M2.951 - Tipologia i cicle de vida de les dades

Pràctica 2 - Neteja i anàlisi de les dades

Autors: Roger Perís Serrano i Albert Cámara Viñals

Desembre, 2020

Contents

Detalls de l'activitat

Presentació

En aquesta pràctica s'elabora un cas pràctic orientat a aprendre a identificar les dades rellevants per un projecte analític i usar les eines d'integració, neteja, validació i anàlisi de les mateixes. Per fer aquesta pràctica haureu de treballar en grups de 2 persones. Haureu de lliurar un sol fitxer amb l'enllaç Github (https://github.com) on es trobin les solucions incloent els noms dels components de l'equip. Podeu utilitzar la Wiki de Github per descriure el vostre equip i els diferents arxius que corresponen a la vostra entrega. Cada membre de l'equip haurà de contribuir amb el seu usuari Github. Malgrat que no es tracta del mateix enunciat, els següents exemples d'edicions anteriors us poden servir com a guia:

- Exemple: https://github.com/Bengis/nba-gap-cleaning
- Exemple complex (fitxer adjunt).

Competències

En aquesta pràctica es desenvolupen les següents competències del Màster de Data Science:

- Capacitat d'analitzar un problema en el nivell d'abstracció adequat a cada situació i aplicar les habilitats i coneixements adquirits per abordar-lo i resoldre'l.
- Capacitat per aplicar les tècniques específiques de tractament de dades (integració, transformació, neteja i validació) per al seu posterior anàlisi.

Objectius

Els objectius concrets d'aquesta pràctica són:

- Aprendre a aplicar els coneixements adquirits i la seva capacitat de resolució de problemes en entorns nous o poc coneguts dintre de contextos més amplis o multidisciplinaris.
- Saber identificar les dades rellevants i els tractaments necessaris (integració, neteja i validació) per dur a terme un projecte analític.
- Aprendre a analitzar les dades adequadament per abordar la informació continguda en les dades.
- Identificar la millor representació dels resultats per tal d'aportar conclusions sobre el problema plantejat en el procés analític.
- Actuar amb els principis ètics i legals relacionats amb la manipulació de dades en funció de l'àmbit d'aplicació.
- Desenvolupar les habilitats d'aprenentatge que els permetin continuar estudiant d'una manera que haurà de ser en gran manera autodirigida o autònoma.
- Desenvolupar la capacitat de cerca, gestió i ús d'informació i recursos en l'àmbit de la ciència de dades.

Descripció de la Pràctica a realitzar

L'objectiu d'aquesta activitat serà el tractament d'un dataset, que pot ser el creat a la pràctica 1 o bé qualsevol dataset lliure disponible a Kaggle (https://www.kaggle.com). Alguns exemples de dataset amb els que podeu treballar són:

- Red Wine Quality (https://www.kaggle.com/uciml/red-wine-quality-cortez-et-al-2009).
- Titanic: Machine Learning from Disaster (https://www.kaggle.com/c/titanic).

L'últim exemple correspon a una competició activa a Kaggle de manera que, opcionalment, podeu aprofitar el treball realitzat durant la pràctica per entrar en aquesta competició.

Seguint les principals etapes d'un projecte analític, les diferents tasques a realitzar (i justificar) són les següents:

1. Descripció del dataset. Perquè és important i quina pregunta/problema pretén respondre?
2. Integració i selecció de les dades d'interès a analitzar.
3. Neteja de les dades.

- Les dades contenen zeros o elements buits? Com gestionaries aquests casos?
- Identificació i tractament de valors extrems.

4. Anàlisi de les dades.

- Selecció dels grups de dades que es volen analitzar/comparar (planificació delsanàlisis a aplicar).
- Comprovació de la normalitat i homogeneïtat de la variància.
- Aplicació de proves estadístiques per comparar els grups de dades. En funció de les dades i de l'objectiu de l'estudi, aplicar proves de contrast d'hipòtesis, correlacions, regressions, etc. Aplicar almenys tres mètodes d'anàlisi diferents.

5. Representació dels resultats a partir de taules i gràfiques.
6. Resolució del problema. A partir dels resultats obtinguts, quines són les conclusions? Els resultats permeten respondre al problema?

7. Codi: Cal adjuntar el codi, preferiblement en R, amb el que s'ha realitzat la neteja, anàlisi i representació de les dades. Si ho preferiu, també podeu treballar en Python.

Recursos

Els següents recursos són d'utilitat per la realització de la pràctica:

- Calvo M., Subirats L., Pérez D. (2019). Introducción a la limpieza y análisis de los datos. Editorial UOC.
- Megan Squire (2015). Clean Data. Packt Publishing Ltd.
- Jiawei Han, Micheine Kamber, Jian Pei (2012). Data mining: concepts and techniques. Morgan Kaufmann.
- Jason W. Osborne (2010). Data Cleaning Basics: Best Practices in Dealing with Extreme Scores. Newborn and Infant Nursing Reviews; 10 (1): pp. 1527-3369.
- Peter Dalgaard (2008). Introductory statistics with R. Springer Science & Business Media.
- Wes McKinney (2012). Python for Data Analysis. O'Reilley Media, Inc.
- Tutorial de Github https://guides.github.com/activities/hello-world.

Criteris de valoració

Tots els apartat són obligatoris. La ponderació dels exercicis és la següent:

- Els apartats 1, 2 i 6 valen 0,5 punts.
- Els apartats 3, 5 i 7 valen 2 punts.
- L'apartat 4 val 2,5 punts.

Es valorarà la idoneïtat de les respostes, que han de ser clares i completes. Les diferents etapes han d'estar ben justificades i acompanyades del codi corresponent. També es valorarà la síntesi i claredat, a través de l'ús de comentaris, del codi resultant, així com la qualitat de les dades finals analitzades.

Format i data de lliurament

Durant la setmana del 21 al 25 de desembre el grup podrà lliurar al professor una entrega parcial opcional. Aquesta entrega parcial és molt recomanable per tal de rebre assessorament sobre la pràctica i verificar que la direcció presa és la correcta. Es lliuraran comentaris als estudiants que hagin efectuat l'entrega parcial però no comptarà per la nota de la pràctica. En l'entrega parcial els estudiants hauran de lliurar per correu electrònic, al profesor encarregat de l'aula, l'enllaç al repositori Github amb el que hagin avançat.

Pel que fa a l'entrega final, cal lliurar un únic fitxer que contingui l'enllaç a Github, el qual no es podrà modificar posteriorment a la data d'entrega, on hi hagi:

1. Una Wiki on hi hagi els noms dels components del grup i una descripció dels fitxers.
2. Un document Word, Open Office o PDF amb les respostes a les preguntes i els noms dels components del grup. A més, al final de document, haurà d'aparèixer la següent taula de contribucions al treball, la qual ha de signar cada integrant del grup amb les seves inicials. Les inicials representen la confirmació de que l'integrant ha participat en aquell apartat. Tots els integrants han de participar en cadascun dels apartats, de manera que, idealment, els apartats hauran d'estar signats per tots els integrants.
3. Una carpeta amb el codi generat per analitzar les dades.
4. El fitxer CSV amb les dades originals.
5. El fitxer CSV amb les dades finals analitzades.

Aquest document de l'entrega final de la Pràctica 2 s'ha de lliurar a l'espai de Lliurament i Registre d'AC de l'aula abans de les 23:59 del dia 5 de gener. No s'acceptaran lliuraments fora de termini.

Resolució

Contribucions	Firma
Investigació prèvia	RPS, ACV
Redacció de les respostes	RPS, ACV
Desenvolupament codi	RPS, ACV

```
# Carreguem les llibreries que farem servir
library(tidyverse)
library(ggplot2)
library(dplyr)
library(GGally)
library(arules)
library(reshape2)
library(regclass)
library(car)
library(nortest)
library(gridExtra)
library(ResourceSelection)
library(DescTools)
library(vcd)
library(ggpubr)
library(corrplot)
library(rpart)
library(rpart.plot)
library(gmodels)
library(rattle)
library(Metrics)
library(grid)
library(gridExtra)

old <- theme_set(theme_minimal())
```

Descripció del dataset

El conjunt de dades que analitzarem serà: Cornell Car Rental Dataset, s'ha obtingut del repositori de dades Kaggle (https://www.kaggle.com/kushleshkumar/cornell-car-rental-dataset?select=CarRentalData.csv). Aquest joc de dades és un recull de registres de diferents portals de lloguer de vehicles a les principals ciutats d'Estats Units, i s'ha generat mitjançant *webscraping*, amb una extracció relitzada al Julol del 2020. Està format per 16 característiques (columnes o atributs) que presentan 5851 registres (files o observacions). Cada una de les obseravacions es correspon a les característiques d'un vehicle de lloguer.

Les característiques, atributs o variables per cada observació (es manté el nom original en anglès) són:

- *fuelType:* Tipus de combustible utilitzat pel vehicle.
- *rating:* Qualificació acumulativa del cotxe per part dels clients
- *renterTripsTaken:* Nombre de viatges realitzats per aquest vehicle (durada desconeguda)
- *reviewCount:* Nombre de valoracions
- *location.city:* Ciutat en què es troba el vehicle
- *location.country:* País en què es troba el vehicle
- *location.latitude:* Coordenada geogràfica (latitud) en què es troba el vehicle
- *location.longitude:* Coordenada geogràfica (longitud) en què es troba el vehicle
- *location.state:* Estat en què es troba el vehicle
- *owner.id:* Identificador (ID) del propietàri del vehicle
- *rate.daily:* Tarifa diària en dòlars

- *vehicle.make:* Marca del vehicle
- *vehicle.model:* Model del vehicle
- *vehicle.type:* Tipus de vehicle
- *vehicle.year:* Any del vehicle (matriculació)
- *airportcity:* Ciutat de l'aeroport més proper a la localització en què es troba el vehicle (normalment es des d'on es realitza el lloguer del vehicle).

Importància i objectius de l'anàlisi

Amb l'entrada en vigor de les zones de zero emissions a les grans ciutats, la classificació de vehicles segons el seu grau de contaminació i de l'objectiu de zero emissions al 2050, promogut per la Comisió Euopea han generat un gran debat envers la compra o lloguer de vehicles. De fet hi ha diversos estudis que mostren que en els últims anys hi ha hagut un increment en el mercat de lloguer de vehicles o *renting*. Per tant a partir del joc de dades escollit "Cornell Car Rental Dataset" ens proposem:

- Realitzar una anàlisi detallada dels atributs propis del sector del lloguer de vehicles per tal d'extreuren nou coneixement i que aquest pugui aportar valor.
- Dur a terme diferents contrastos d'hipótesis que permetin identificar propietats interessants subjacents en les mostres que puguin ser inferides respecte a la població.
- Generar un model que permeti preveure el preu per dia d'un automòbil de lloguer.
- Generar un model que permeti preveure, donades les característiques d'un vehicle de lloguer, si aquest serà llogat o no.
- Generar un model que pugui preveure els ingressos anuals que aportarà un automòbil de lloguer com a aproximació als beneficis que aportaria.

Aquests anàlisis són de gran rellevància en gairebé qualsevol sector relacionat amb el lloguer de vehicles. Per una banda pot resultar interessant per les agències de lloguer, ja que aquestes podran conèixer amb més detall els seus clients, i per tant generar ofertes personalitzades o decidir quin és el preu més adient o quin model de vehicle és més adequat per renovar la flota. D'altre banda també pot resultar interessant per a l'usuari final, ja que a partir de les seves necesitats podrà determinar si li convé llogar un vehicle o no o si el seu preu és correspont amb el del mercat actual.

Neteja de dades

Lectura de dades

En primer lloc, obrim el fitxer de dades i examinem el tipus de dades amb els que R ha interpretat cada variable. A més examinem també els valors resum de cada tipus de variable.

Comencem carregant el joc de dades en un dataframe d'*R*.

```
# Carreguem el joc de dades
ds <- read.csv('../data/CarRentalDataV1.csv', stringsAsFactors = FALSE, header = TRUE, sep=',', strip.wh
```

Tot seguit examinem l'estructura del joc de dades, per validar que s'han interpetat correctament.

```
# Verifiquem les dimensions del joc de dades
dim(ds)
```

```
## [1] 5851   16
```

```
# Verifiquem l'estructura del joc de dades
str(ds)
```

```
## 'data.frame':    5851 obs. of  16 variables:
##  $ fuelType          : chr  "ELECTRIC" "ELECTRIC" "HYBRID" "GASOLINE" ...
##  $ rating            : num  5 5 4.92 5 5 5 4.42 4.9 5 4.76 ...
```

```
##  $ renterTripsTaken  : num  13 2 28 21 3 13 13 12 1 22 ...
##  $ reviewCount       : num  12 1 24 20 1 12 12 10 1 17 ...
##  $ location.city     : chr  "Seattle" "Tijeras" "Albuquerque" "Albuquerque" ...
##  $ location.country  : chr  "US" "US" "US" "US" ...
##  $ location.latitude : num  47.4 35.1 35.1 35.1 35.2 ...
##  $ location.longitude: num  -122 -106 -107 -107 -107 ...
##  $ location.state    : chr  "WA" "NM" "NM" "NM" ...
##  $ owner.id          : num  12847615 15621242 10199256 9365496 3553565 ...
##  $ rate.daily        : num  135 190 35 75 47 58 42 117 102 49 ...
##  $ vehicle.make      : chr  "Tesla" "Tesla" "Toyota" "Ford" ...
##  $ vehicle.model     : chr  "Model X" "Model X" "Prius" "Mustang" ...
##  $ vehicle.type      : chr  "suv" "suv" "car" "car" ...
##  $ vehicle.year      : num  2019 2018 2012 2018 2010 ...
##  $ airportcity       : chr  "Albuquerque" "Albuquerque" "Albuquerque" "Albuquerque" ...
```

A partir dels detalls anteriors podem veure que disposem d'un joc de dades amb 5851 observacions, amb 16 atributs o variables per observació i que el tipus de dades s'ha interpretat correctament.

Previsualització de les dades d'interès

A continuació realitzem una previsualització, anàlisi visual de les dades, per intentar identificar les possibles anomalies, distribucions i característiques de les diferents variables.

Per això el primer que farem serà distingir les variables categòriques de les variables numèriques. la qual cosa ens facilitarà el tractament futur.

```
categorical_features_names = c('fuelType', 'location.city', 'location.country', 'location.state', 'owne

numeric_features_names = c('rating', 'renterTripsTaken', 'reviewCount', 'location.latitude', 'location.l
```

Anàlisi univariant

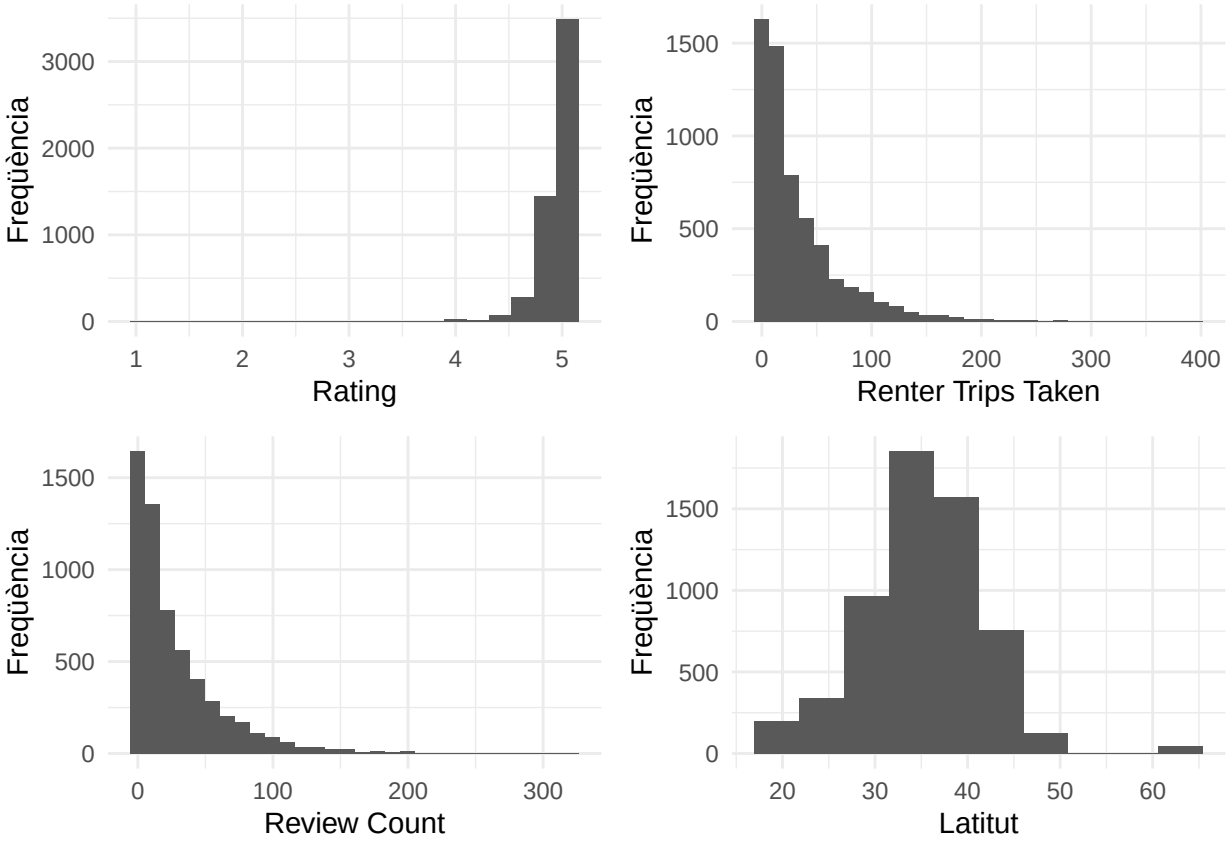
Comencem visualitzant els principals descriptors estadístics de cadascuna de les variables numèriques.

```
# Mostrem un resum dels principals estadístics de cada variable
summary(ds[, numeric_features_names])
```

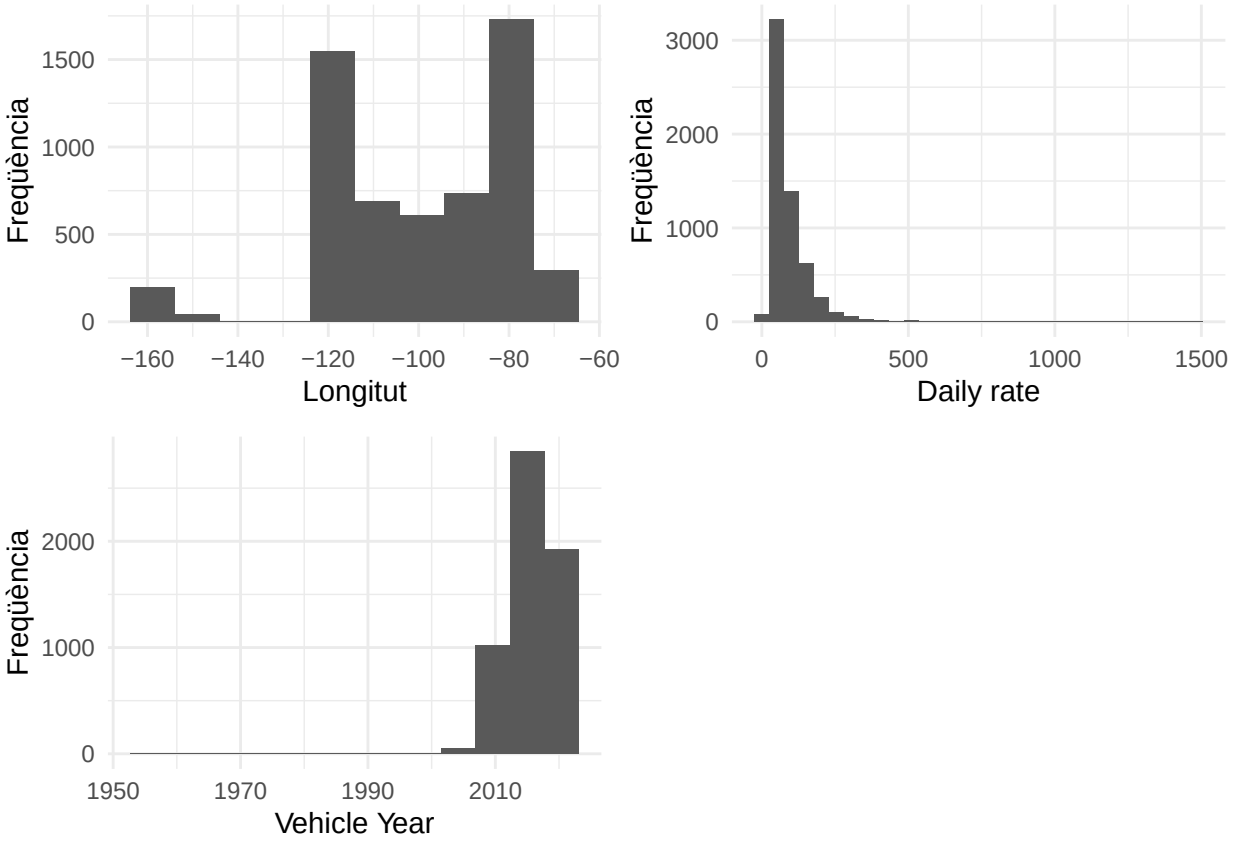
```
##      rating     renterTripsTaken  reviewCount     location.latitude
##  Min.   :1.00   Min.   :  0.00   Min.   :  0.00   Min.   :21.27
##  1st Qu.:4.90   1st Qu.:  5.00   1st Qu.:  4.00   1st Qu.:30.45
##  Median :5.00   Median : 18.00   Median : 16.00   Median :35.55
##  Mean   :4.92   Mean   : 33.48   Mean   : 28.45   Mean   :35.58
##  3rd Qu.:5.00   3rd Qu.: 46.00   3rd Qu.: 39.00   3rd Qu.:40.00
##  Max.   :5.00   Max.   :395.00   Max.   :321.00   Max.   :64.89
##  NA's   :501
##  location.longitude   rate.daily       vehicle.year
##  Min.   :-158.17    Min.   :  20.00   Min.   :1955
##  1st Qu.:-117.16    1st Qu.:  45.00   1st Qu.:2014
##  Median : -95.67    Median :  69.00   Median :2016
##  Mean   : -99.63    Mean   :  93.69   Mean   :2015
##  3rd Qu.: -81.54    3rd Qu.: 110.00   3rd Qu.:2018
##  Max.   : -68.82    Max.   :1500.00   Max.   :2020
##
```

Anàlisi dels atributs numèrics Tot seguit visualitzem les distribucions dels valors de les variables numèriques.

```r
# Calculem histogrames de les variables numèriques
hist01 <- ggplot(data=ds, aes(x=rating)) + geom_histogram(bins = 20) + xlab('Rating') + ylab('Freqüència
hist02 <- ggplot(data=ds, aes(x=renterTripsTaken)) + geom_histogram(bins=30) + xlab('Renter Trips Taken
hist03 <- ggplot(data=ds, aes(x=reviewCount)) + geom_histogram(bins=30) + xlab('Review Count') + ylab('
hist04 <- ggplot(data=ds, aes(x=location.latitude)) + geom_histogram(bins=10) + xlab('Latitut') + ylab(
hist05 <- ggplot(data=ds, aes(x=location.longitude)) + geom_histogram(bins=10) + xlab('Longitut') + yla
hist06 <- ggplot(data=ds, aes(x=rate.daily)) + geom_histogram(bins=30) + xlab('Daily rate') + ylab('Fre
hist07 <- ggplot(data=ds, aes(x=vehicle.year)) + geom_histogram(bins=13) + xlab('Vehicle Year') + ylab(

grid.arrange(hist01, hist02, hist03, hist04, nrow = 2, ncol = 2)
```




```r
grid.arrange(hist05, hist06, hist07, nrow = 2, ncol = 2)
```



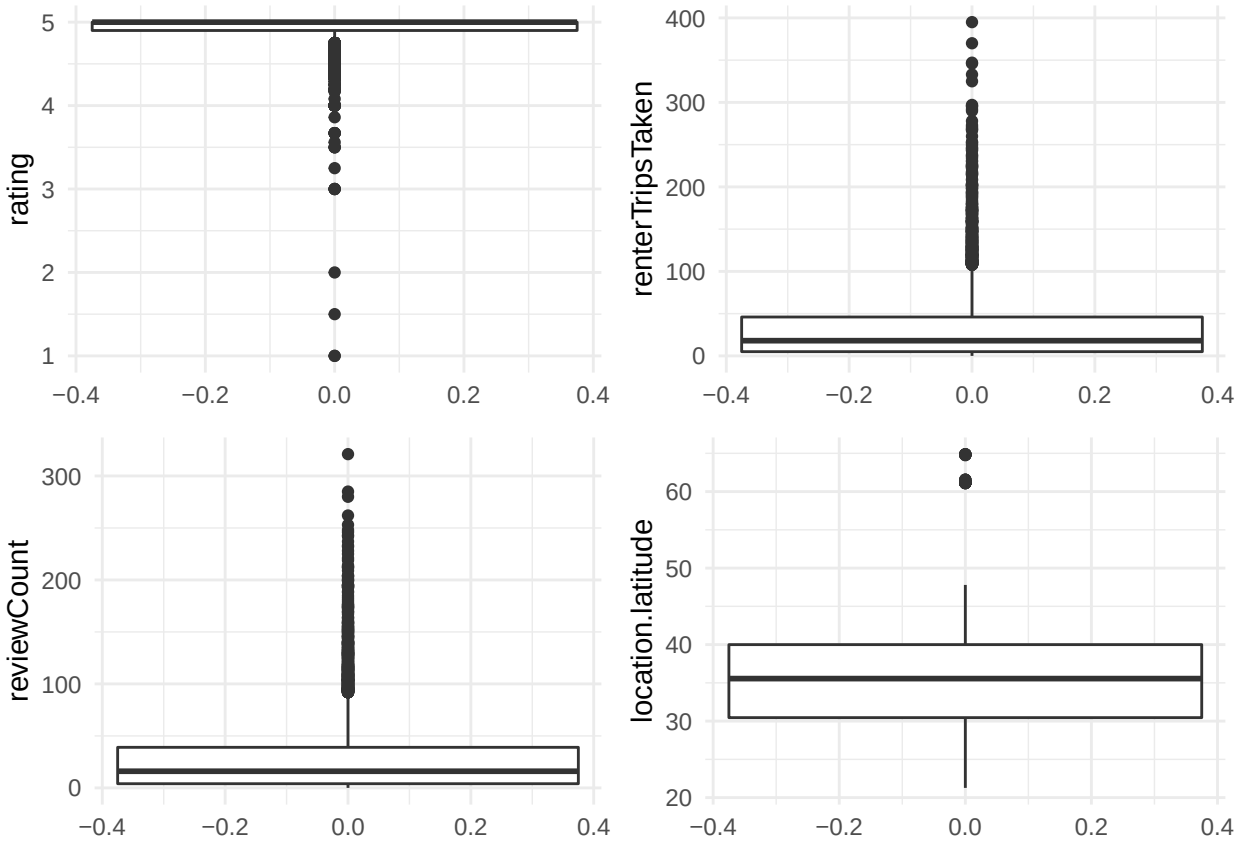
A partir de les visualitzacions anteriors podem fer les següents apreciacions:

- **rating:** Histograma unimodal amb cua a la dreta on podem observar que la gran majoria de vehicles reben valoracions elevades, en concret la puntuació màxima (5). A més s'observa que la presència de vehicles amb baixa puntuació és insignificant.
- **renterTripsTaken:** Histograma unimodal amb cua a l'esquerra, la qual cosa ens indica que els vehicles se solen llogar poques vegades, per sota de 100 vegades. I amb una mitjana de 33.48 lloguers.
- **reviewCount:** Histograma unimodal amb cua a l'esquerra, la qual cosa ens indica que els vehicles solen rebre poques valoracions, per sota de 100 vegades. I amb una mitjana de 28.45 valoracions.
- **location.latitude:** Histograma amb un distribució bastant normal. La qual cosa ens fa pensar que gran part dels vehicles es localitzen en una mateixa latitud, coordenada geogràfica (latitud) en què es troba el vehicle.
- **location.longitude:** Histograma bimodal, amb un pic prop de -120 i un altre pic prop de -80, de nou podem dir que la zona on s'estudia el lloguer de vehicles també queda bastant acotada horitzontalment, coordenada geogràfica (longitud) en què es troba el vehicle.
- **rate.daily:** Histograma unimodal amb cua a l'esquerra, la qual cosa ens indica que els vehicles se solen llogar per menys de 500 dòlars/dia. Amb un lloguer mitjà de 93.69 dòlars/dia, un lloguer mínim de 20 dòlars/dia i un lloguer màxim de 1500 dòlars/dia.
- **vehicle.year:** Histograma unimodal amb cua a l'esquerra, la qual cosa ens indica que els vehicles de lloguer no solen ser gaire antics. Sent la mitja de 5 anys, el vehicle més antic del 1955 i el vehicle més nou del 2020.

