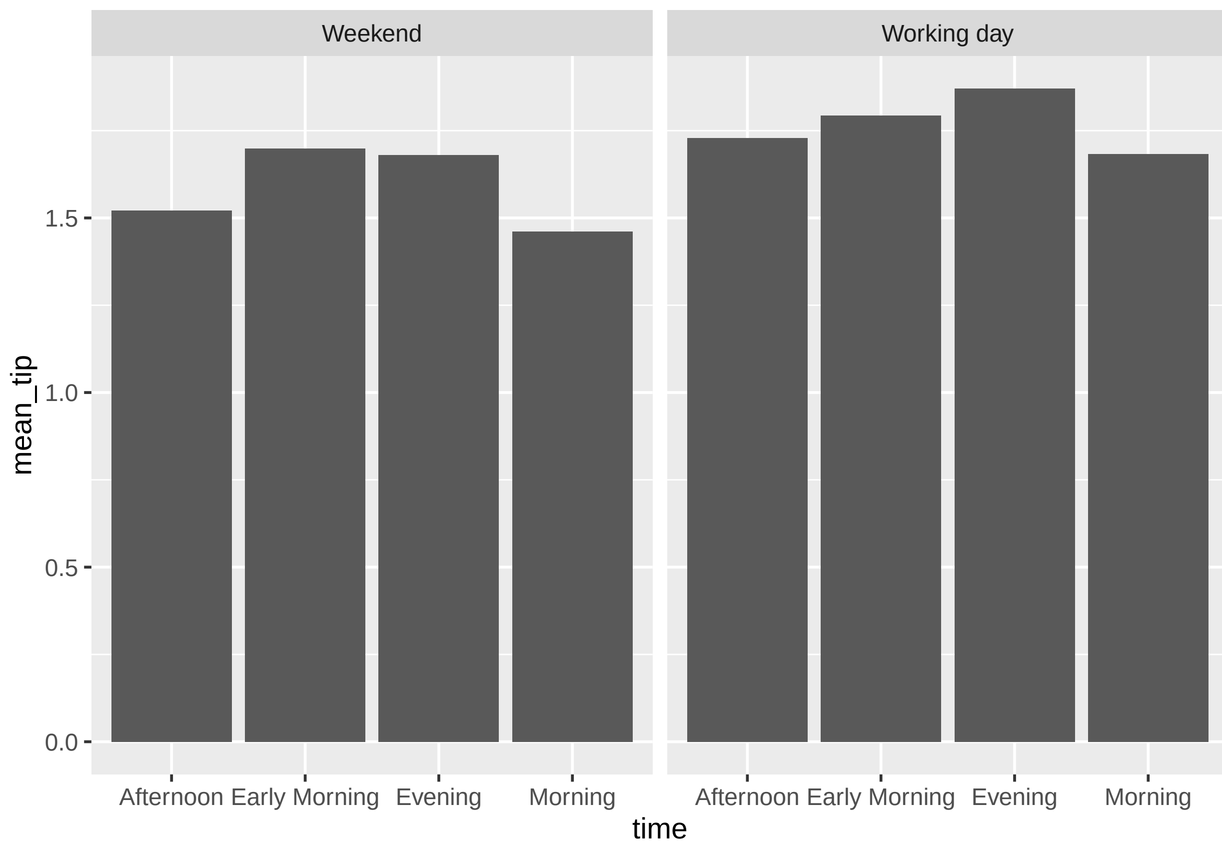



```
week2.tidy %>%
  group_by(payment_type) %>%
  summarise(mean_tip = mean(tip_amount)) %>%
    ggplot(aes(x=payment_type, y=mean_tip)) + geom_bar(stat='identity')
```




```
week2.tidy %>%
  group_by(time, pickupday_type) %>%
  summarise(mean_tip=mean(tip_amount)) %>%
  ggplot(aes(x=time, y=mean_tip)) + geom_bar(stat='identity') + facet_wrap(~pickupday_type)
```

```
## `summarise()` has grouped output by 'time'. You can override using the
## `.groups` argument.
```