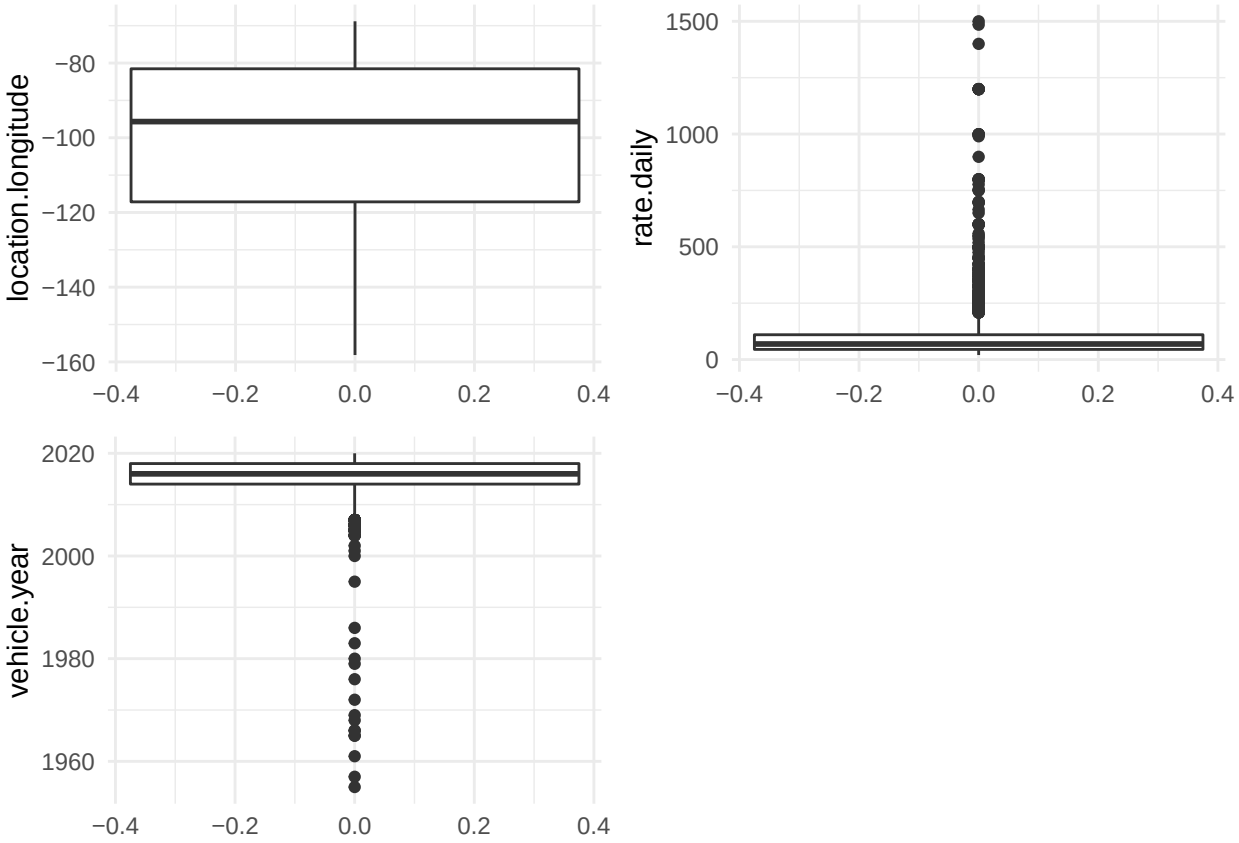
Un altre de les visualitzacions que sol ser de força utilitat per analitzar els jocs de dades són els diagrames de caixes, amb aquests podem observar la dispersió de les dades. A continuació visualitzarem els diagrames de caixes dels atributs numèrics.

```
# Calculem diagrames de caixa de les variables numèriques
box01 <-ggplot(ds, aes(y=rating)) + geom_boxplot()
```

```
box02 <-ggplot(ds, aes(y=renterTripsTaken)) + geom_boxplot()
box03 <-ggplot(ds, aes(y=reviewCount)) + geom_boxplot()
box04 <-ggplot(ds, aes(y=location.latitude)) + geom_boxplot()
box05 <-ggplot(ds, aes(y=location.longitude)) + geom_boxplot()
box06 <-ggplot(ds, aes(y=rate.daily)) + geom_boxplot()
box07 <-ggplot(ds, aes(y=vehicle.year)) + geom_boxplot()

grid.arrange(box01, box02, box03, box04,  nrow = 2, ncol = 2)
```




```
grid.arrange(box05, box06, box07, nrow = 2, ncol = 2)
```



A continuació podem veure un detall dels possibles valors extrems.

```
n <- length(ds$rating)

n - boxplot.stats(ds$rating)$n
```

```
## [1] 501
```

```
n - boxplot.stats(ds$renterTripsTaken)$n
```

```
## [1] 0
```

```
n - boxplot.stats(ds$reviewCount)$n
```

```
## [1] 0
```

```
n - boxplot.stats(ds$location.latitude)$n
```

```
## [1] 0
```

```
n - boxplot.stats(ds$location.longitude)$n
```

```
## [1] 0
```

```
n - boxplot.stats(ds$rate.daily)$n
```

```
## [1] 0
```

```
n- boxplot.stats(ds$vehicle.year)$n
```

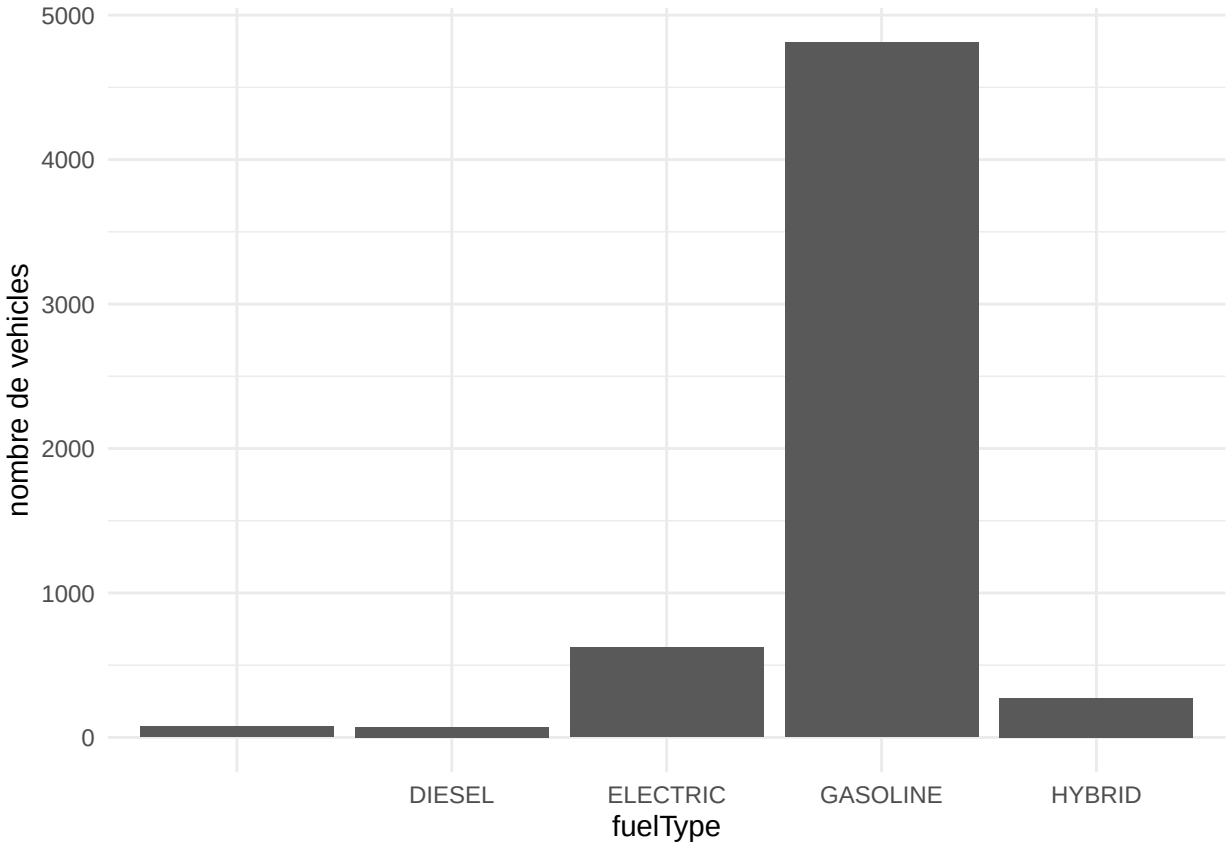
```
## [1] 0
```

A partir del resultats anteriors podem dir que la única variable que presenta valors extrems és les valoracions *'rating'*. En concret tenim 501 observacions que s'alluyen molt dels valors esperats per aquesta variable.

Anàlisi dels atributs categòrics Continuem visualitzant les distribucions dels valors de les variables categòriques

Revisem la variable fuelType

```
ggplot(data=ds,aes(x=fuelType))+geom_bar() + ylab("nombre de vehicles")
```



Podem observar que la majoria de vehicles son **GASOLINE**, seguits de lluny pels **ELECTRIC** i després pels **HYBRID**. Finalment els **DIESEL** son residuals i queda un subgrup sense etiquetar que el tractarem tot seguit.

Fem un petit tractament de valors Nuls. Aquest el realitzem calculant quin és el valor més freqüent de combustible i substituint els valors nuls per aquest.

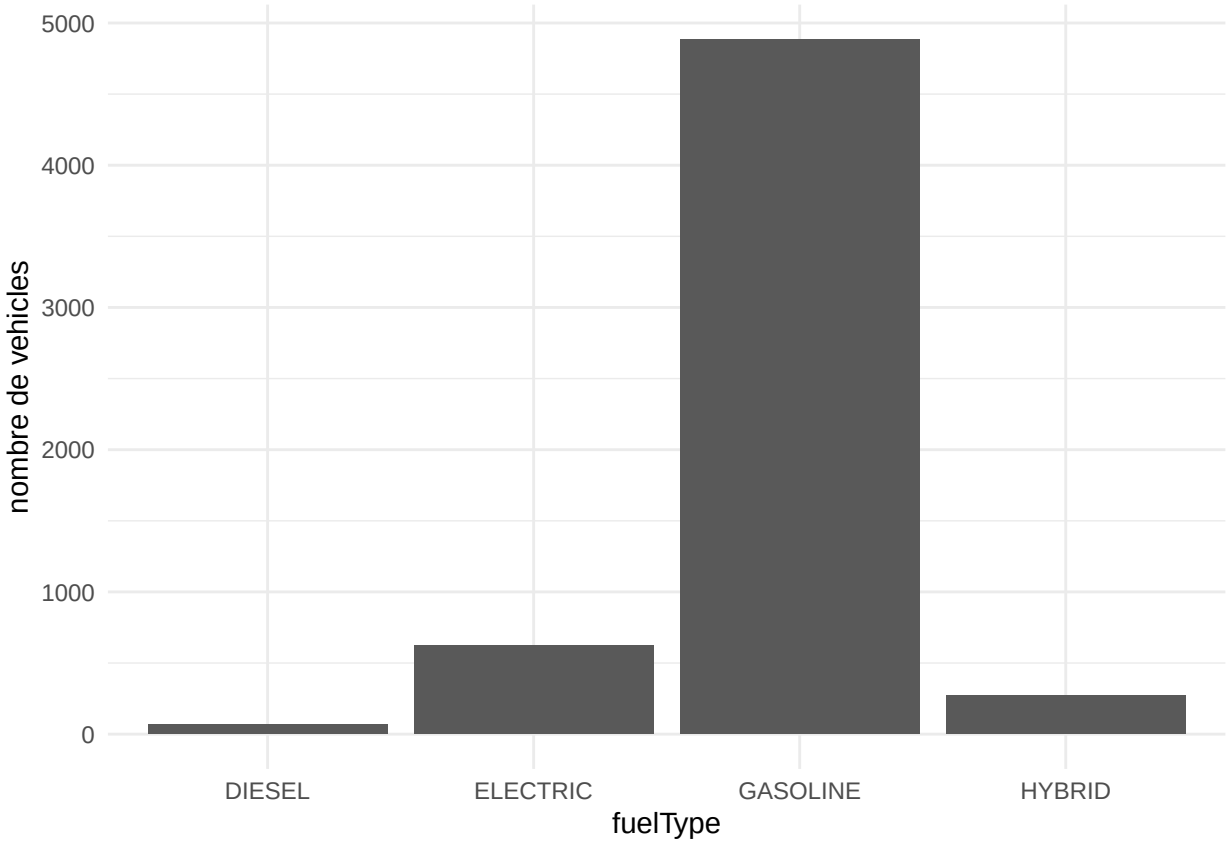
```
most_freq_fuelType <- names(which.max(table(ds$fuelType)))

most_freq_fuelType
```

```
## [1] "GASOLINE"
```

```
# Prenem el valor més freqüent per als valors buits
ds$fuelType[ds$fuelType == ""] = most_freq_fuelType
```

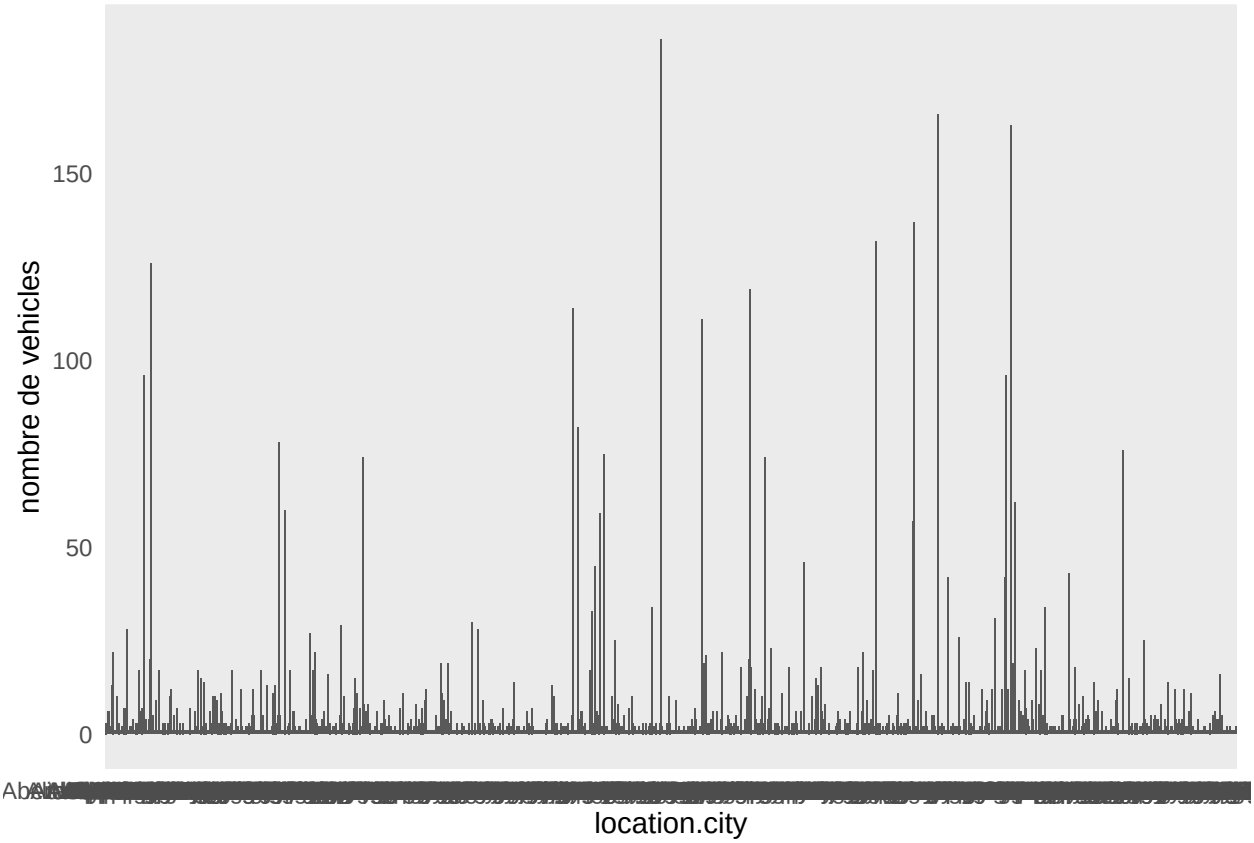
Grafiquem de nou **fuelType**

```
ggplot(data=ds,aes(x=fuelType))+geom_bar() + ylab("nombre de vehicles")
```



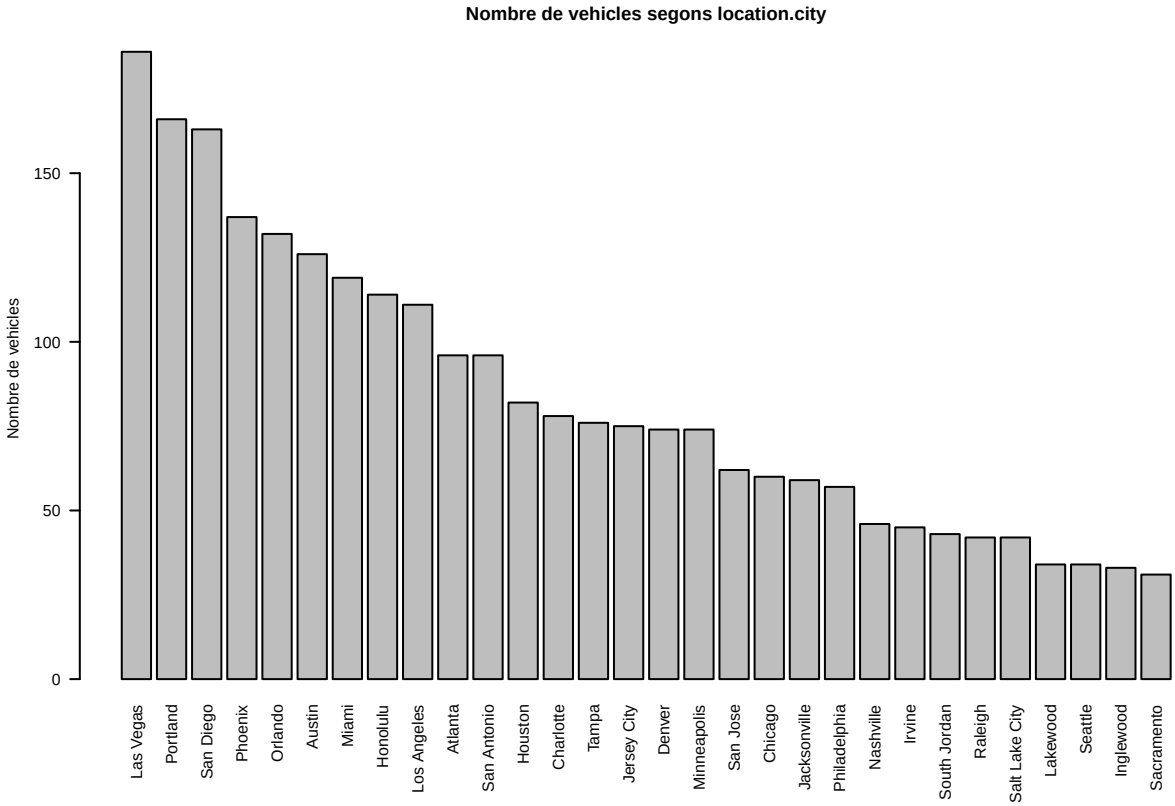
Revisem la variable location.city

```
ggplot(data=ds,aes(x=location.city))+geom_bar() + ylab("nombre de vehicles")
```



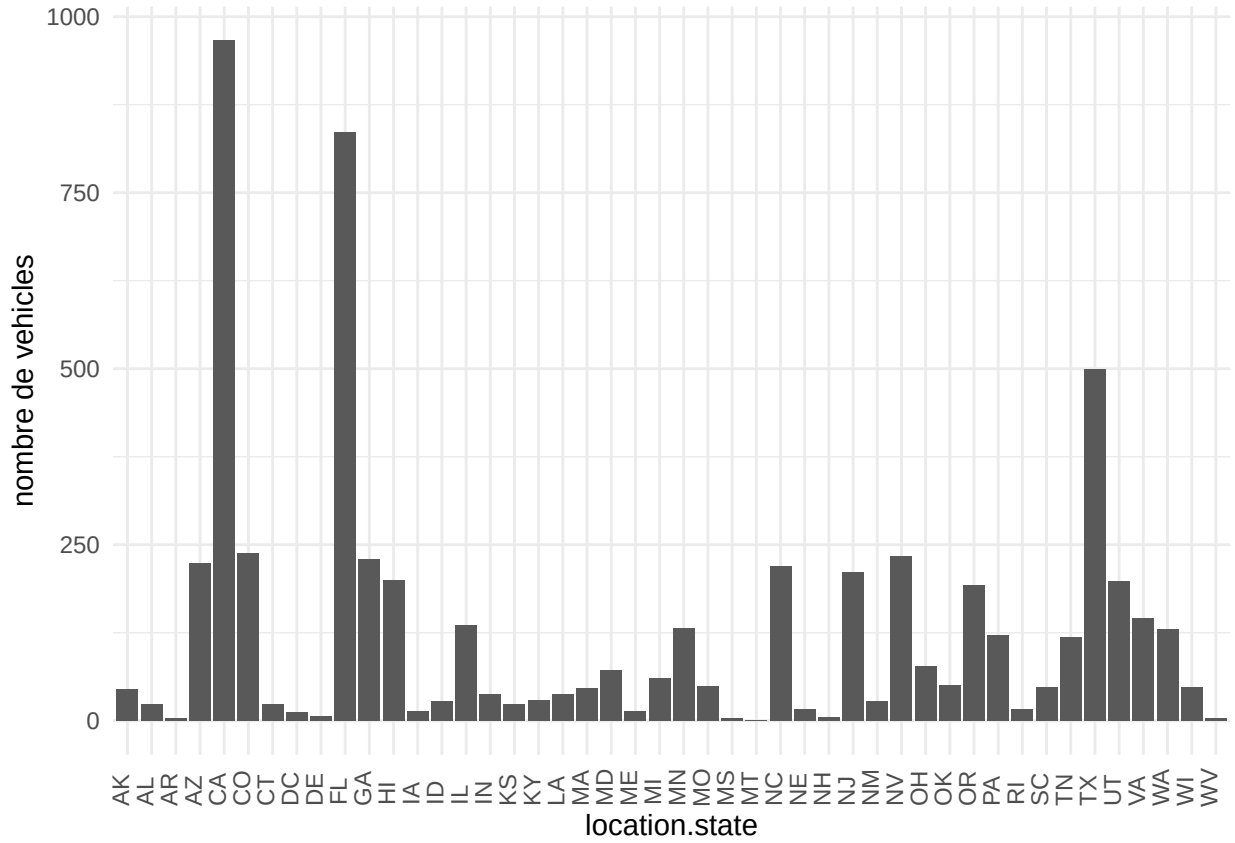
Veiem que hi ha tantes categories que és impossible revisar acuradament la gràfica. Llavors optem per graficar només les 30 categories més comuns.

```
# Seleccionem les dades
location.city_ordered <- sort(table(ds$location.city),decreasing = TRUE)

# Creem la visualització
par(las=2, cex=0.5, mar=c(8,4,4,2))
barplot(location.city_ordered[0:30], main="Nombre de vehicles segons location.city", ylab="Nombre de ve
```



Revisem la variable location.state

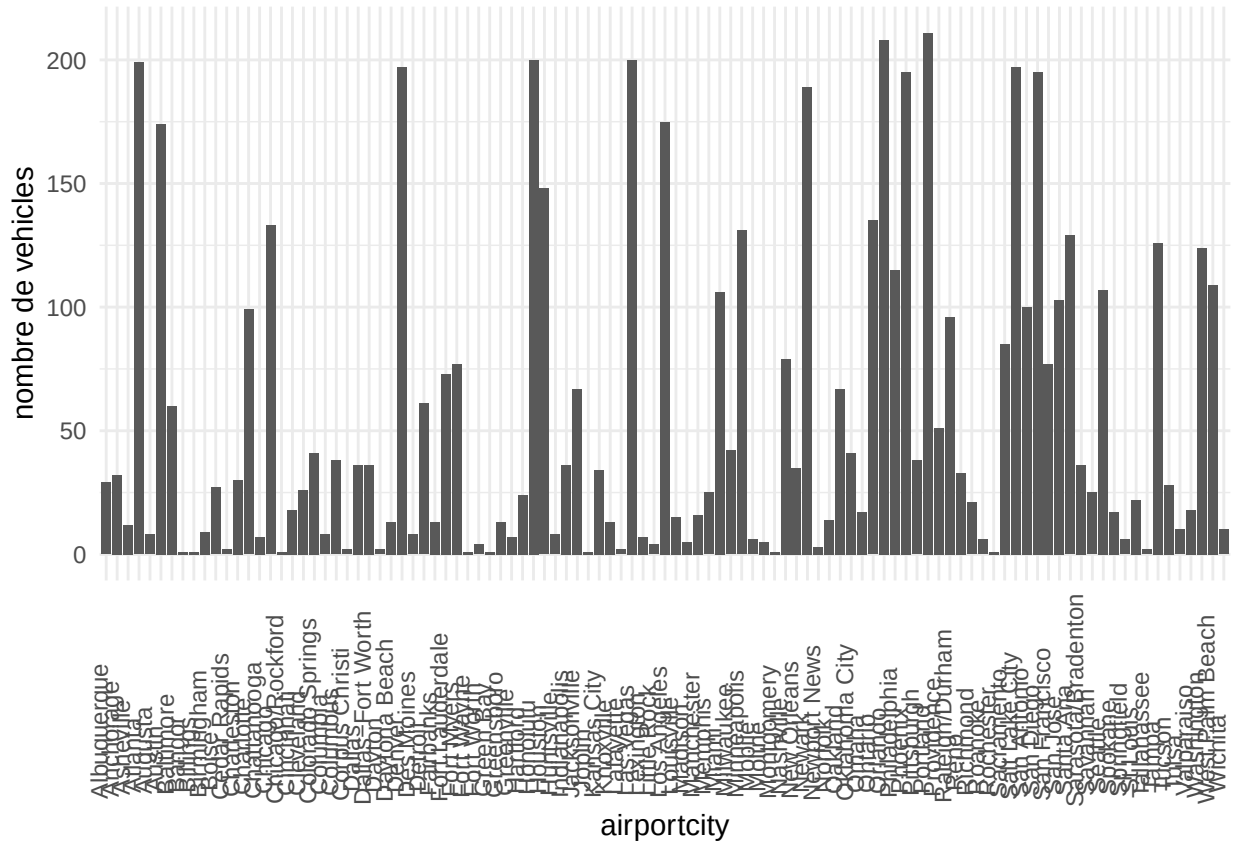
```
ggplot(data=ds,aes(x=location.state))+geom_bar()+theme(axis.text.x = element_text(angle = 90,vjust = 0.
```



Veiem que entre les categories destaquen **CA**, **FL** i **TX**.

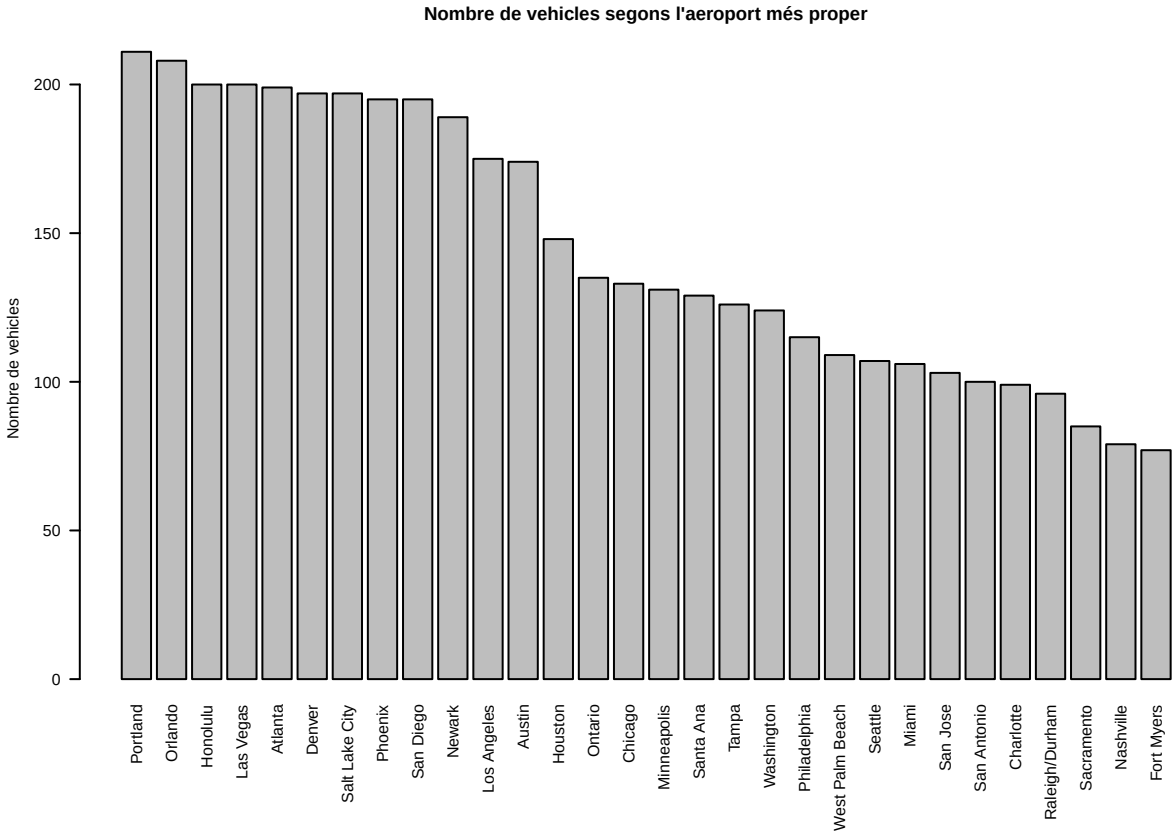
Revisem la variable airportcity

```
ggplot(data=ds,aes(x=airportcity))+geom_bar() + theme(axis.text.x = element_text(angle = 90,vjust = 0.1
```



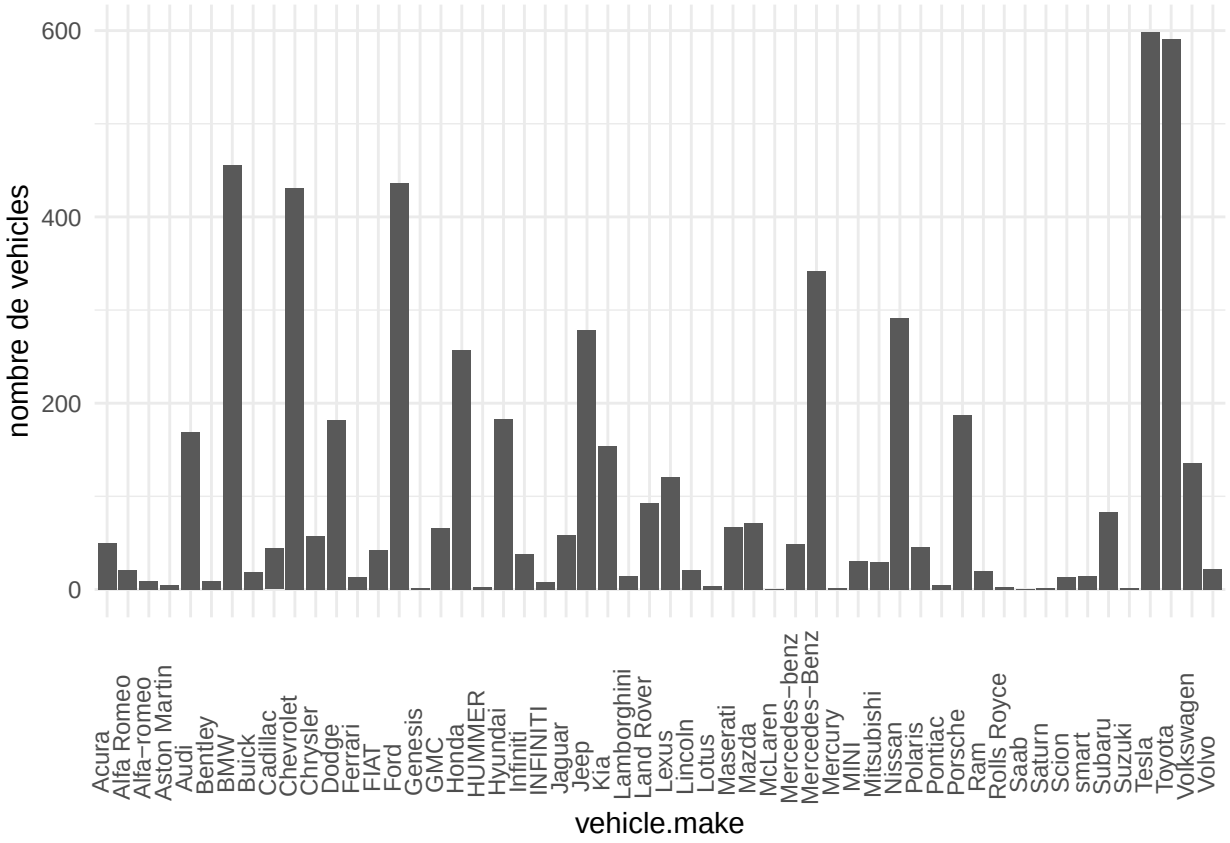
Tornem a trobar-nos que hi ha massa categories per a fer una visualització cómode. Llavors optem per graficar només les 30 categories més comuns.

```
# Seleccionem les dades
airportcity_ordered <- sort(table(ds$airportcity),decreasing = TRUE)
# Creem la visualització
par(las=2, cex=0.5, mar=c(8,4,4,2))
barplot(airportcity_ordered[0:30], main="Nombre de vehicles segons l'aeroport més proper", ylab="Nombre
```



Revisem la variable vehicle.make

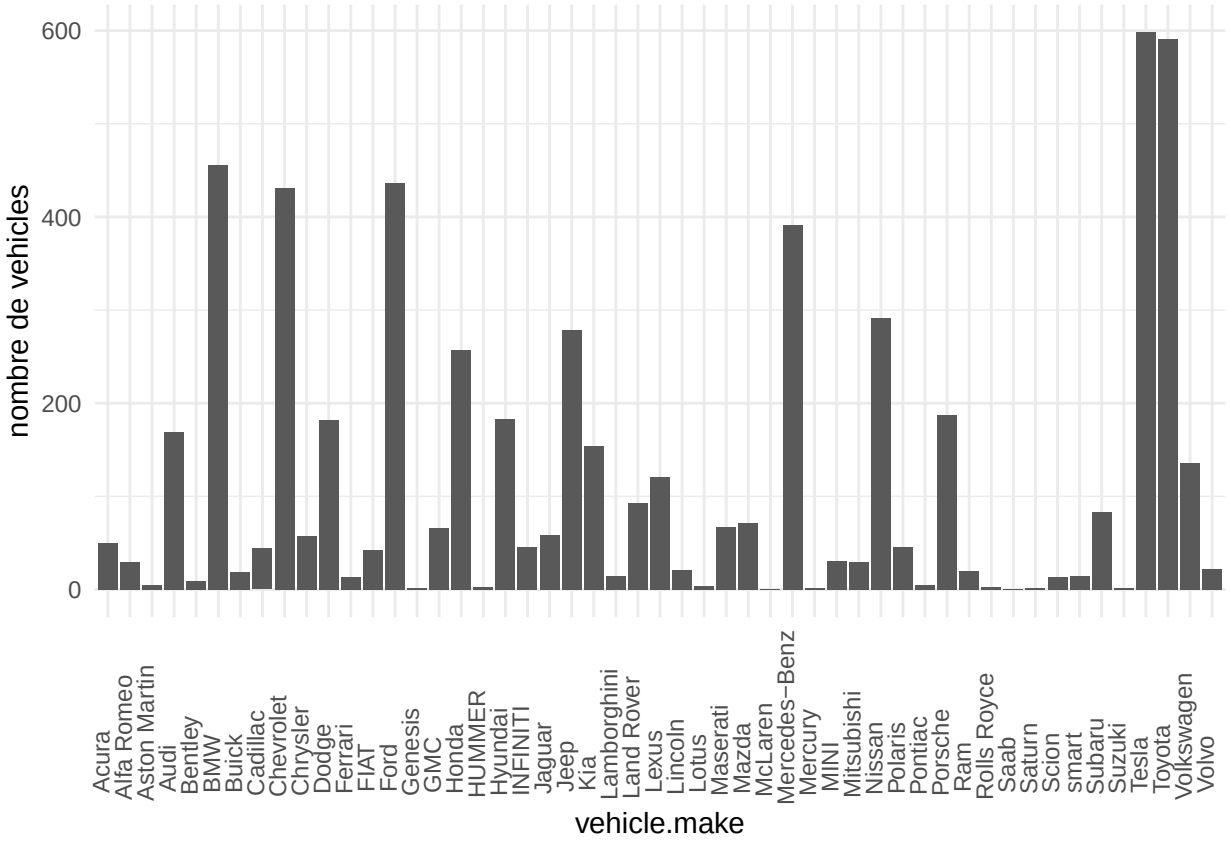
```
ggplot(data=ds,aes(x=vehicle.make))+geom_bar() + theme(axis.text.x = element_text(angle = 90,vjust = 0.
```



Aquí ens trobem al límit del que sería una visualització raonable, i veiem que hi ha categories repetides ja que es troben en algún cas mal escrites (per falta de lletres majúscules principalment).

Per tant realitzem una primera correcció de les categories per a continuació reproduir de nou la visualització.

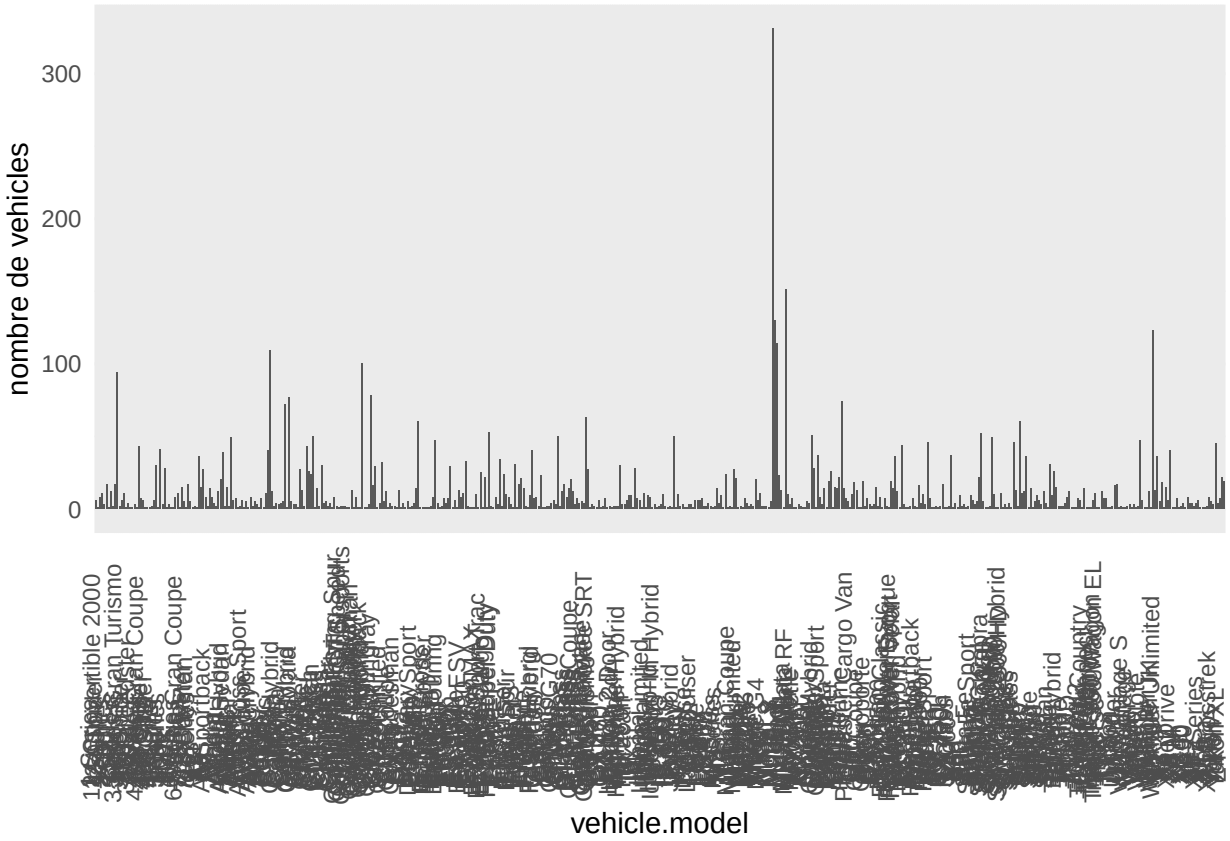
```
ds$vehicle.make[ds$vehicle.make == "Alfa-romeo"] = "Alfa Romeo"
ds$vehicle.make[ds$vehicle.make == "Infiniti"] = "INFINITI"
ds$vehicle.make[ds$vehicle.make == "Mercedes-benz"] = "Mercedes-Benz"
ggplot(data=ds,aes(x=vehicle.make))+geom_bar() + theme(axis.text.x = element_text(angle = 90,vjust = 0.
```



A la vista de les dades es pot observar que **TESLA** i **TOYOTA** son les marques més freqüents, seguides de **BMW**, **CHEVROLET**, **FORD** i **Mercedes-Benz**.

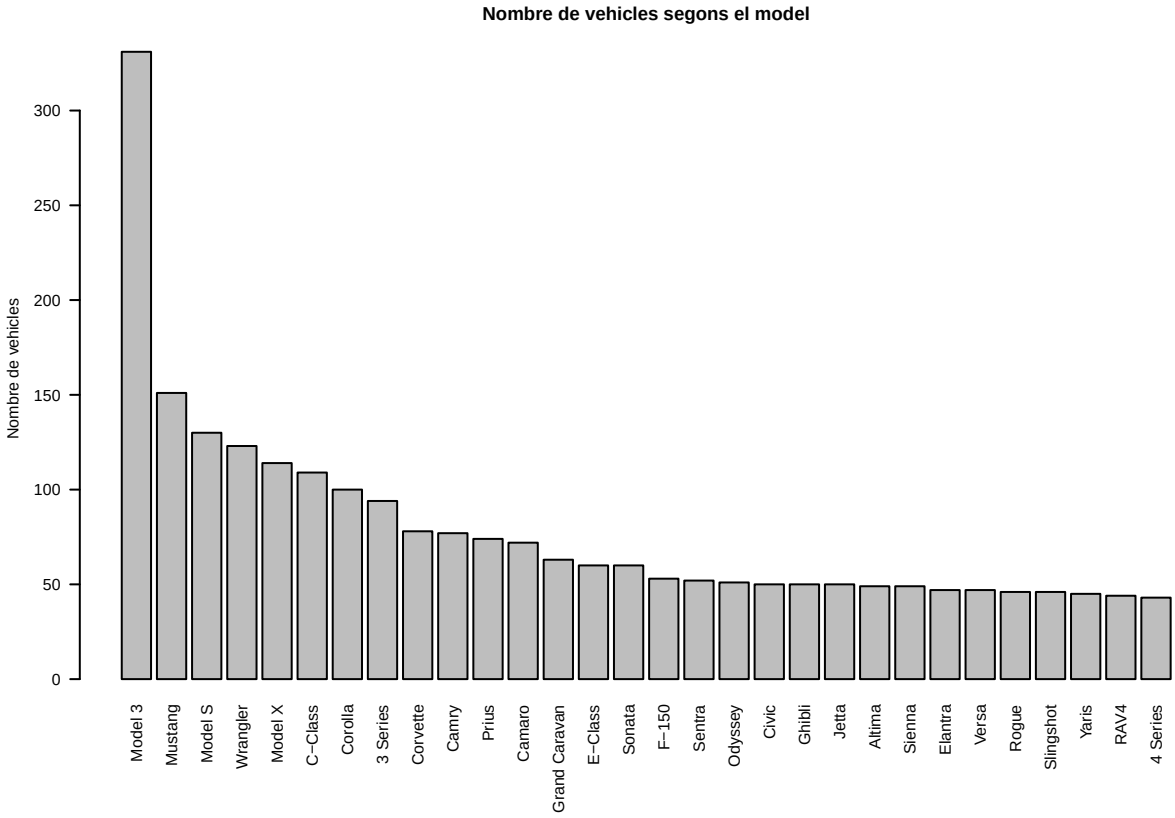
Revisem la variable vehicle.model

```
ggplot(data=ds,aes(x=vehicle.model))+geom_bar() + theme(axis.text.x = element_text(angle = 90,vjust = 0
```



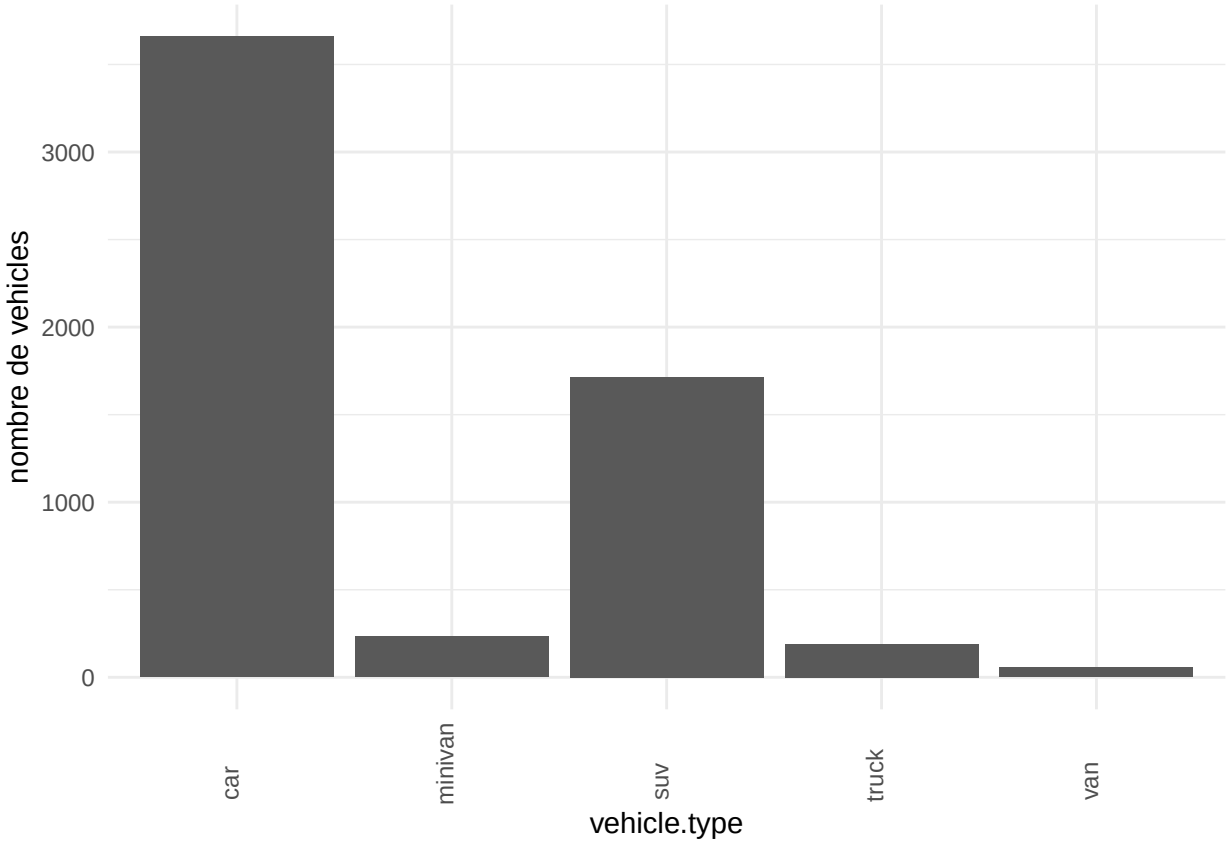
Tornem a trobar-nos que hi ha massa categories per a fer una visualització cómode. Llavors optem per graficar només les 30 categories més comuns.

```
# Seleccionem les dades
vehicle.model_ordered <- sort(table(ds$vehicle.model),decreasing = TRUE)

# Creem la visualització
par(las=2, cex=0.5, mar=c(8,4,4,2))
barplot(vehicle.model_ordered[0:30], main="Nombre de vehicles segons el model", ylab="Nombre de vehicles
```



Revisem la variable vehicle.type

```
ggplot(data=ds,aes(x=vehicle.type))+geom_bar() + theme(axis.text.x = element_text(angle = 90,vjust = 0.
```

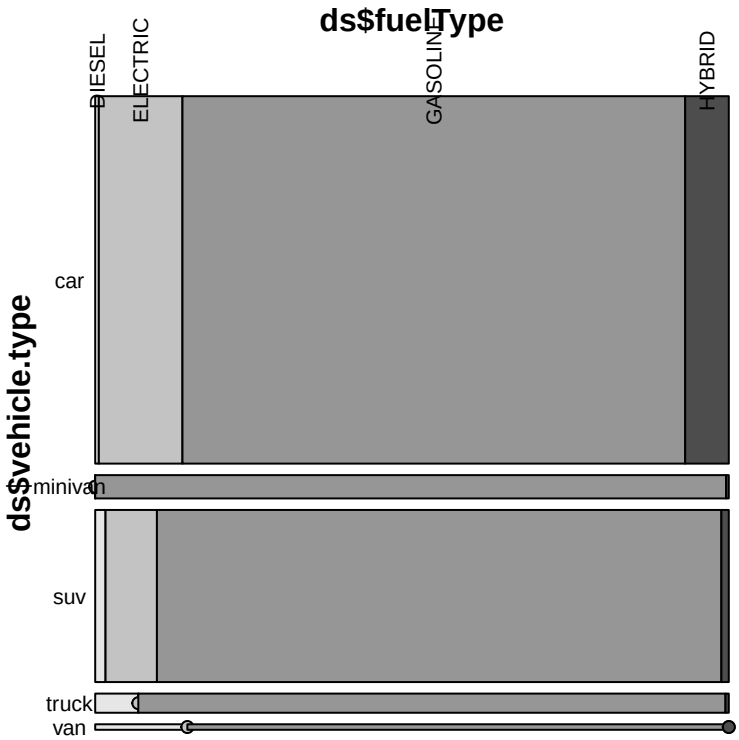


S'observa que la categoria **car** seguida de lluny per **SUV** són les més comuns, sent la resta (**minivan**, **truck** i **van**) gairebé residuals.