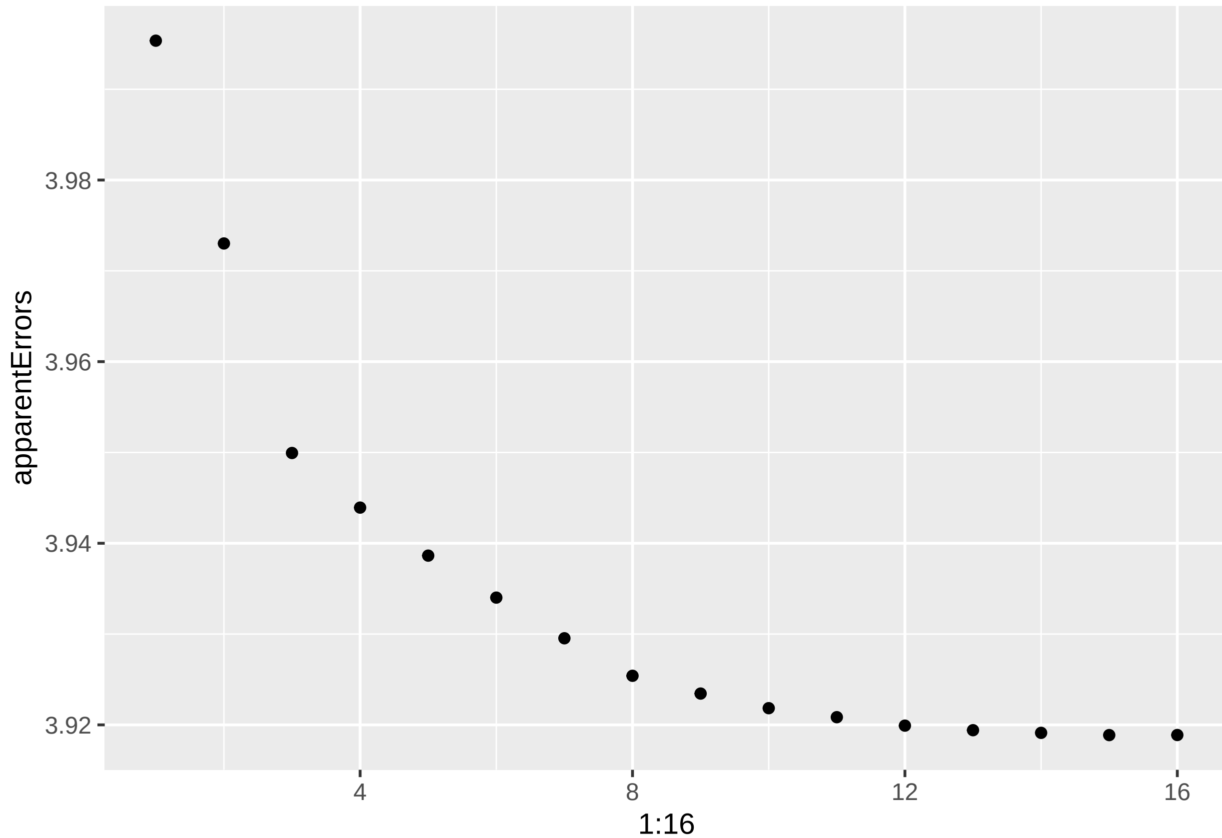

```
# remove more variables
week2.tidy = week2.tidy %>%
  select(-tolls_amount, -dropoff_location, -pickup_dow, -payment_type, -total_amount)
```

We have fitted some graphs to explore the data and we found some interesting points. We can observe that there is higher average tips in the weekdays than weekends. From the scatter plots, we can see there is an increase in scatter for the average tip as x increases. On average, there is highest mean tip in Queen and we can see that tips on different types of rates in different area don't vary as much except for Brooklyn. Although we see a high increase in tips for high amount of passengers, there isn't much high amount of passengers (7, or 8) proportion to the whole data set. We also have decided to drop some variables like extra, toll amount, total amount, an drop off location, and pickup dow because I did not see a relationship with tip or other variables can cover it and we do not want complicated model. The tips only include for credit cards so we decided to remove payment type as there will be no interaction with tips and cash payment. The total payment includes the tip it can't be used to predict tip so total payment was removed.

```
# Sampling data because it takes too much time and RAM

set.seed(369)
index = sample(1:nrow(week2.tidy), nrow(week2.tidy)*0.2)
week2.sample = week2.tidy[index,]

mf<-model.frame(tip_amount~., data=week2.sample)
X<-model.matrix(tip_amount~., mf)[,-1]

library(leaps)
```

```
## Warning: package 'leaps' was built under R version 4.1.3
```

```
subsets1.reg = regsubsets(X, week2.sample$tip_amount, nvmax = 16, method = "backward")
subsets1.summary = summary(subsets1.reg)
apparentErrors = subsets1.summary$rss / (nrow(week2.sample) - 1:16)
qplot(y = apparentErrors, x= 1:16)
```




```
allyhat<-function(xtrain, ytrain, xtest,lambdas,nvmax){
  n<-nrow(xtrain)
  yhat<-matrix(nrow=nrow(xtest),ncol=length(lambdas))
  search<-regsubsets(xtrain,ytrain, nvmax=nvmax, method="back")
  summ<-summary(search)
  for(i in 1:length(lambdas)){
    penMSE<- n*log(summ$rss)+lambdas[i]*(1:nvmax)
    best<-which.min(penMSE)  #lowest AIC
    betahat<-coef(search, best) #coefficients
    xinmodel<-cbind(1,xtest)[,summ$which[best,]] #predictors in that model
    yhat[,i]<-xinmodel%*%betahat
  }
  yhat
}

y = week2.sample$tip_amount

n<-nrow(X)
```

```r
folds<-sample(rep(1:10,length.out=n))
lambdas<-c(2,4,6,8,10,12)
fitted<-matrix(nrow=n,ncol=length(lambdas))
for(k in 1:10){
  train<- (1:n)[folds!=k]
  test<-(1:n)[folds==k]
  fitted[test,]<-allyhat(X[train,],y[train],X[test,],lambdas,nvmax = 16)
}
rbind(lambdas,colMeans((y-fitted)^2))
```

```
##             [,1]     [,2]     [,3]     [,4]      [,5]      [,6]
## lambdas 2.000000 4.000000 6.000000 8.000000 10.000000 12.000000
##         3.951907 3.951907 3.952305 3.952305  3.952305  3.952305
```

```r
colMeans((y-fitted)^2)
```

```
## [1] 3.951907 3.951907 3.952305 3.952305 3.952305 3.952305
```