Visualització multivariant de les dades categòriques

Usem el gràfic de mosaic per visualitzar la convinació entre **fuelType** i **vehicle.type**

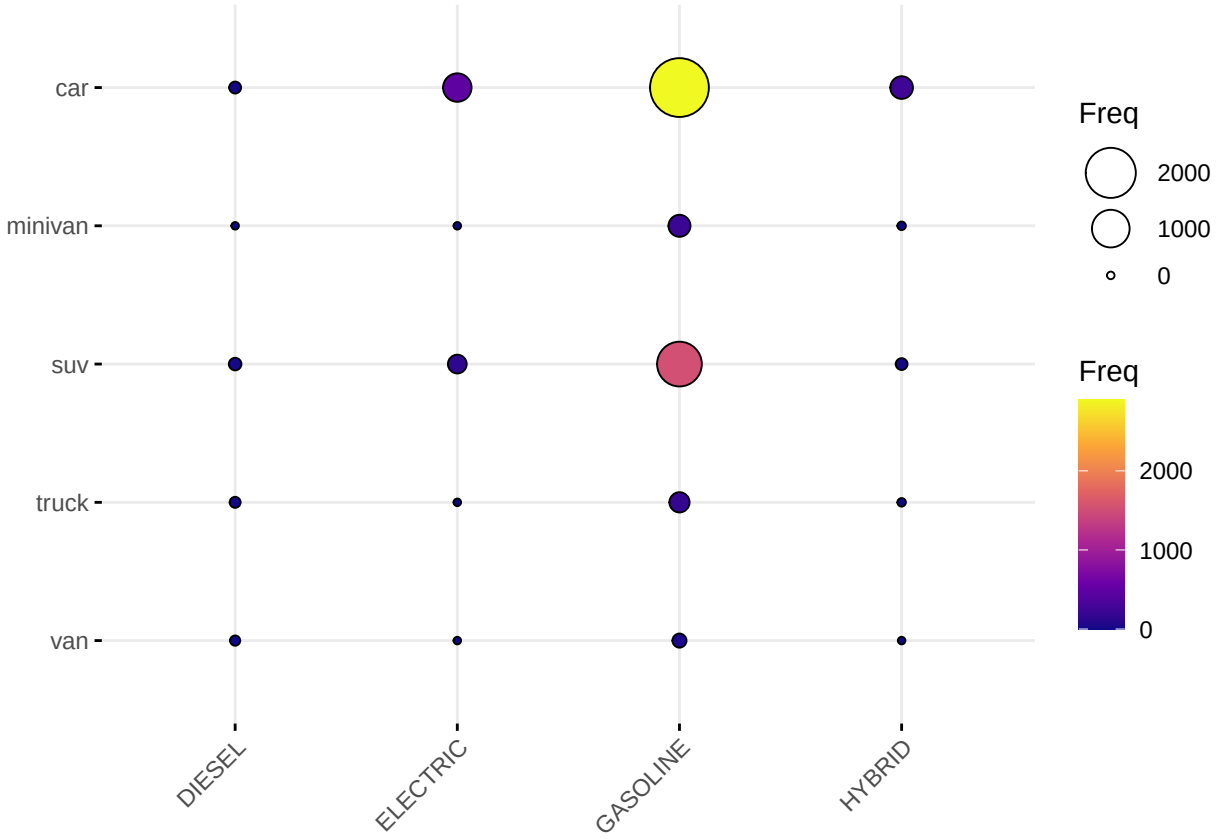
```
mosaic(ds$fuelType ~ ds$vehicle.type, gp_labels = gpar(fontsize = 8), las = 2, cex.axis = 2, rot_labels=
```



Gràcies al gràfic de mosaic es pot observar que només hi ha la categoria **HYBRID** en vehicles tipus **car** (en la resta és residual). Per altre banda DESTACA que no hi ha vehicles **ELECTRIC** per **van** ni **minivan** les quals son gairebé totalment **GASOLINE**.

Alternativament visualitzem les mateixes dades amb un ballonplot

```
# Pren els ajustos per defecte
theme_set(theme_pubr())
# Preparem les dades
taula = as.data.frame(table(ds$fuelType,ds$vehicle.type))
# Visualitza el Ballonplot
ggballoonplot(taula, fill = "value")+
  scale_fill_viridis_c(option = "C")
```



Zeros i atributs buits

Ara farem el tractament dels valors buits i convertirem les variables discretes a factors.

```
# Estadístiques de valors buits, validem si hi ha valors buits
colSums(is.na(ds))
```

```
##           fuelType             rating   renterTripsTaken        reviewCount
##                  0                501                  0                  0
##      location.city   location.country  location.latitude location.longitude
##                  0                  0                  0                  0
##     location.state           owner.id        rate.daily       vehicle.make
##                  0                  0                  0                  0
##      vehicle.model       vehicle.type       vehicle.year        airportcity
##                  0                  0                  0                  0
```

```
# Estadístiques de valors buits, validem si hi ha valors buits
colSums(ds=="")
```

```
##           fuelType             rating   renterTripsTaken        reviewCount
##                  0                 NA                  0                  0
##      location.city   location.country  location.latitude location.longitude
##                  0                  0                  0                  0
##     location.state           owner.id        rate.daily       vehicle.make
##                  0                  0                  0                  0
##      vehicle.model       vehicle.type       vehicle.year        airportcity
##                  0                  0                  0                  0
```

Es pot veure que queda 1 variable amb valors buits (rating). Anem a calcular doncs quin és el seu valor més freqüent

```
most_freq_rating <- names(which.max(table(ds$rating)))

most_freq_rating
```

```
## [1] "5"
```

Assignem el valor més freqüent als valors buits i comprovem que després del tractament no hi hagi valors buits

```
# Prenem el valor més freqüent per als valors buits
ds$rating[is.na(ds$rating)] = as.numeric(most_freq_rating)

# Visualitzem de nou si hi ha valors buits
colSums(is.na(ds))
```

```
##           fuelType             rating   renterTripsTaken        reviewCount
##                  0                  0                  0                  0
##      location.city   location.country  location.latitude location.longitude
##                  0                  0                  0                  0
##     location.state           owner.id         rate.daily       vehicle.make
##                  0                  0                  0                  0
##      vehicle.model       vehicle.type       vehicle.year        airportcity
##                  0                  0                  0                  0
```

```
colSums(ds=="")
```

```
##           fuelType             rating   renterTripsTaken        reviewCount
##                  0                  0                  0                  0
##      location.city   location.country  location.latitude location.longitude
##                  0                  0                  0                  0
##     location.state           owner.id         rate.daily       vehicle.make
##                  0                  0                  0                  0
##      vehicle.model       vehicle.type       vehicle.year        airportcity
##                  0                  0                  0                  0
```

Valors extrems (Outliers)

Després del tractament dels valors buits examinem de nou els possibles valors extrems.

```
n <- length(ds$rating)

n - boxplot.stats(ds$rating)$n
```

```
## [1] 0
```

Podem veure com el joc de dades ja no presenta valors extrems. I que els valors extrems de la variable *'rating'* es corresponien amb dades buides.

Discretització de variables

Ara examinarem per quines variables tindria sentit realitzar una discretització.

```
# Per a quines variables tindria sentit un procés de discretització?
apply(ds, 2, function(x) length(unique(x)))
```

```
##           fuelType             rating   renterTripsTaken        reviewCount
##                  4                 80                238                203
```

```
##      location.city   location.country  location.latitude location.longitude
##                964                 1               5725               5716
##     location.state          owner.id        rate.daily       vehicle.make
##                 46              3093               294                 51
##      vehicle.model       vehicle.type       vehicle.year        airportcity
##                526                 5                 34                103
```

Per aquelles variables amb pocs valors possibles podem realitzar una discretització. Per això convertim les variables discretes a factors d'*R*.

```
# Convertim les variables discretes a factors
ds[,categorical_features_names] <- lapply(ds[,categorical_features_names] , factor)

# Mostrem el resultat
str(ds)
```

```
## 'data.frame':    5851 obs. of  16 variables:
##  $ fuelType          : Factor w/ 4 levels "DIESEL","ELECTRIC",..: 2 2 4 3 3 3 3 3 3 3 ...
##  $ rating            : num  5 5 4.92 5 5 5 4.42 4.9 5 4.76 ...
##  $ renterTripsTaken  : num  13 2 28 21 3 13 13 12 1 22 ...
##  $ reviewCount       : num  12 1 24 20 1 12 12 10 1 17 ...
##  $ location.city     : Factor w/ 964 levels "Aberdeen Township",..: 801 877 7 7 7 7 7 7 7 7 ...
##  $ location.country  : Factor w/ 1 level "US": 1 1 1 1 1 1 1 1 1 1 ...
##  $ location.latitude : num  47.4 35.1 35.1 35.1 35.2 ...
##  $ location.longitude: num  -122 -106 -107 -107 -107 ...
##  $ location.state    : Factor w/ 46 levels "AK","AL","AR",..: 44 32 32 32 32 32 32 32 32 32 ...
##  $ owner.id          : Factor w/ 3093 levels "5105","12107",..: 2602 3026 2192 2007 1002 1726 918 204
##  $ rate.daily        : num  135 190 35 75 47 58 42 117 102 49 ...
##  $ vehicle.make      : Factor w/ 51 levels "Acura","Alfa Romeo",..: 48 48 49 14 10 32 16 14 14 14 ..
##  $ vehicle.model     : Factor w/ 526 levels "1 Series","124 Convertible 2000",..: 318 318 348 322 41
##  $ vehicle.type      : Factor w/ 5 levels "car","minivan",..: 3 3 1 1 1 3 3 3 1 3 ...
##  $ vehicle.year      : num  2019 2018 2012 2018 2010 ...
##  $ airportcity       : Factor w/ 103 levels "Albuquerque",..: 1 1 1 1 1 1 1 1 1 1 ...
```

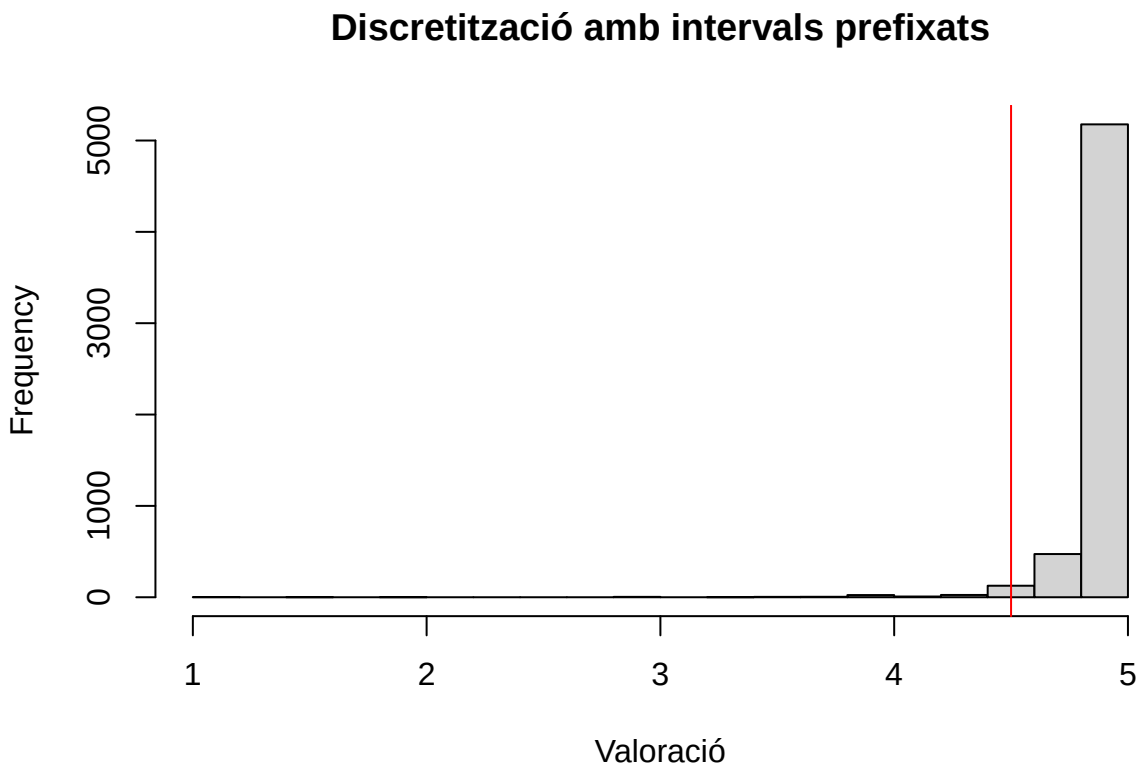
Transformació d'atributs

A continuació realitzarem algunes transformacions sobre alguns atributs, amb la finalitat de generar diferents punts de vista de les dades.

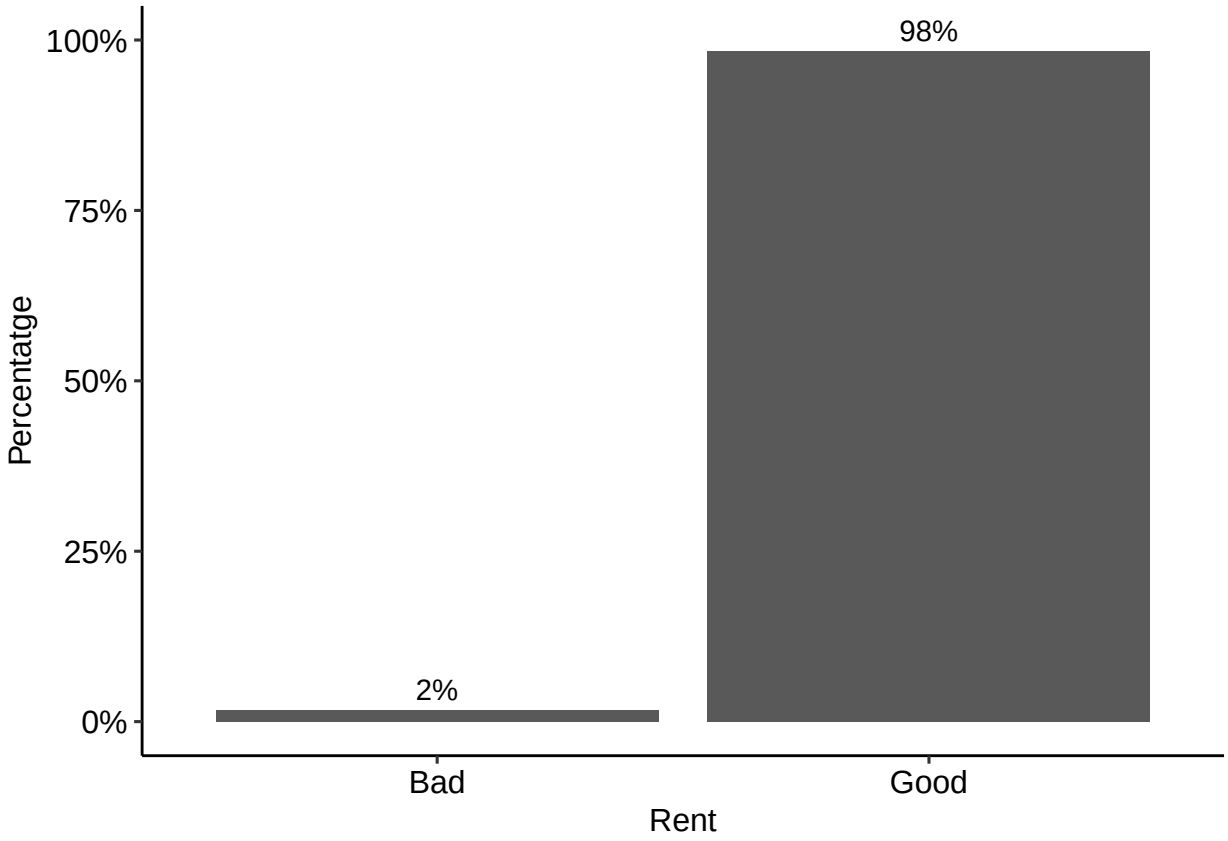
```
# Discretització amb intevals prefixats, de les valoracions
table(discretize(ds$rating, method = "fixed", c(0, 4.5, Inf), labels = c('Bad', 'Good')))
```

```
##
##  Bad Good
##   97 5754
```

```
hist(ds$rating, breaks = 20, main = "Discretització amb intervals prefixats", xlab = "Valoració")
cuts_rating <- discretize(ds$rating, method = "fixed",c(0, 4.5, Inf), onlycuts = TRUE)
abline(v = cuts_rating, col = "red")
```




```
# Discretització amb intevals prefixats, de les valoracions
ds['rating.discret'] <- discretize(ds$rating, method = "fixed", breaks = c(0, 4.5, Inf), labels = c('Ba

ds$rating.discret = as.factor(ds$rating.discret)

# Calculem gràfic de barres
ggplot(data=ds, aes(x = rating.discret)) + geom_bar(aes(y = (..count..)/sum(..count..))) + geom_text(ae
```




```
# Correcció de les marques de vehicles, diferències tipogràfiques
ds$vehicle.make[ds$vehicle.make == "Alfa-romeo"] = "Alfa Romeo"
ds$vehicle.make[ds$vehicle.make == "Mercedes-benz"] = "Mercedes-Benz"
ds$vehicle.make[ds$vehicle.make == "INFINITI"] = "Infinitti"

ds$vehicle.make <- droplevels.factor(ds$vehicle.make)
```

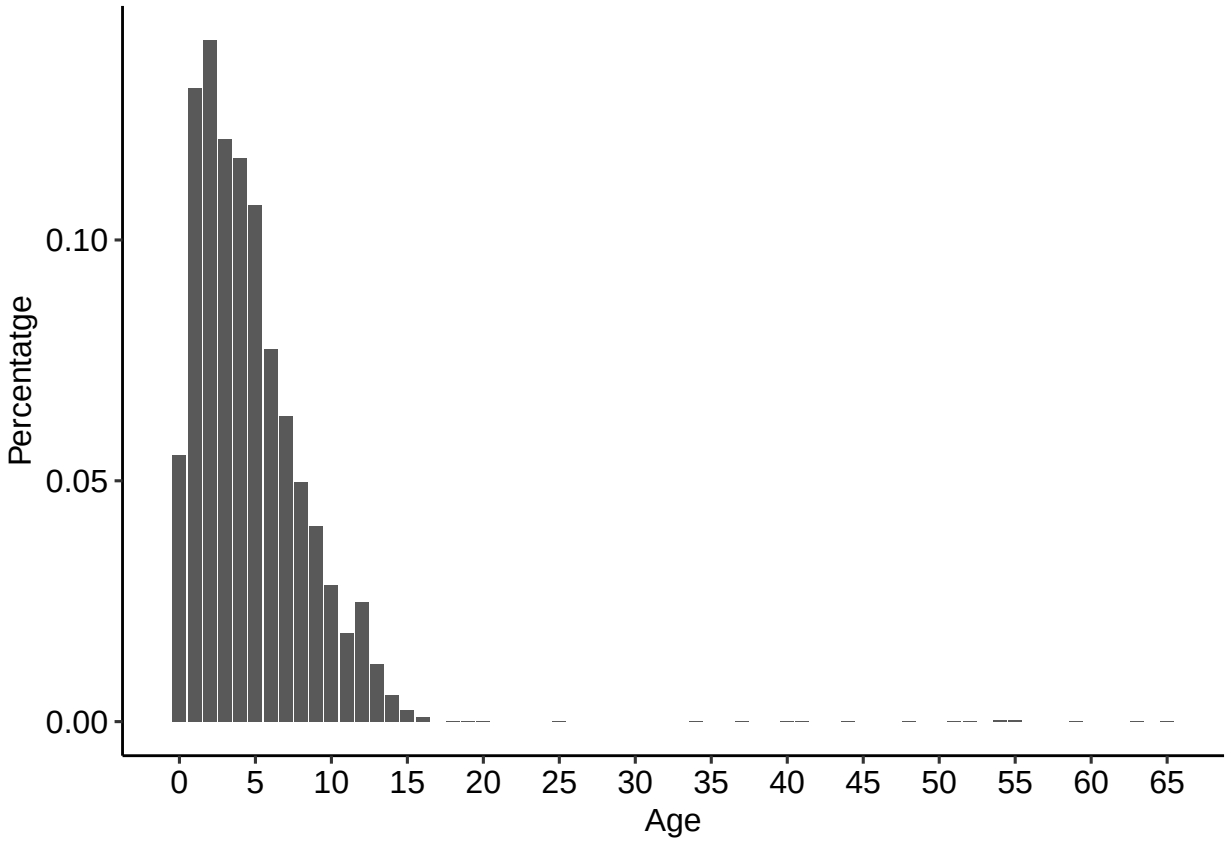
Creació de nous indicadors

Un altre dels passos interessant en l'anàlisi d'un joc de dades és la generació de nous atributs a partir dels existents.

Indicador: age

Creem un nou indicador o atribut on emmagatzemem l'antiguitat del vehicle.

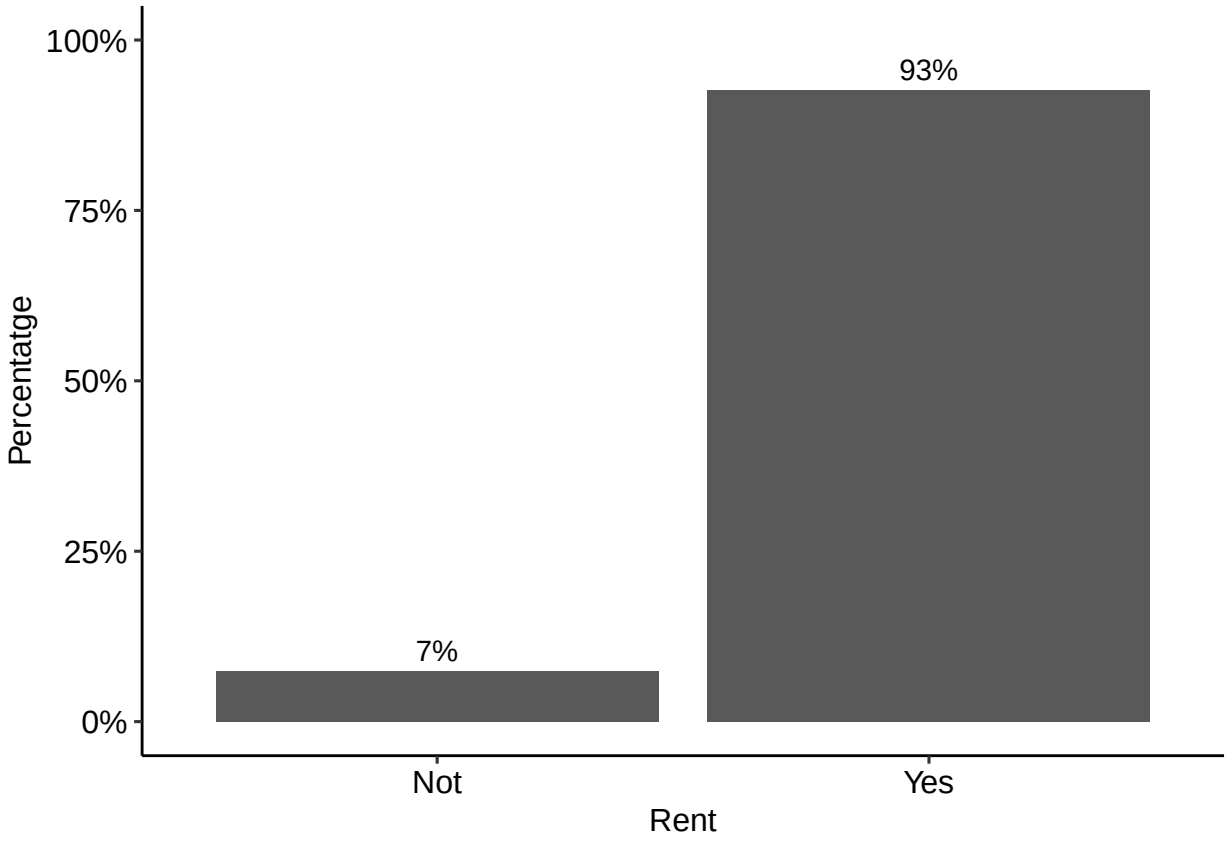
```
# Antiguitat del vehicle
ds['age'] <- as.integer(format(Sys.Date(), "%Y")) - ds$vehicle.year

# Calculem gràfic de barres de la nova variable
ggplot(data=ds, aes(x = age)) + geom_bar(aes(y = (..count..)/sum(..count..))) + scale_x_continuous(breal
```



Indicador: rent

Creem un altre indicador o atribut nou on emmagatzemem si el vehicle va ser llogat o no.

```
# Vehicle llogat/no llogat
ds['rent'] <- ifelse(ds['renterTripsTaken'] > 0, 1, 0)
ds$rent <- as.factor(ds$rent)
levels(ds$rent) <- c("Not","Yes")

# Calculem gràfic de barres de la variable objectiu
ggplot(data=ds, aes(x = rent)) + geom_bar(aes(y = (..count..)/sum(..count..))) + geom_text(aes(y = ((..
```



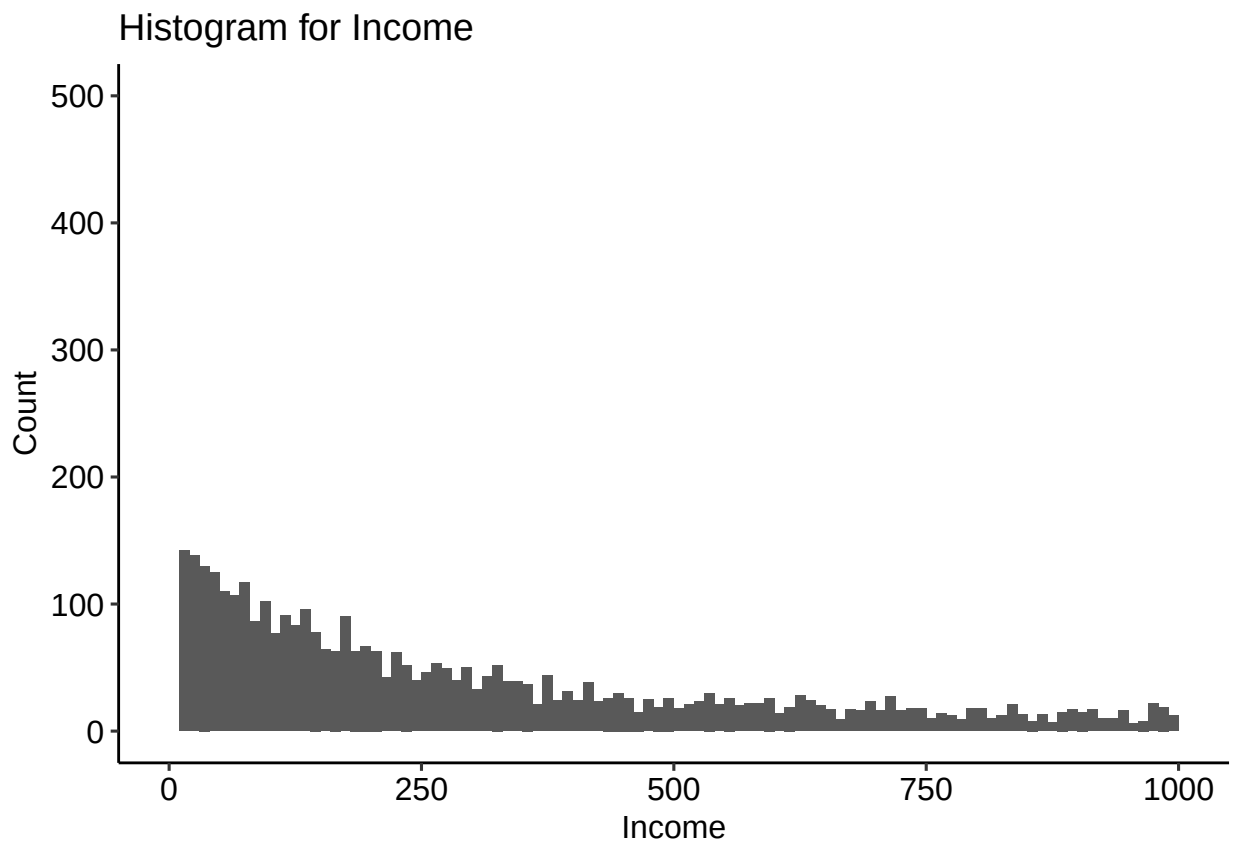
Indicador: income

Creem un altre indicador o atribut nou on emmagatzemem els ingressos anuals que va aportar el vehicle.

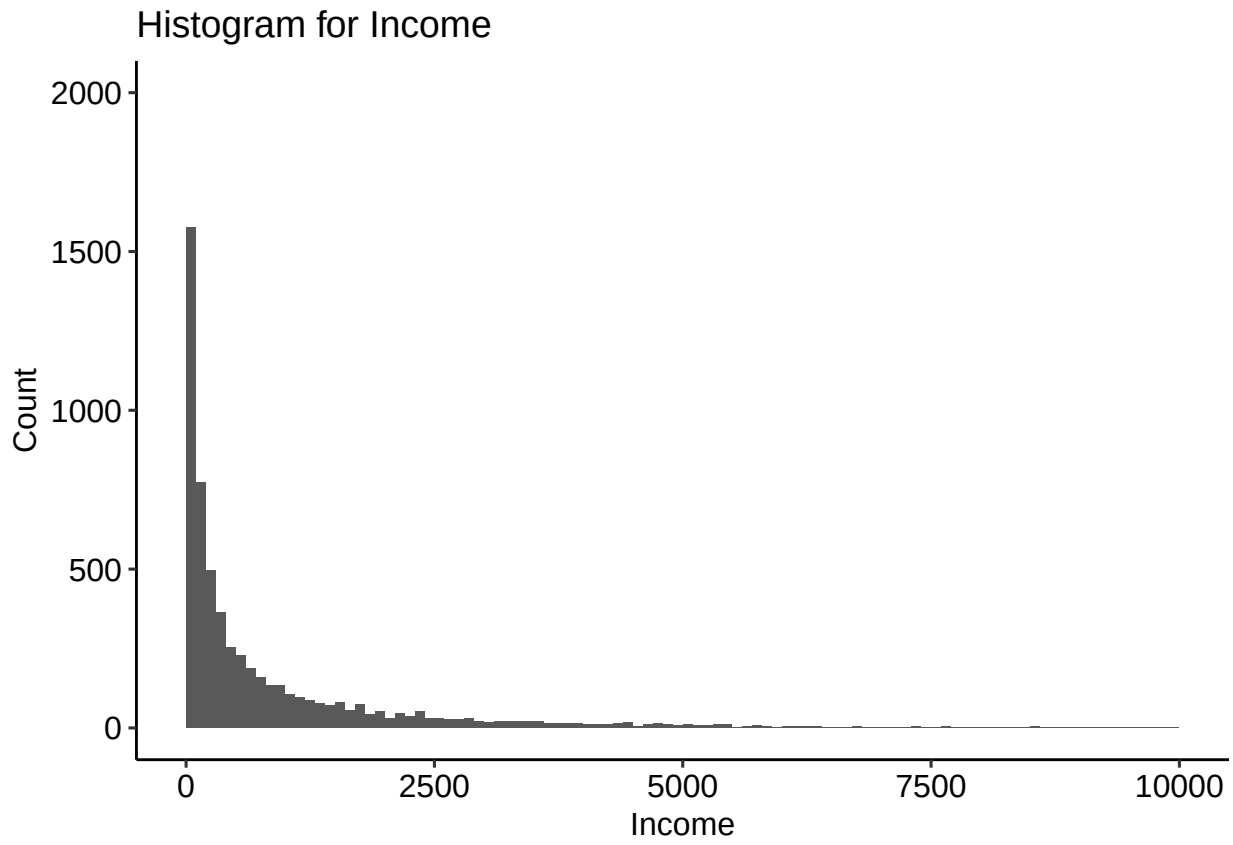
Per crear aquesta variable **income** (ingressos anuals mitjans), s'assumeix el següent:

- **renterTripsTaken** és equivalent al nombre de dies que el cotxe ha estat llogat i que per tant es poden aproximar els ingressos totals aportats pel vehicle com a resultat del producte entre **renterTripsTaken** i **rate.daily**.
- s'assumeix que **age** representa tots els anys de vida del cotxe, per tant si es vol calcular els ingressos anuals cal repartir-los entre age

```
# Ingressos anuals vehicle

# En primer lloc establim que la edat mínima amb la que treballem és d'un any
# per tal d'evitar divisors de "0"
edat = ds$age
edat[edat == 0] = 1

ds['income'] <- ds$rate.daily * ds$renterTripsTaken / edat

# Calculem gràfic de barres de la nova variable
ggplot(data=ds, aes(x=ds$income)) +
  geom_histogram(breaks=seq(0, 1000, by=10),
                 alpha = 1) +
  labs(title="Histogram for Income", x="Income", y="Count") +
  xlim(c(0,1000)) +
  ylim(c(0,500))
```



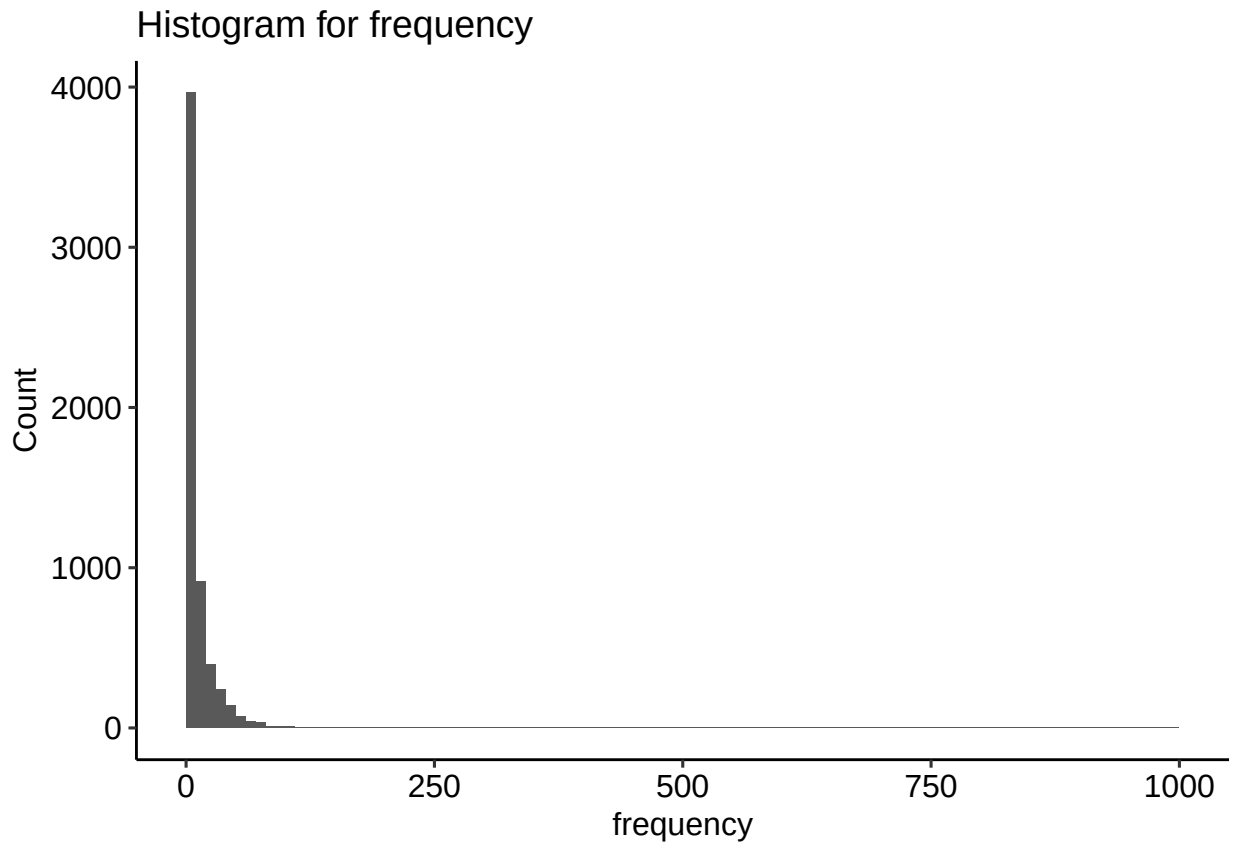

```
ggplot(data=ds, aes(x=ds$income)) +
  geom_histogram(breaks=seq(0, 10000, by=100),
                 alpha = 1) +
  labs(title="Histogram for Income", x="Income", y="Count") +
  xlim(c(0,10000)) +
  ylim(c(0,2000))
```



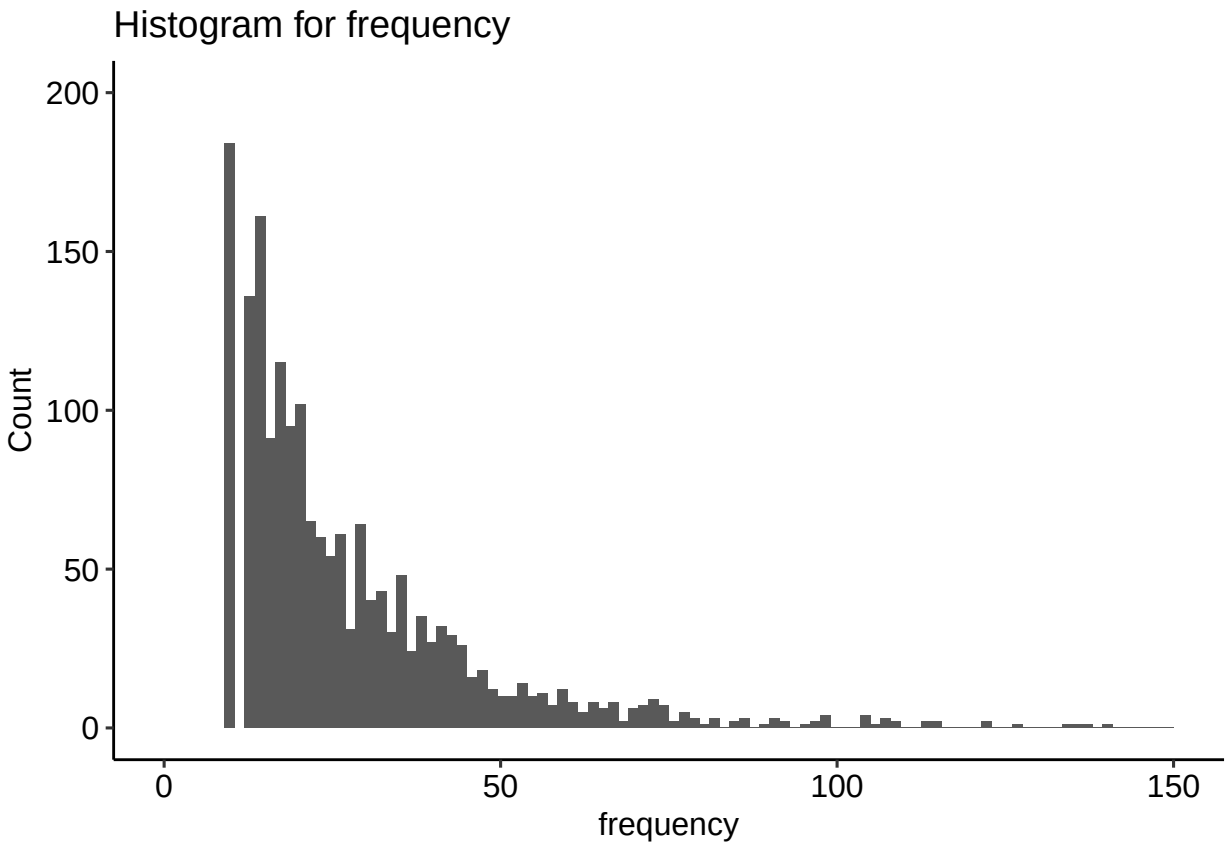
Indicador: frequency

Creem un altre indicador o atribut nou on emmagatzemem la freqüència amb la que es va llogar el vehicle (**frequency**).

```
# Freqüència d'us del vehicle (vegades a l'any)

# En primer lloc establim que la edat mínima amb la que treballem és d'un any
# per tal d'evitar divisors de "0"
edat = ds$age
edat[edat == 0] = 1

ds['frequency'] <- ds$renterTripsTaken / edat

# Calculem gràfic de barres de la nova variable
ggplot(data=ds, aes(x=ds$frequency)) +
  geom_histogram(breaks=seq(0, 1000, by=10),
                 alpha = 1) +
  labs(title="Histogram for frequency", x="frequency", y="Count")
```




```
ggplot(data=ds, aes(x=ds$frequency)) +
  geom_histogram(breaks=seq(0, 150, by=1.5),
                 alpha = 1) +
  labs(title="Histogram for frequency", x="frequency", y="Count") +
  xlim(c(0,150)) +
  ylim(c(0,200))
```



Exportació de les dades preprocesades

Una vegada que hem realitzat sobre el conjunt de dades inicial els procediments de preprocessament integració, validació i neteja, procedim a guardar aquest nou joc de dades en un fitxer anomenat CarRentalDataV1_Clean.csv.

```
# Exportació de les dades preprocessades a un fitxer .CSV
write.csv(ds, '../data/CarRentalDataV1_Clean.csv')
```

Anàlisi de les dades

La gran majoria dels atributs presents en el conjunt de dades es corresponen amb característiques dels diversos vehicles de lloguer o de la seva ubicació, per tant serà convenient tenir-los en consideració durant la realització de les anàlisis. No obstant això, podem prescindir de tres atributs, l'estat (location.country; que sempre és *US*) i les dues coordenades geogràfiques (location.latitude i location.longitude) ja que la informació que ens aporten aquestes variables ja es troba implicita en la resta d'atributs de localització i, per tant, són menys rellevants a l' hora de resoldre el nostre problema.

Selecció dels grups de dades a analitzar

```
# Eliminem location.country i les coordenades geogràfiques
ds <- subset(ds, select=-c(location.latitude,location.longitude,
location.country))

categorical_features_names = c('fuelType', 'location.city', 'location.state', 'owner.id', 'vehicle.make
```

```
numeric_features_names = c('rating', 'renterTripsTaken', 'reviewCount', 'rate.daily', 'age', 'income',
```

Comprobació de la normalitat

Per a la comprovació que els valors que prenen les nostres variables quantitatives provenen d'una població distribuïda normalment, utilitzarem la prova de normalitat d'Anderson-Darling. Per això, es comprova que per cada prova s'obté un *p-valor* superior el nivell de significació prefixat = 0. 05 (Nivell de confiança del 95%). Si això es compleix, llavors es considera que la variable en qüestió segueix una distribució normal.

Test de normalitat

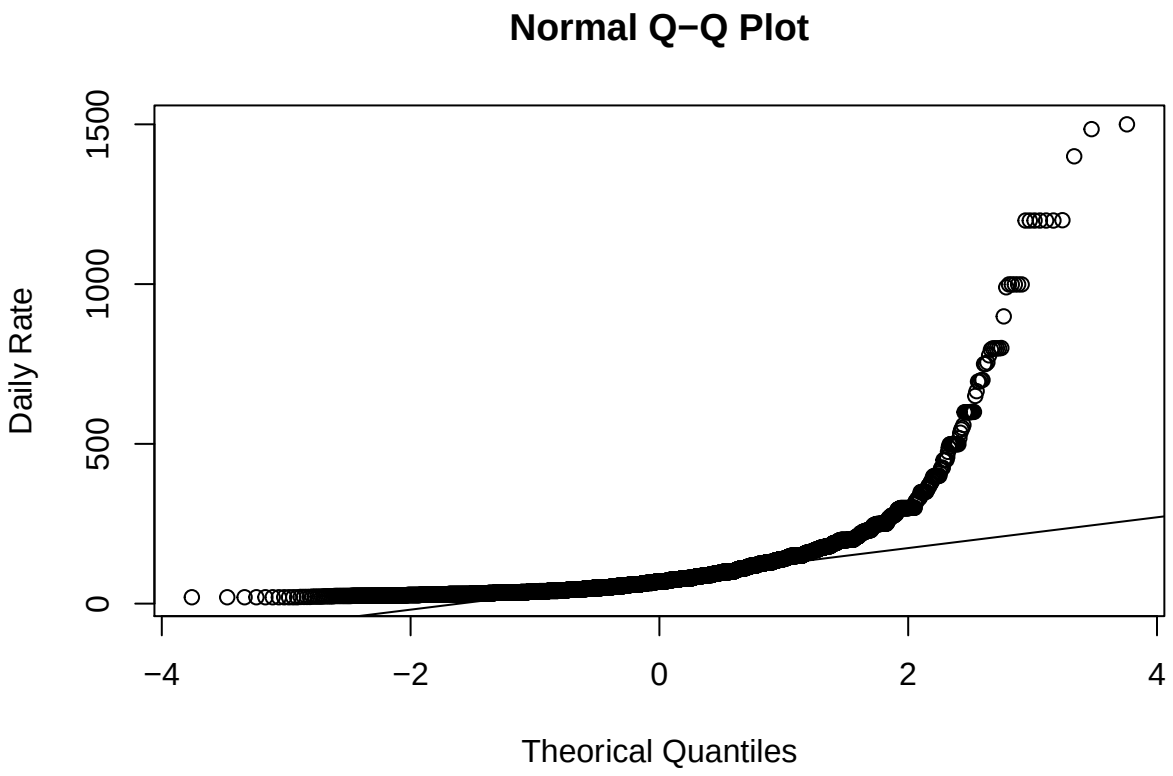
```
alpha = 0.05
col.names = colnames(ds)

for (i in 1:ncol(ds)) {
  if (i == 1) cat("Variables que no presenten una distribució normal:\n")
    if (is.integer(ds[,i]) | is.numeric(ds[,i])) {
      p_val = ad.test(ds[,i])$p.value
      if (p_val < alpha) {
        cat(col.names[i])
        # Format output
        if (i < ncol(ds) - 1) cat(", ")
        if (i %% 3 == 0) cat("\n")
      }
    }
}
```

```
## Variables que no presenten una distribució normal:
## rating, renterTripsTaken,
## reviewCount, rate.daily, vehicle.year,
## age,
## incomefrequency
```

A partir del resultat anterior podem veure com que cap dels atributs quantitaius presenta una distribució normal. Tot i així ens centrarem ara un moment el la variable depenent o variable objectiu en el nostre estudi, el preu diari del vehicle.

Si observem els valors mínim, mitjà i màxim veiem clarament que no és una variable amb distribució normal. Però en la majoria de models numèrics d'aprenentatge automàtic, necessitarem que les variables segueixin la distribució normal. Per això podem emprar visualitzacions de les dades per analitzar la normalitat. Concretament, el gràfic Q-Q, on la Q denota quantil, és un tipus de visualització que s'utilitza per a diagnosticar la desviació de les dades de la mostra en relació amb una població normal.

```
qqnorm(ds$rate.daily, ylab="Daily Rate", xlab="Theorical Quantiles", main="Normal Q-Q Plot")
qqline(ds$rate.daily)
```

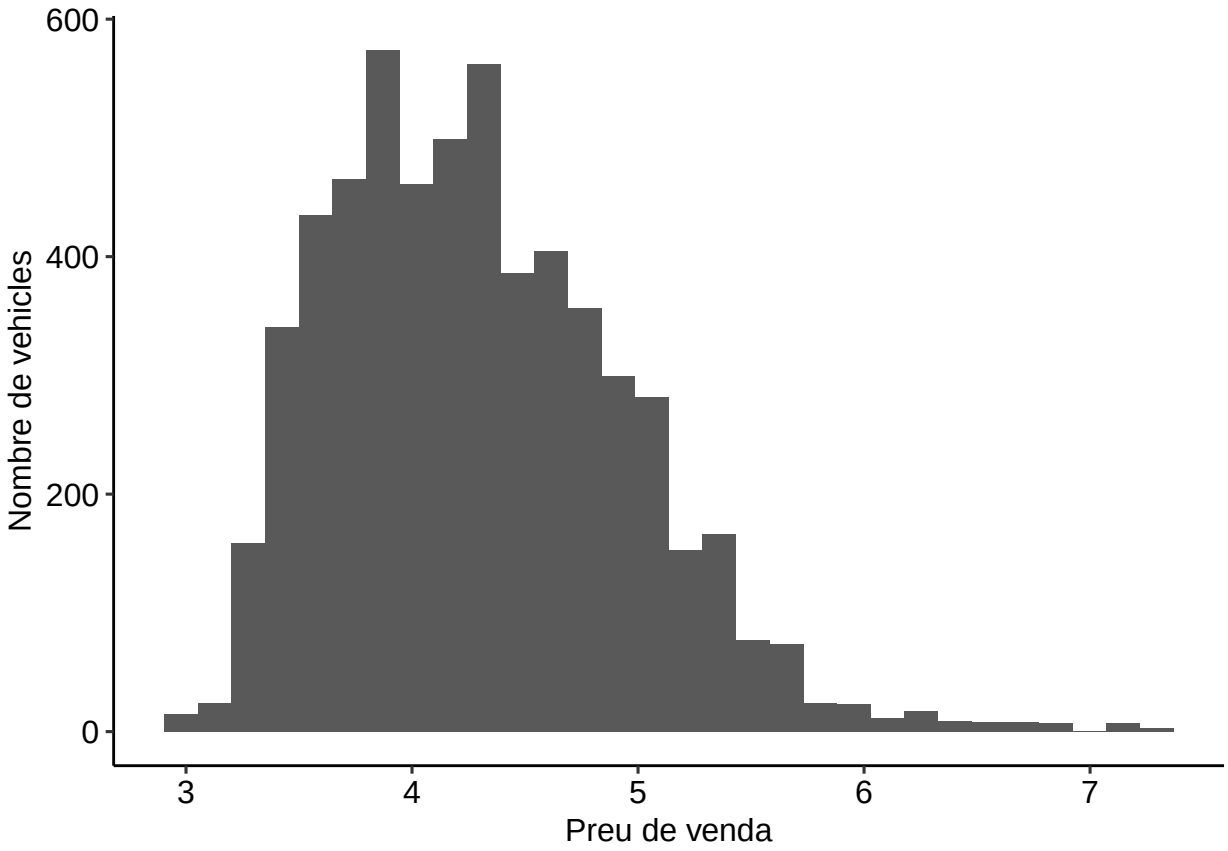


Quan tenim una variable que no és normal, una de les transformacions clàssiques que pot funcionar és aplicar el logaritme a la variable.