Due to limited computational resources (my Rstudio could not handle it and it crashed), I have decided to sample the data and fit a linear model without any interaction. Interaction was also not fitted because it required higher computational resources and there wasn't much difference in average tips for adding interaction like different locations and time or weekends vs weekdays as shown above. Using cost-complex strategy, we have fitted various lambdas and last four lambda give us the same result. The lambda of 2 will be used as we do not want to penalise the model if it gives the same result with higher penalties. Looks like AIC is good enough for this model.

```r
search<-regsubsets(X,y, nvmax=16, method="back")
summ<-summary(search)
penMSE<- nrow(X)*log(summ$rss)+2*(1:16)
best<-which.min(penMSE) #lowest penalisedRSS
betahat<-coef(search, best) #coefficients
xinmodel<-cbind(1,X)[,summ$which[best,]]
yhat2<-xinmodel%*%betahat
betahat
```

```
##                   (Intercept)                passenger_count
##                    0.20374223                    -0.06143116
##                 trip_distance                   RatecodeIDJFK
##                    0.13370425                     0.67716427
##              RatecodeIDNewark RatecodeIDNassau or Westchester
##                    2.01230135                     0.86359566
##      RatecodeIDNegotiated fare                    fare_amount
##                   -1.25429379                     0.09693716
##                         extra              timeEarly Morning
##                    0.09697920                    -0.13040965
##                   timeEvening                    timeMorning
##                    0.11441157                     0.04599229
##       pickup_locationManhattan           pickup_locationOther
##                   -0.10520610                     0.27248767
##          pickup_locationQueen       pickupday_typeWorking day
##                   -1.43785589                     0.09983914
```

```
week4.tidy = week4.df %>%
  filter(trip_distance >0) %>%
  filter(RatecodeID <= 6) %>%
  filter(fare_amount >0) %>%
  filter(extra >=0) %>%
  filter(mta_tax >=0) %>%
  filter(tip_amount>=0) %>%
  filter(improvement_surcharge>=0) %>%
  filter(total_amount > 0) %>%
  filter(passenger_count >0 & passenger_count <=4) %>%
  filter(between(pickup_longitude, -75, -70)) %>%
  filter(between(pickup_latitude, 40, 42)) %>%
  mutate(time = case_when(hour(tpep_pickup_datetime) >= 0 & hour(tpep_pickup_datetime) <= 5 ~ "Early Mo
                         hour(tpep_pickup_datetime) >= 6 & hour(tpep_pickup_datetime) <= 11 ~ "Morning
                         hour(tpep_pickup_datetime) >= 12 & hour(tpep_pickup_datetime) <= 17 ~ "Aftern
                         hour(tpep_pickup_datetime) >= 18 & hour(tpep_pickup_datetime) <= 23 ~ "Evenin
  mutate(pickup_location = case_when(between(pickup_latitude, 40.7, 40.88) & between(pickup_longitude, 
                             between(pickup_latitude, 40.57, 40.7378) & between(pickup_longitude, -74.
                             between(pickup_latitude, 40.63, 40.739) & between(pickup_longitude, -73.9
                             TRUE~"Other")) %>%
  mutate(dropoff_location = case_when(between(dropoff_latitude, 40.7, 40.88) & between(dropoff_longitud
                             between(dropoff_latitude, 40.57, 40.7378) & between(dropoff_longitude, -7
                             between(dropoff_latitude, 40.63, 40.739) & between(dropoff_longitude, -73
                             TRUE~"Other")) %>%
  mutate(pickup_dow = factor(weekdays(tpep_pickup_datetime), levels=c("Monday", "Tuesday", "Wednesday",
  mutate(payment_type = fct_recode(factor(payment_type), "Credit card" = "1", "cash" = "2", "No charge"
  mutate(RatecodeID = fct_recode(factor(RatecodeID), "Standard rate" = "1", "JFK" = "2", "Newark" = "3"
  mutate(pickupday_type = case_when(pickup_dow %in% c("Monday", "Tuesday", "Wednesday", "Thursday", "Fr
                            pickup_dow %in% c("Saturday", "Sunday") ~ "Weekend")) %>%
  select(tip_amount, passenger_count, trip_distance, RatecodeID, payment_type, fare_amount, extra, toll
```

```
## Warning: Unknown levels in `f`: 5, 6
```

```
week4.tidy = week4.tidy %>%
  select(-tolls_amount, -dropoff_location, -pickup_dow, -payment_type, -total_amount)
```