Alternativament podem usar la funció *BoxCox* la qual ens ajuda a seleccionar la transformació óptima de la variable per a que s'acabi distribuint de forma "normal".

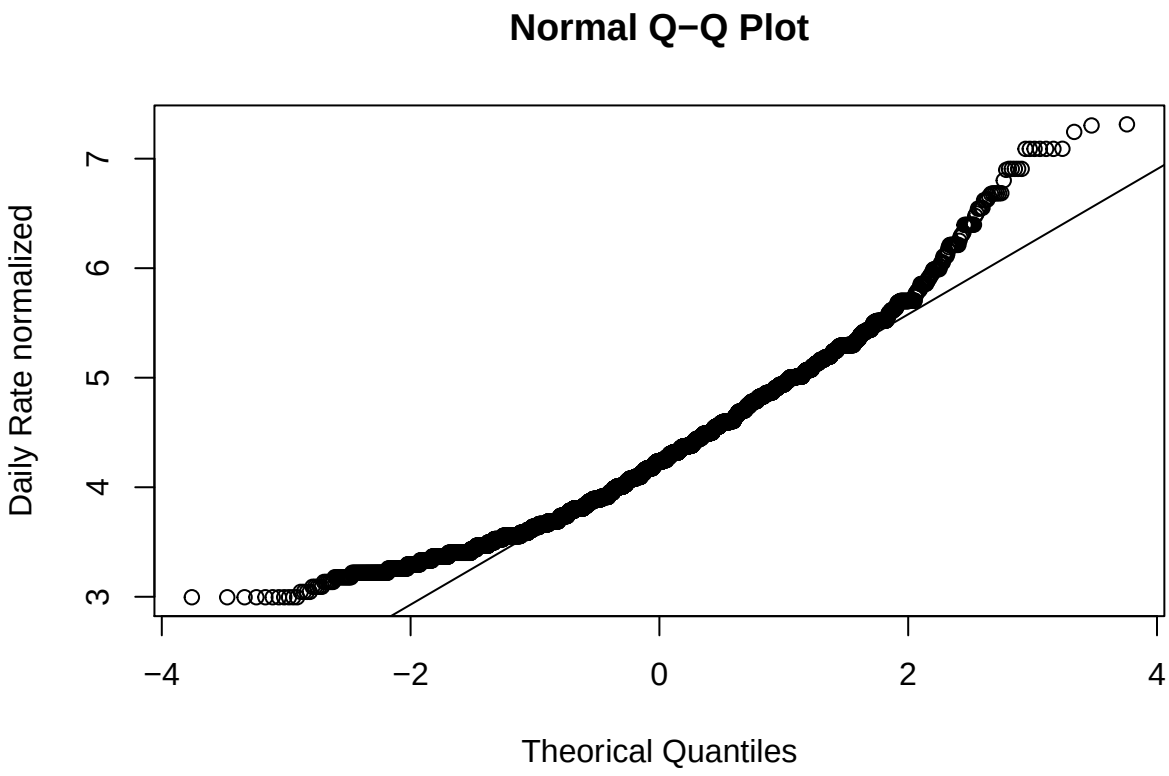
Normalització de Daily Rate (Preu)

```
ds$rate.daily <- log(ds$rate.daily)

ggplot(ds, aes(x = rate.daily)) +  geom_histogram() +  ylab("Nombre de vehicles") + xlab("Preu de venda
```



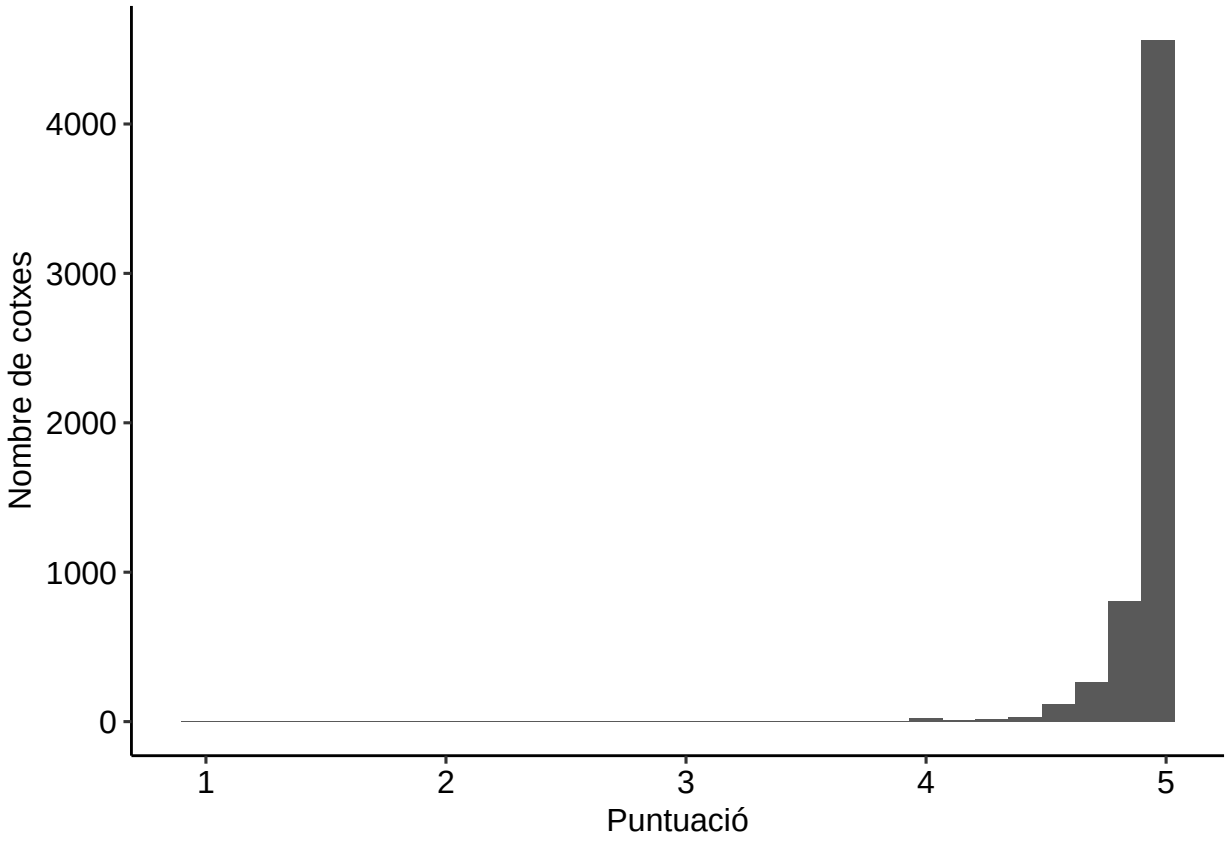
Veiem com ara sí que té forma normal. Ho comprovem amb el Q-Q plot per confirmar-ho.

```
qqnorm(ds$rate.daily, ylab="Daily Rate normalized", xlab="Theorical Quantiles", main="Normal Q-Q Plot")
qqline(ds$rate.daily)
```

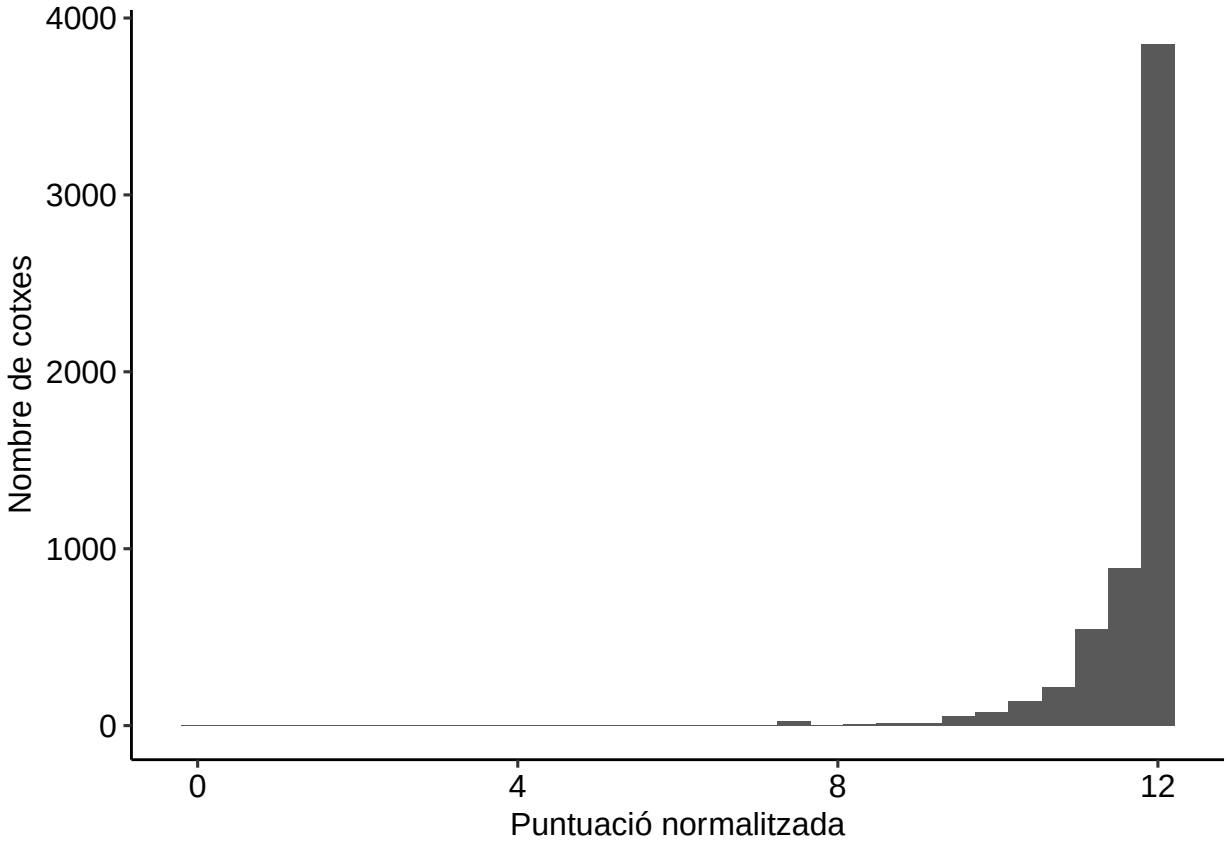


Normalització de Rating (Puntuació)

```
lambda_optima <- BoxCoxLambda(ds$rating)
ds$rating_norm <- BoxCox(ds$rating, lambda = lambda_optima)

ggplot(ds, aes(x = rating)) +  geom_histogram() +  ylab("Nombre de cotxes") + xlab("Puntuació")
```



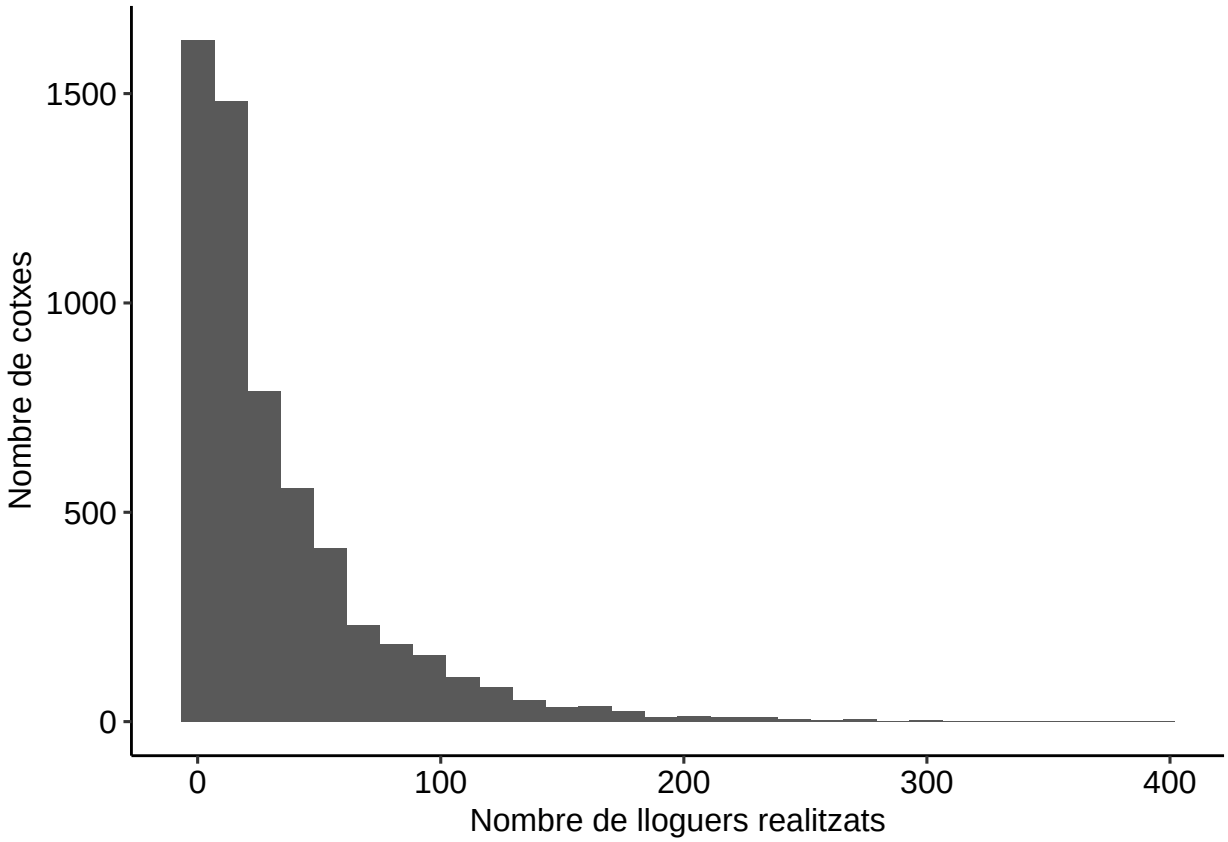

```
ggplot(ds, aes(x = rating_norm)) +  geom_histogram() +  ylab("Nombre de cotxes") + xlab("Puntuació norma
```



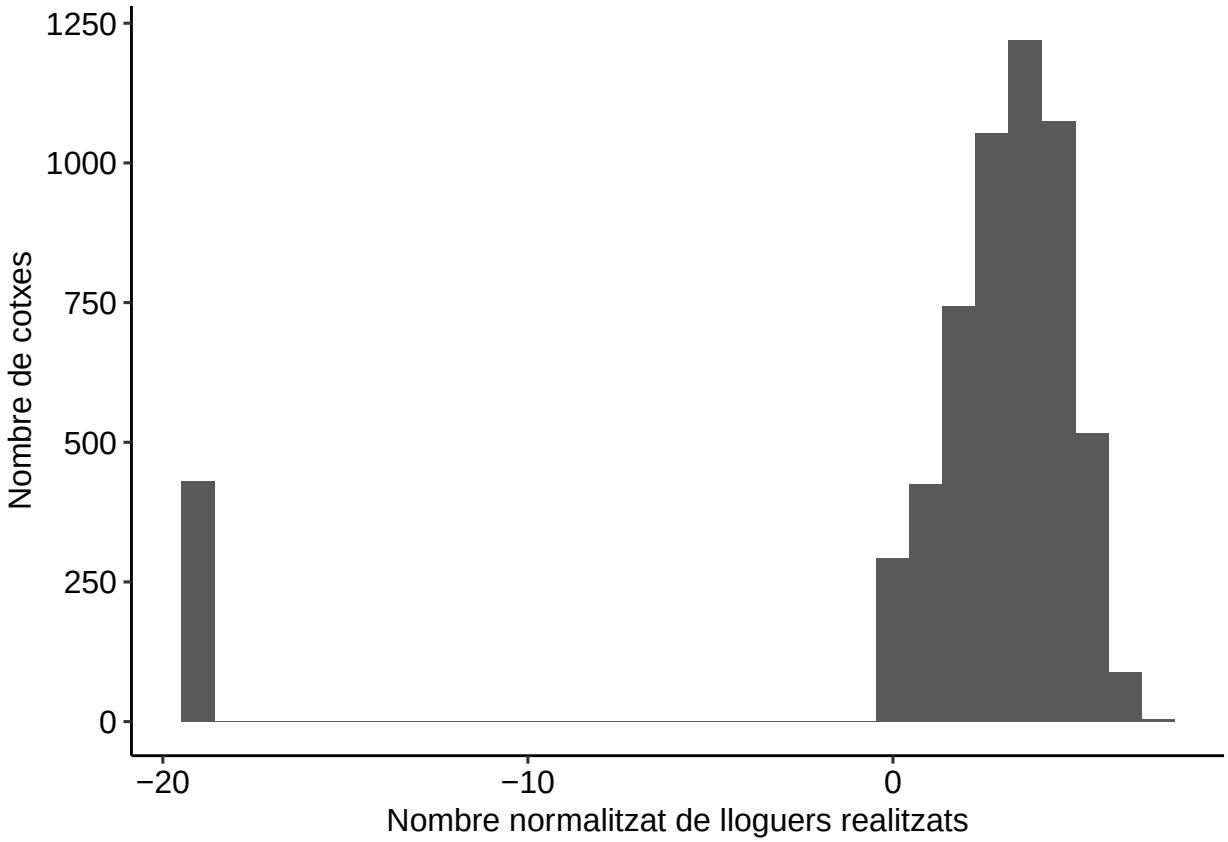
Veiem de forma evident que, tot i amb la conversió, no s'obté forma de distribució normal.

Normalització de Renter Trips Taken (Nombre de lloguers realitzats)

```
lambda_optima <- BoxCoxLambda(ds$renterTripsTaken)
ds$renterTripsTaken_norm <- BoxCox(ds$renterTripsTaken, lambda = lambda_optima)

ggplot(ds, aes(x = renterTripsTaken)) +  geom_histogram() +  ylab("Nombre de cotxes") + xlab("Nombre de
```



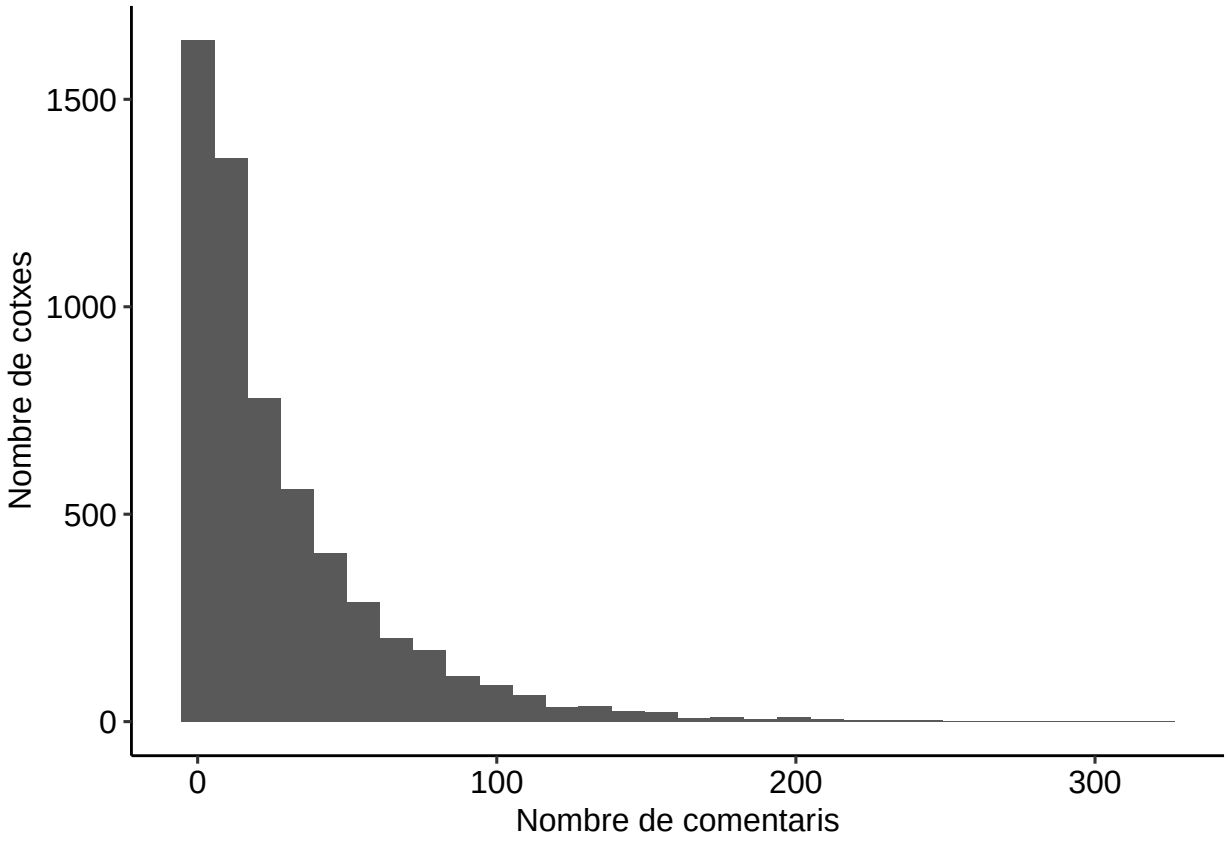

```
ggplot(ds, aes(x = renterTripsTaken_norm)) +  geom_histogram() +  ylab("Nombre de cotxes") + xlab("Nombr
```



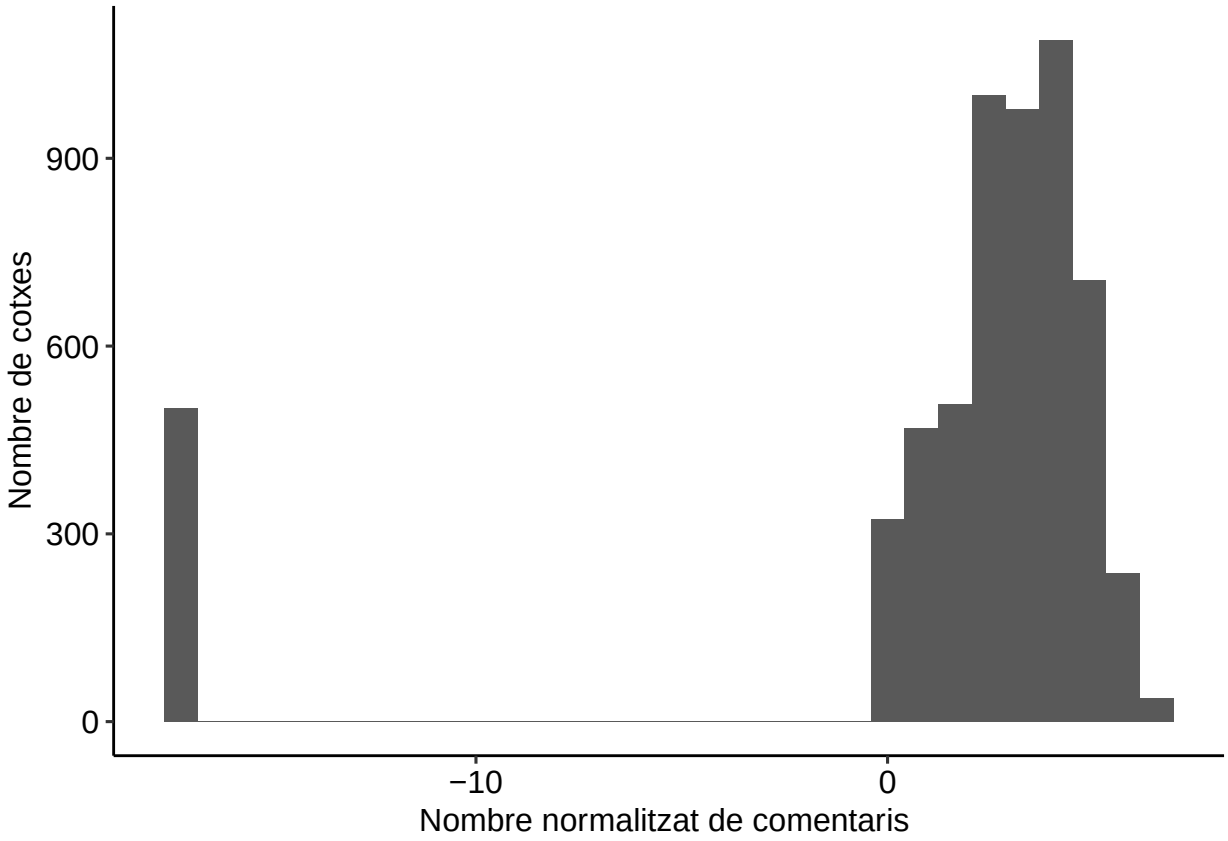
Veiem que, tot i amb la conversió, no s'obté forma de distribució normal ja que queden un grup significatiu de valors aïllats.

Normalització de reviewCount (Nombre de comentaris)

```
lambda_optima <- BoxCoxLambda(ds$reviewCount)
ds$reviewCount_norm <- BoxCox(ds$reviewCount, lambda = lambda_optima)

ggplot(ds, aes(x = reviewCount)) +  geom_histogram() +  ylab("Nombre de cotxes") + xlab("Nombre de comen
```



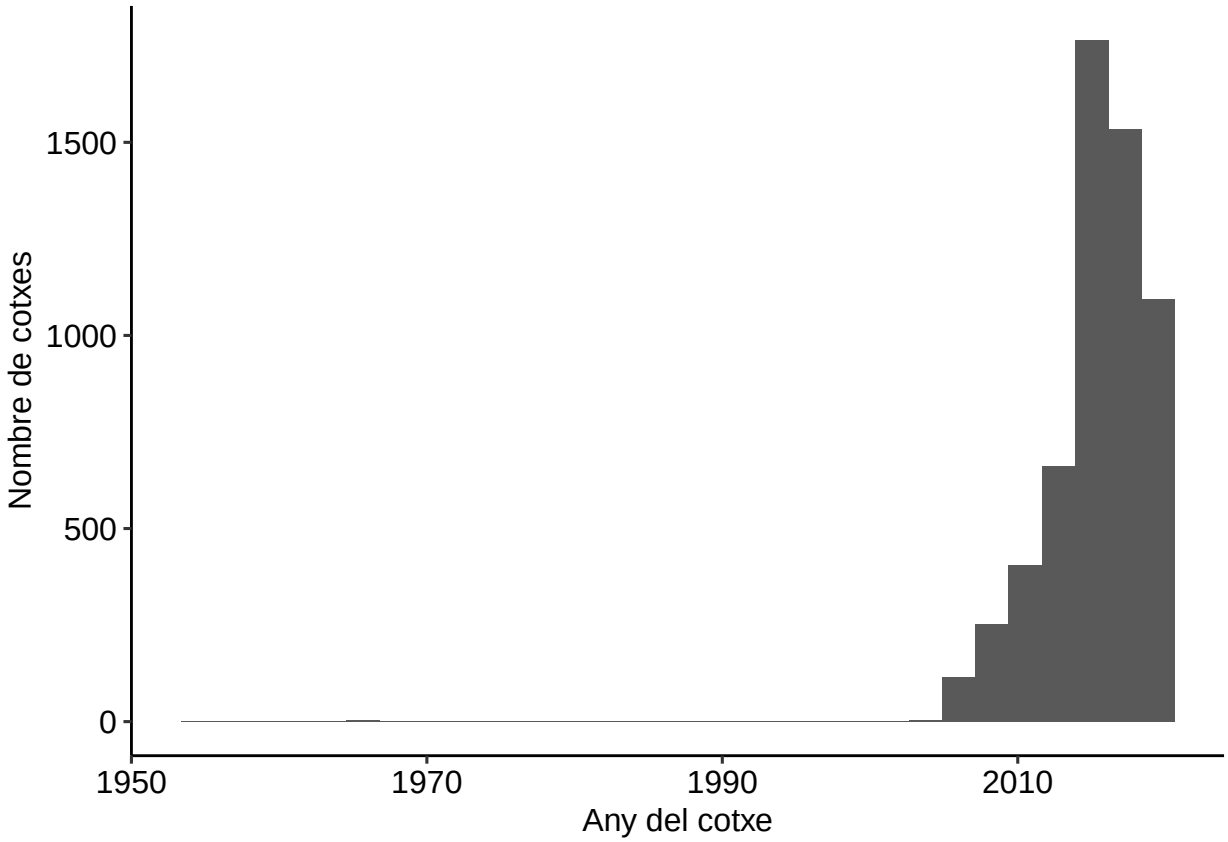

```
ggplot(ds, aes(x = reviewCount_norm)) +  geom_histogram() +  ylab("Nombre de cotxes") + xlab("Nombre no
```



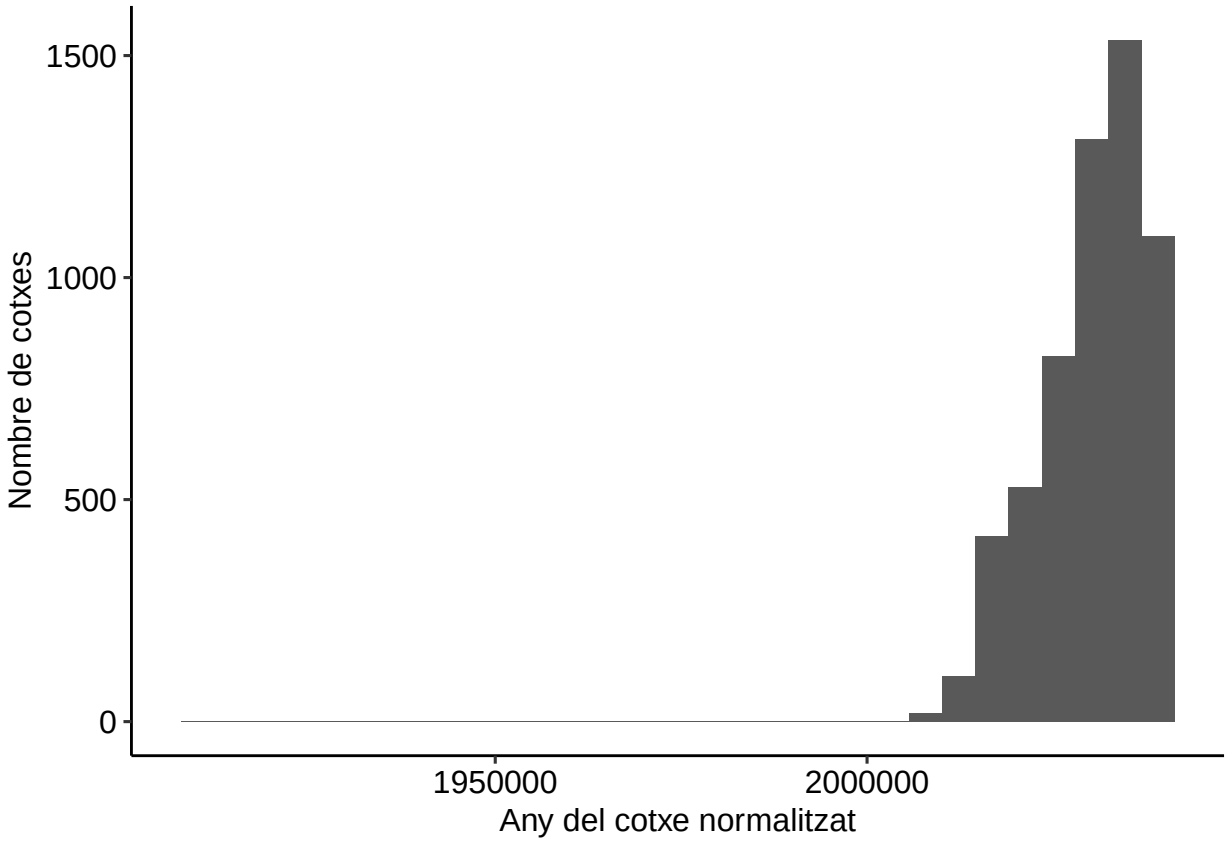
Veiem que, tot i amb la conversió, no s'obté forma de distribució normal ja que queden un grup significatiu de valors aïllats.

Normalització de vehicle.year (Any del cotxe)

```
lambda_optima <- BoxCoxLambda(ds$vehicle.year)
ds$vehicle.year_norm <- BoxCox(ds$vehicle.year, lambda = lambda_optima)

ggplot(ds, aes(x = vehicle.year)) +  geom_histogram() +  ylab("Nombre de cotxes") + xlab("Any del cotxe
```



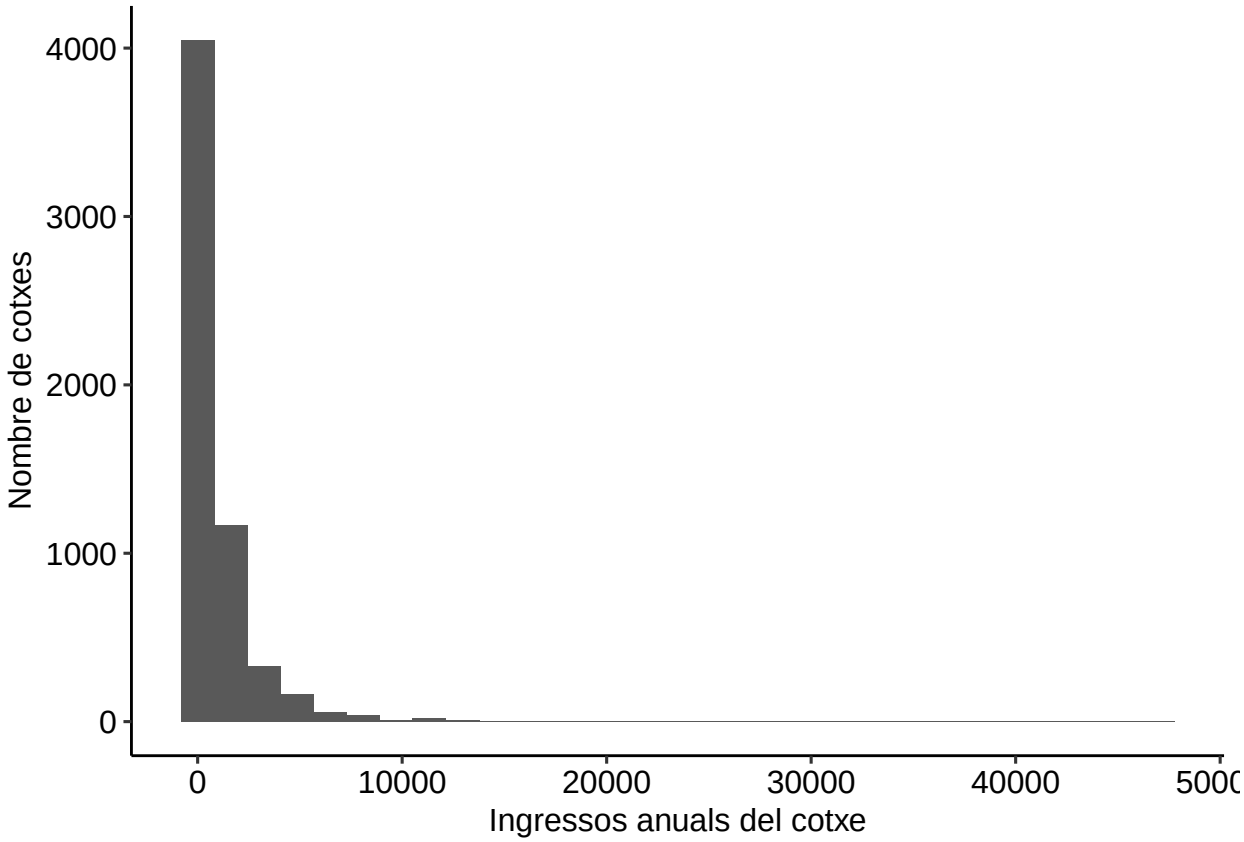

```
ggplot(ds, aes(x = vehicle.year_norm)) +  geom_histogram() +  ylab("Nombre de cotxes") + xlab("Any del
```



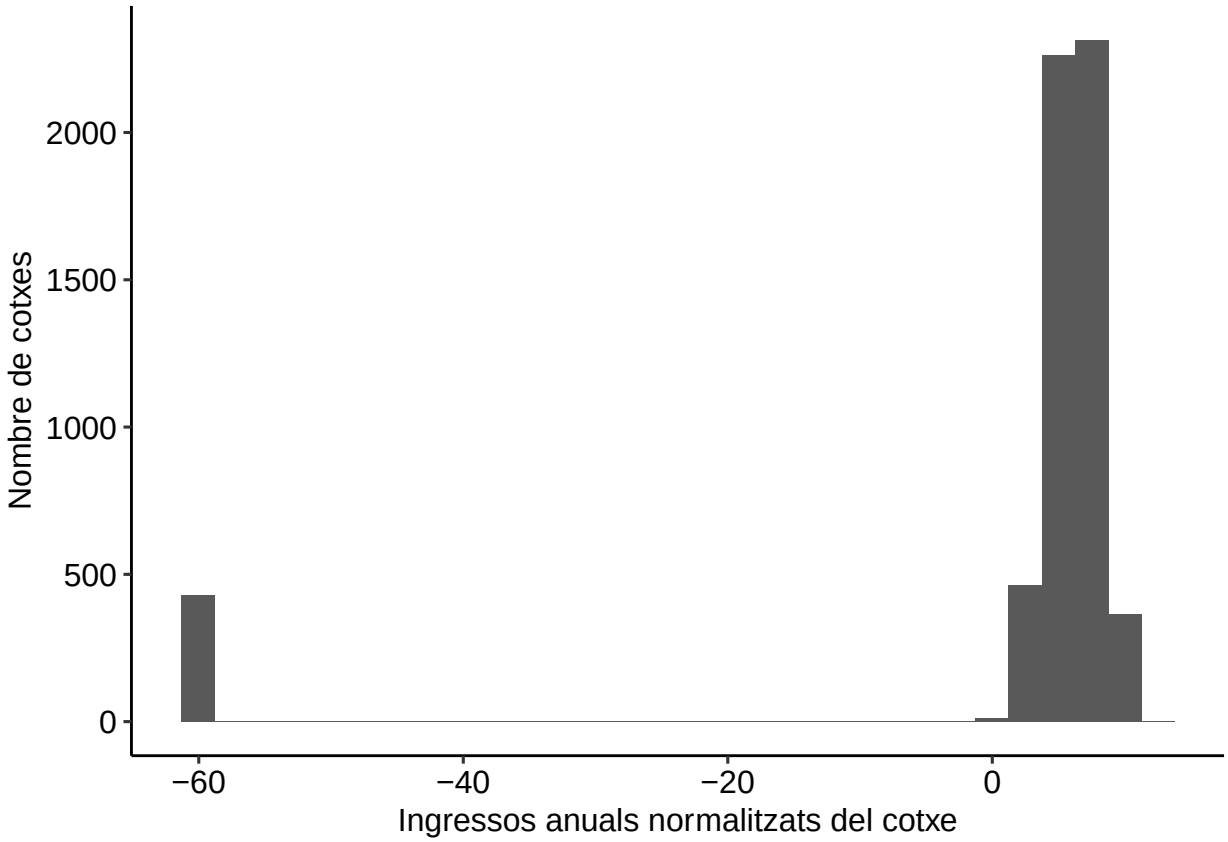
Veiem que, tot i amb la conversió, no s'obté forma de distribució normal.

Normalització de Income (Ingressos anuals del cotxe)

```
lambda_optima <- BoxCoxLambda(ds$income)
ds$income_norm <- BoxCox(ds$income, lambda = lambda_optima)

ggplot(ds, aes(x = income)) +  geom_histogram() +  ylab("Nombre de cotxes") + xlab("Ingressos anuals del
```



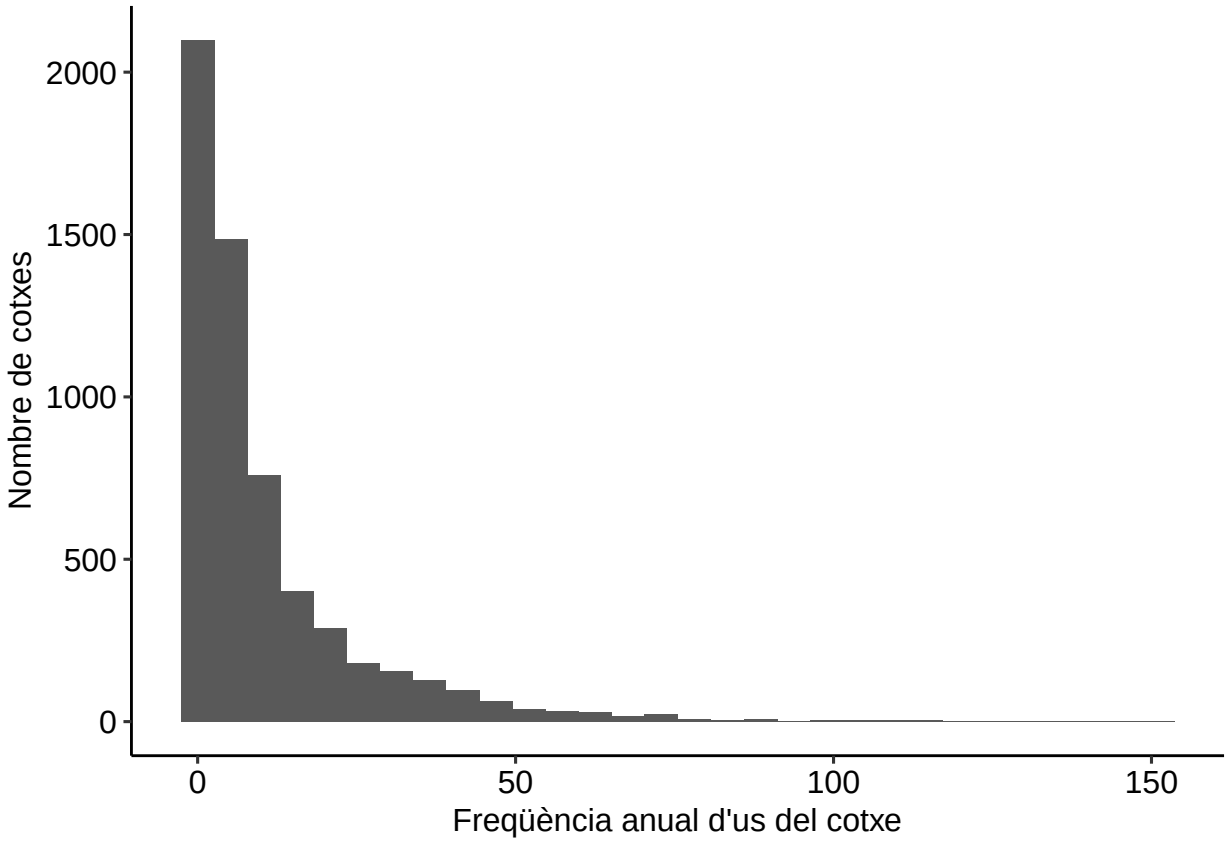

```
ggplot(ds, aes(x = income_norm)) +  geom_histogram() +  ylab("Nombre de cotxes") + xlab("Ingressos anual
```



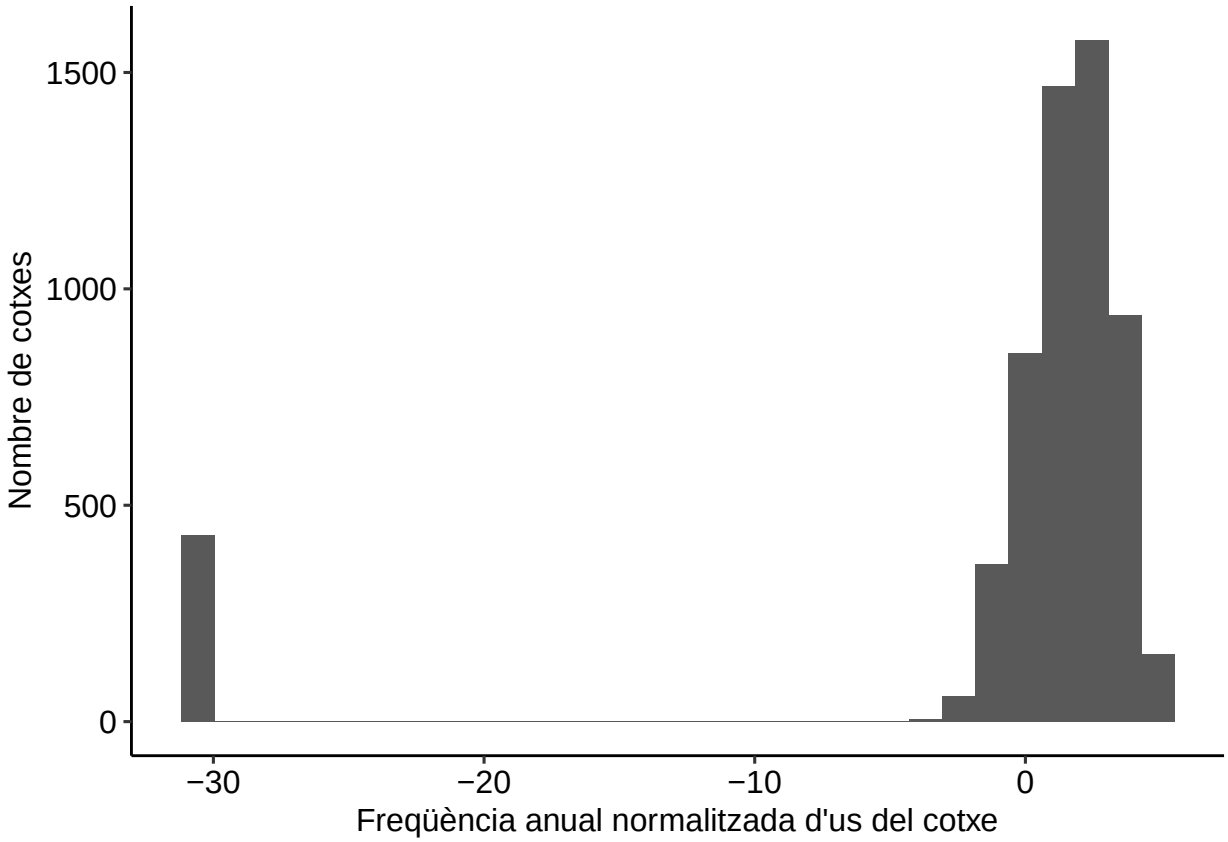
Veiem que, tot i amb la conversió, no s'obté forma de distribució normal ja que queden un grup significatiu de valors aïllats.

Normalització de Frecuency (freqüència anual d'us del cotxe)

```
lambda_optima <- BoxCoxLambda(ds$frequency)
ds$frequency_norm <- BoxCox(ds$frequency, lambda = lambda_optima)

ggplot(ds, aes(x = frequency)) +  geom_histogram() +  ylab("Nombre de cotxes") + xlab("Freqüència anual
```




```
ggplot(ds, aes(x = frequency_norm)) +  geom_histogram() +  ylab("Nombre de cotxes") + xlab("Freqüència a
```



Veiem que, tot i amb la conversió, no s'obté forma de distribució normal ja que queden un grup significatiu de valors aïllats.

Neteja de noves variables creades que no han tingut èxit en la normalització

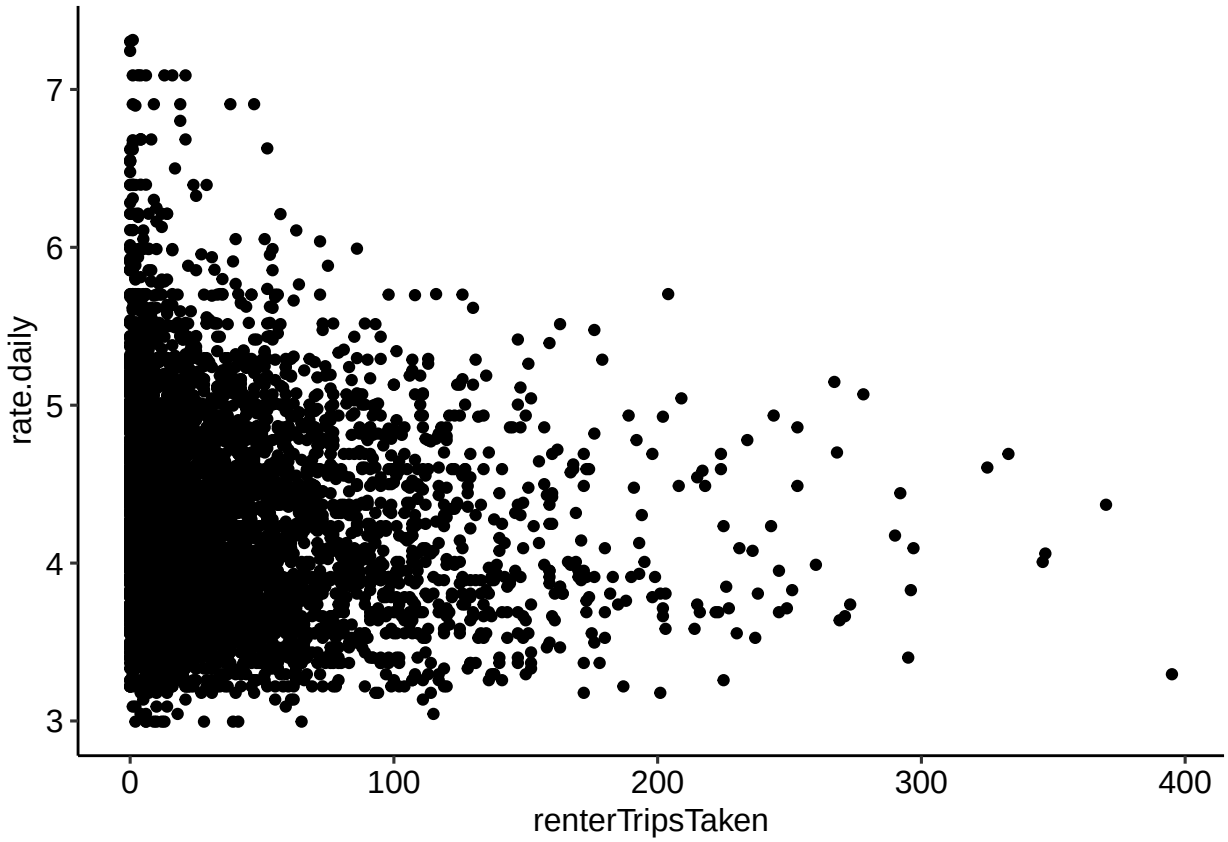
```
# Eliminem location.country i les coordenades geogràfiques
ds <- subset(ds, select=-c(rating_norm, renterTripsTaken_norm, reviewCount_norm, vehicle.year_norm, inc

numeric_features_names = c('rating', 'renterTripsTaken', 'reviewCount', 'rate.daily', 'age', 'income',
```

Anàlisi multivariant

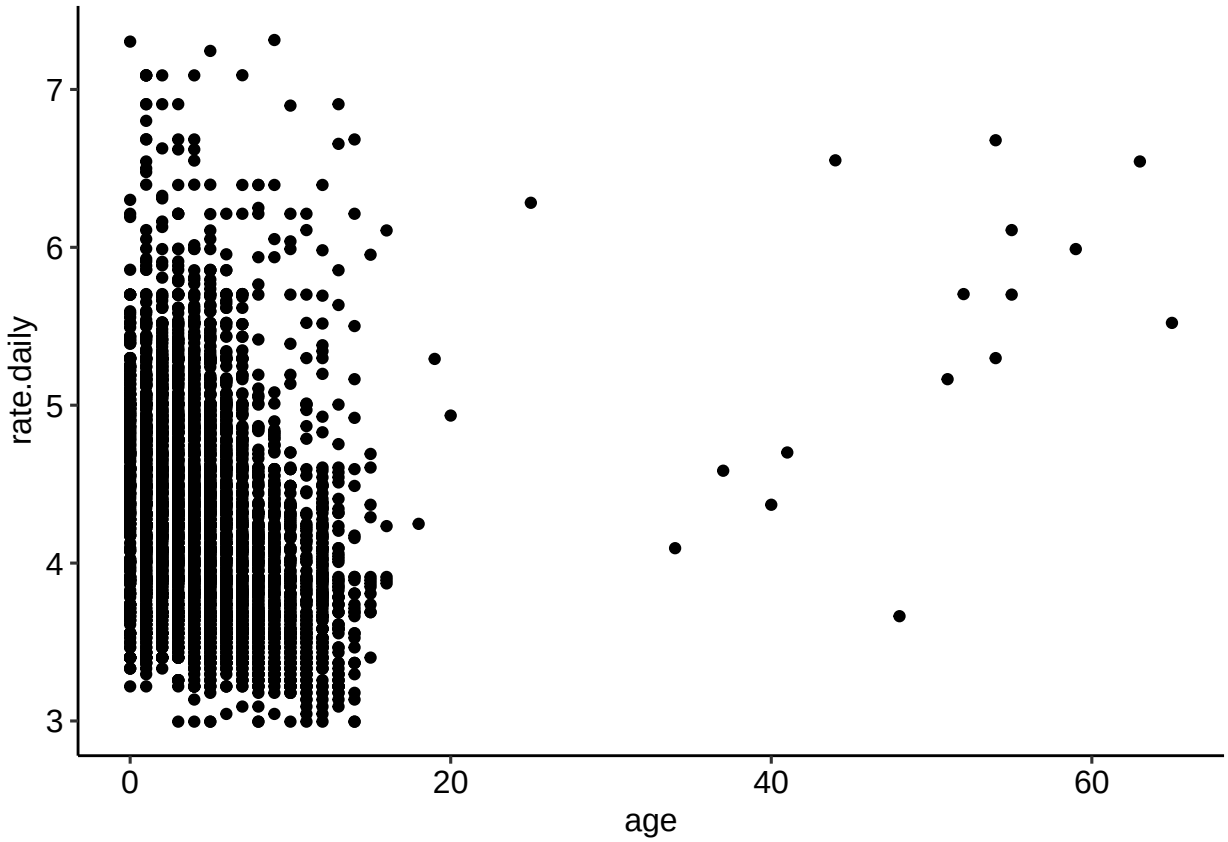
Per analitzar un conjunt de dades necessitem tenir en compte més d'una variable alhora. L'anàlisi bivariant permet identificar les relacions entre dues variables, i fins quina manera una pot predir l'altra.