Week 4 data have been cleaned the same way as week 2 to remove any meaningless columns and outliers that do not make sense. The variables need to be consistent with week 2 training data to test the model with week 4 test data.

```
set.seed(369)
index2 = sample(1:nrow(week4.tidy), nrow(week4.tidy)*0.2)
week4.sample = week4.tidy[index2,]

mf<-model.frame(tip_amount~., data=week4.sample)
X2<-model.matrix(tip_amount~., mf)[,summ$which[best,]]

fitted = X2 %*% betahat

MSPEsample = sum((week4.sample$tip_amount - fitted)^2) / length(fitted)
MSPEsample
```

```
## [1] 4.201155
```

```r
prediction.test = week4.sample %>%
  filter(tip_amount == 1.72) %>%
  select(-tip_amount) %>%
  slice(1)

prediction.test
```

```
## # A tibble: 1 x 8
##   passenger_count trip_distance RatecodeID    fare_amount extra time
##             <dbl>         <dbl> <fct>               <dbl> <dbl> <chr>
## 1               1          1.67 Standard rate          13     0 Morning
## # ... with 2 more variables: pickup_location <chr>, pickupday_type <chr>
```

```r
# Using the beta calculated,and a row of values with tip of 1.72
prediction = as.numeric(betahat[1] + betahat[2] *prediction.test[1] + betahat[3] * prediction.test[2]
prediction
```

```
## [1] 1.666406
```

```r
apparent_error = sum((week2.sample$tip_amount - yhat2))/ (nrow(week2.sample) - length(betahat))
apparent_error
```

```
## [1] -1.786346e-14
```

```r
summary(week2.sample$tip_amount)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   0.000   1.350   1.719   2.260 520.380
```

I have sampled 10% of the week 4 dataset due to limitation in computational power. Using the sample, I have calculated the MSPE of 2.29% and apparent error is almost nil. The MSPE drastically increased in comparison to the apparent error but it's not that bad. Although there is a huge range in the tip amount from week 2 sample, the MSPE is only 2.29%. This means that the model fitted using week 2 sample does a decent job in estimating for week 4 sample. This could have been better if the model had interaction terms; the model could have been fitted 2 way interaction and use the power of regsubsets to get the best model for predicting tips. However, due to constraint in computational resources a simple linear model was fitted.

To put in context, we have used a row of with a tip of 1.72 and used the beta we calculated and found out that the model calculated 1.806589. The model has predicted the mean tip within about 3% error which is a decent accuracy.

There was also a problem with specifying borough as locations had to be estimated using latitude and longitude and unless the location is square shaped, it had to be a rough estimation. It was also beyond of the course, so location had to be roughly estimated.

Face validity of the model: The whole point of the assignment was to construct a model that predicts the amount of a tip and we have. On the surface level, the model predicts the amount of tip but it only predicts tip that is paid by credit cards. From the plot where it shows proportion of the payment type depending on the day, we can clearly observe that on average there is 65% proportion on paying by credit card. This means that on average, 35% is neglected because it was never recorded in the dataset and that is a good amount of cash uses. Therefore, the model only predicts tips paid by credit cards and it misses out on the

35% of the cash tips so the model does not accurately predict the amount of tip. To conclude, the model does not predict properly on tips due to the missing data.

Given above information, the model's predictability has space for improvement and once the lock down is finished, I could use the school computer to try the 2 way interaction model and see how much MSPE have improved or worsened. However, for now we will leave it.