En aquest cas podem veure quina és la relació entre el preu diari del lloguer i el nombre de vegades que s'ha llogat el vehicle amb un scatter plot o diagrama de punts.

```
# Relació entre el preu diari del lloguer i el nombre de vegades que s'ha llogat el vehicle
ggplot(ds, aes(x=renterTripsTaken, y=rate.daily)) + geom_point()
```

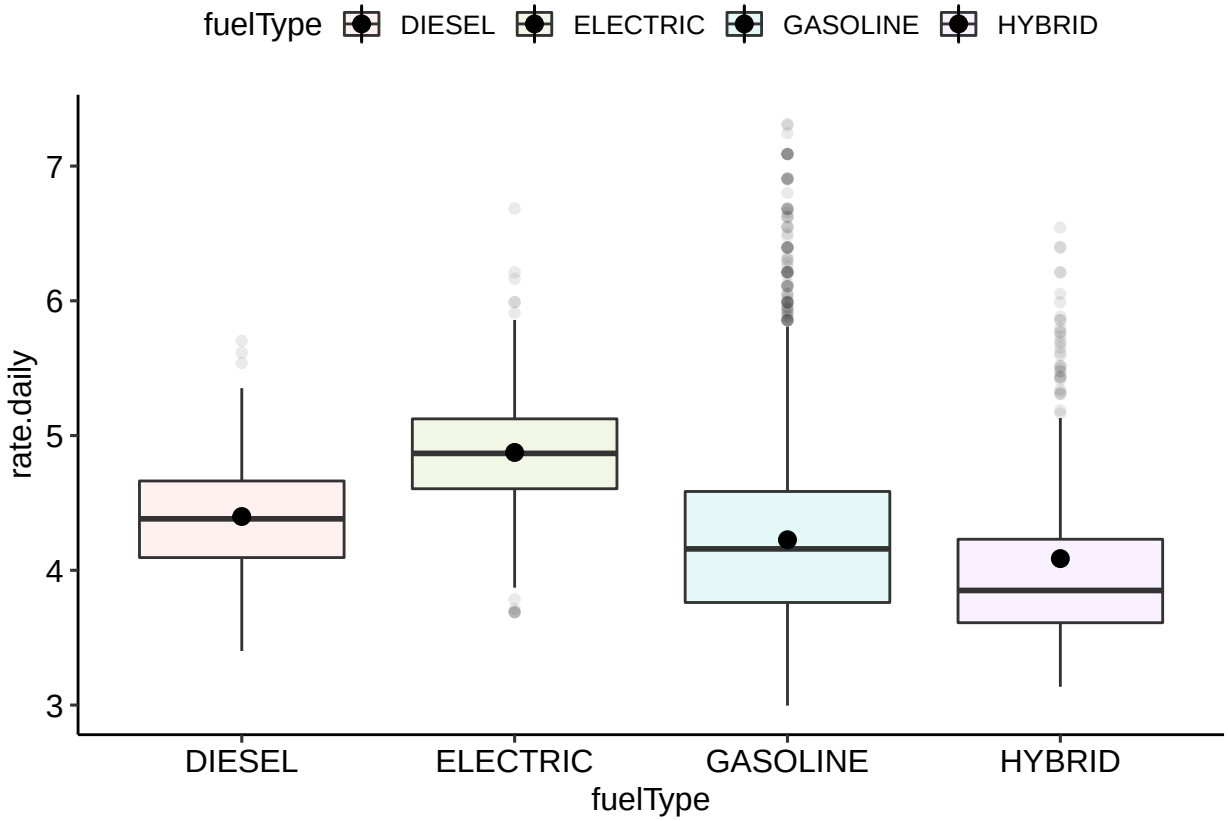



```
# Relació entre el preu diari del lloguer i el nombre de vegades que s'ha llogat el vehicle
ggplot(ds, aes(x=age, y=rate.daily)) + geom_point()
```

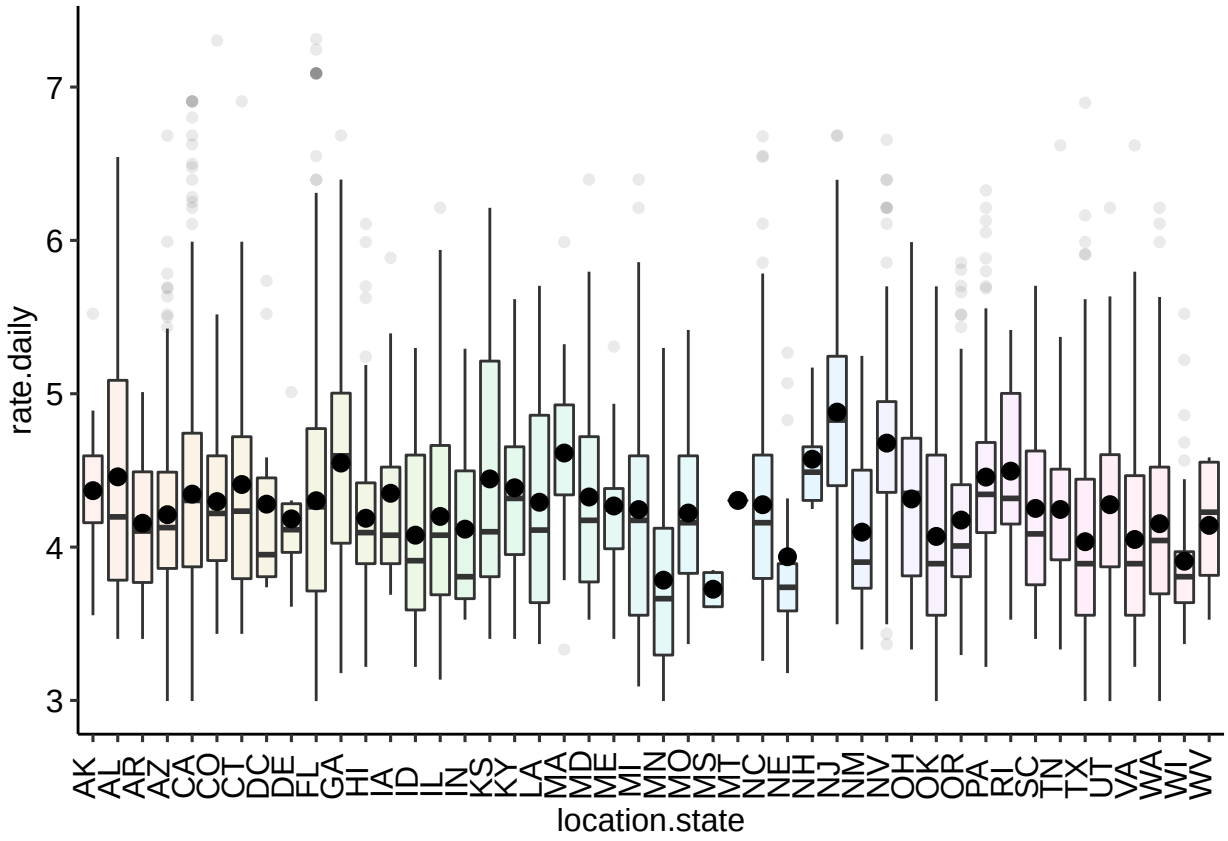


En el cas d'una variable categòrica com el tipus de combustible o la marca del vehicle, podem visualitzar la relació amb un boxplot o diagrama de caixes.

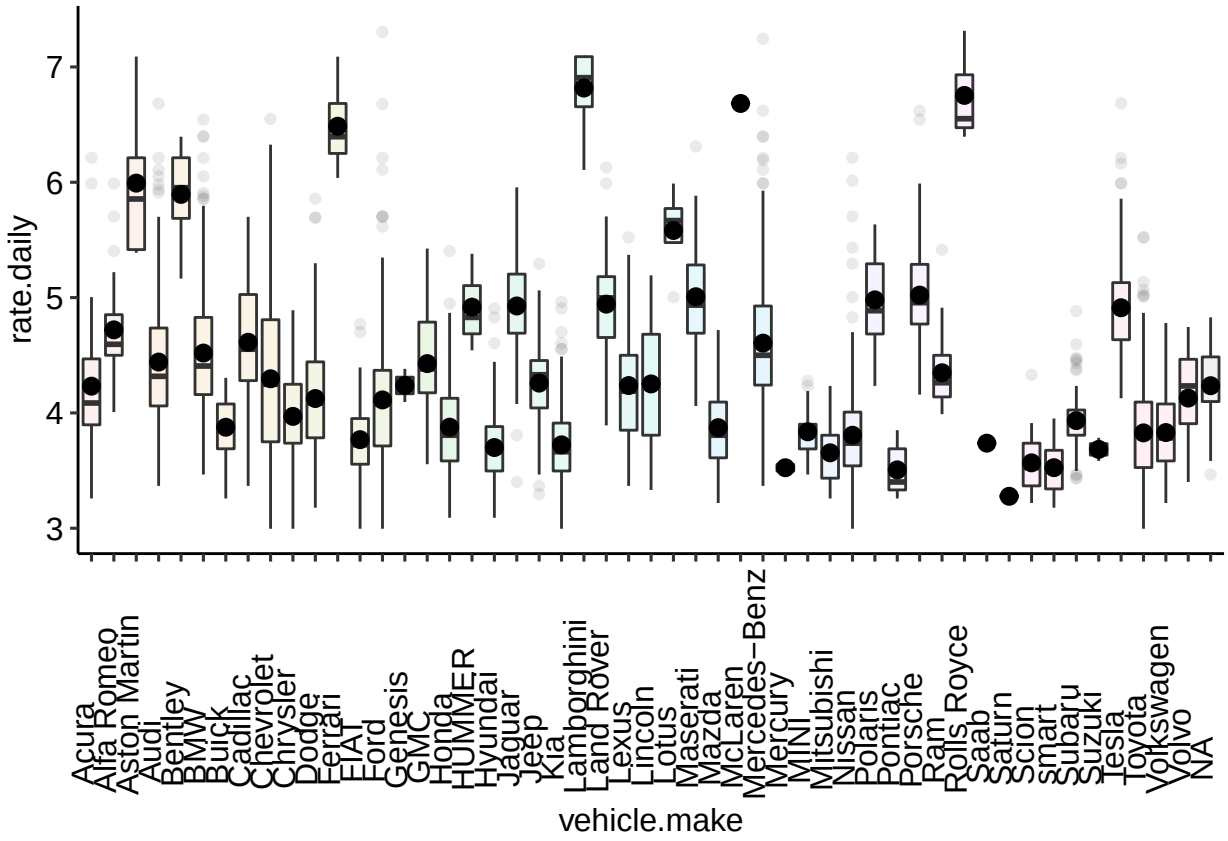
```
# Relació entre el preu diari del lloguer i el tipus de combustible
ggplot(ds, aes(x=fuelType, y=rate.daily, fill=fuelType)) + geom_boxplot(alpha=0.1) + stat_summary(fun.y=
```



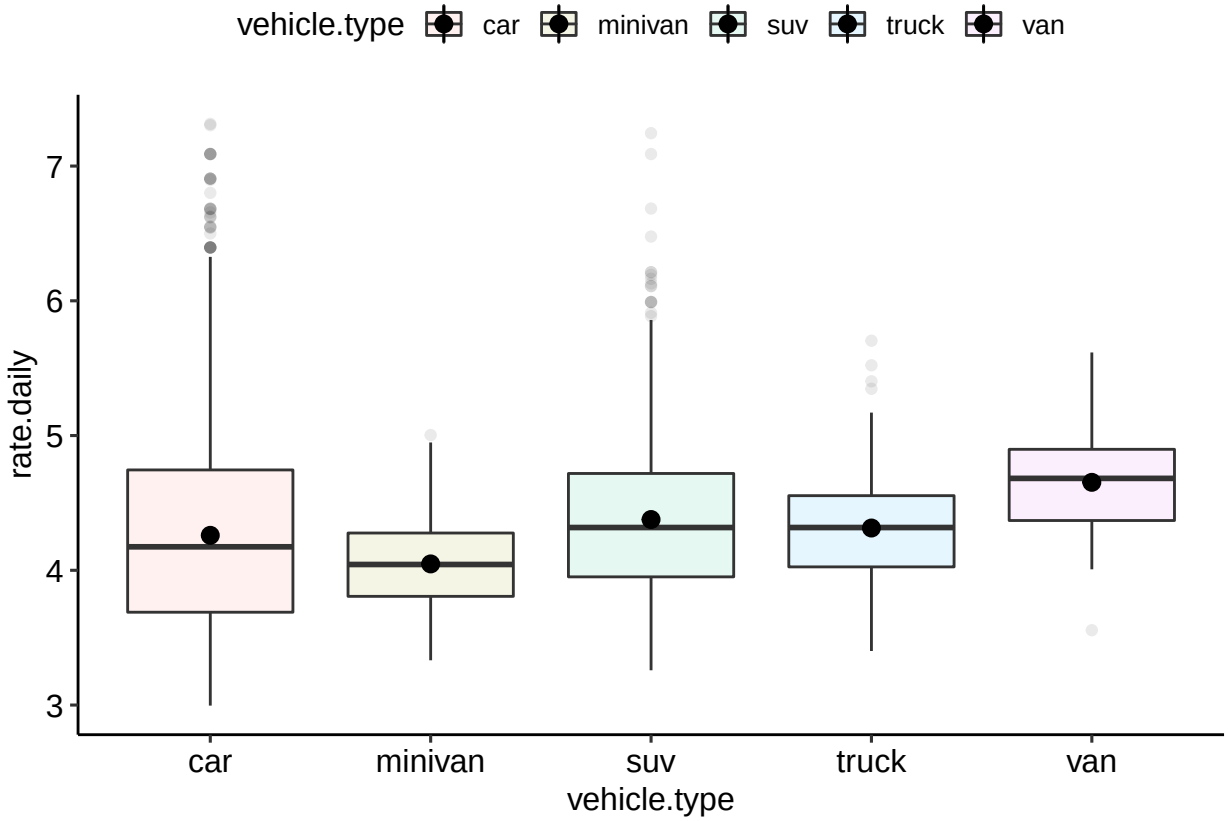

```
# Relació entre el preu diari del lloguer i l'estat on es localitza el vehicle
ggplot(ds, aes(x=location.state, y=rate.daily, fill=location.state)) + geom_boxplot(alpha=0.1) + stat_su
```




```
# Relació entre el preu diari del lloguer i la marca del vehicle
ggplot(ds, aes(x=vehicle.make, y=rate.daily, fill=vehicle.make)) + geom_boxplot(alpha=0.1) + stat_summa
```



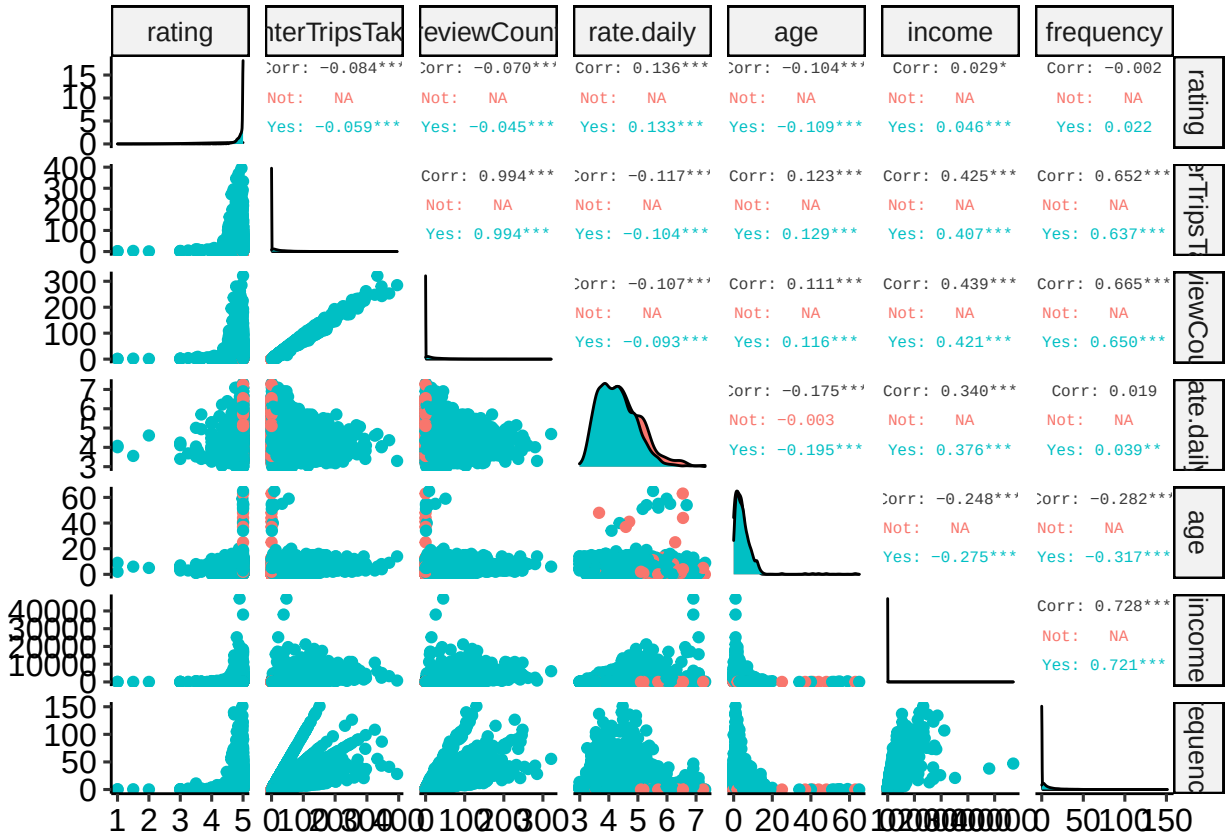

```
# Relació entre el preu diari del lloguer i el tipus de vehicle
ggplot(ds, aes(x=vehicle.type, y=rate.daily, fill=vehicle.type)) + geom_boxplot(alpha=0.1) + stat_summa
```



Si volem visualitzar alhora les relacions creuades entre diverses variables, podem fer un pairplot.

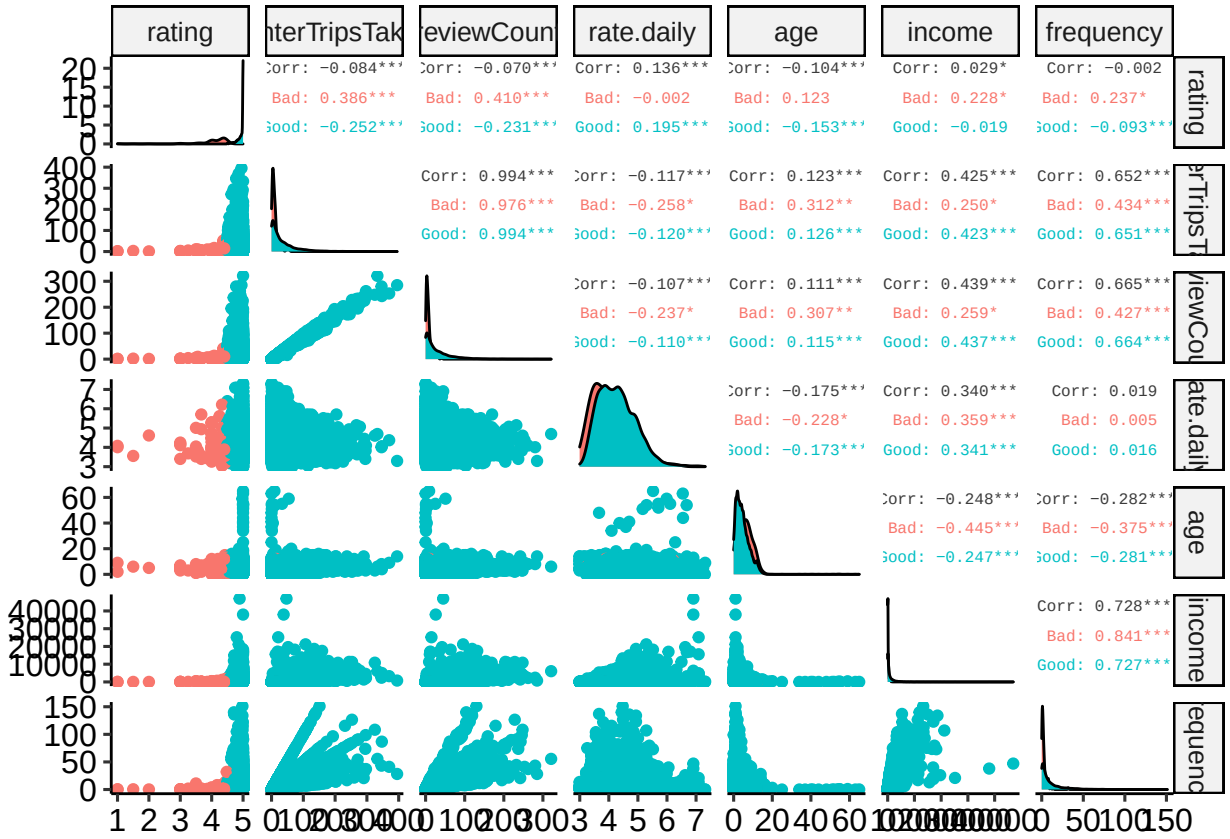
Examinem els atributs numèrics envers si el vehicle va se llogat o no.

```
ggpairs(ds, columns=numeric_features_names, mapping=aes(color=rent), upper = list(continuous = wrap('co
```



Examinem els atributs numèrics envers la qualificació que va obtenir el vehicle.

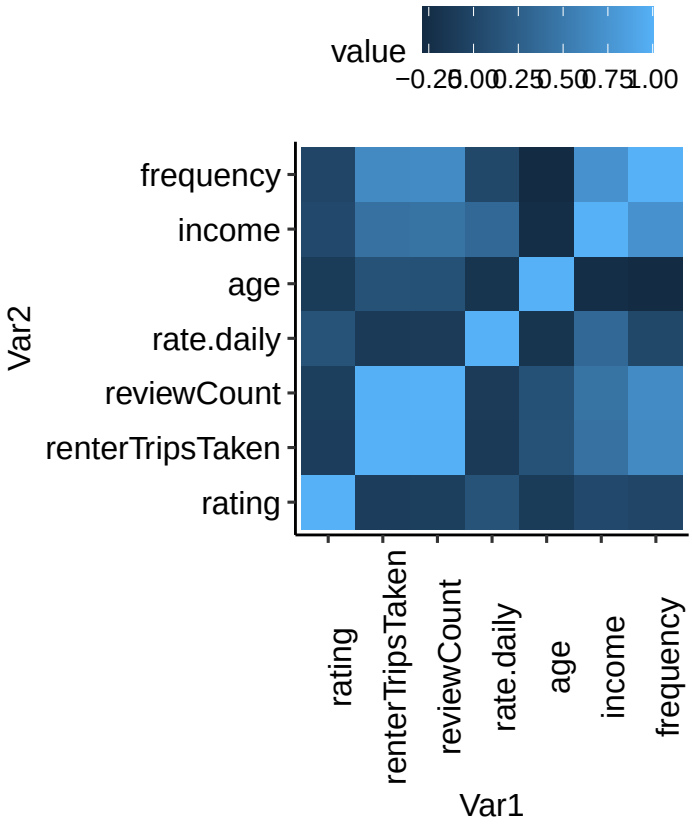
```
ggpairs(ds, columns=numeric_features_names, mapping=aes(color=rating.discret), upper = lis(continuous =
```



Finalment, ens interessarà veure quins conjunts de variables estan relacionats entre si. Per això, farem servir tècniques estadístiques d'anàlisi multivariant.

Una de les eines més útils és calcular la matriu de correlació entre les variables. Amb la funció qplot i la correlació de variables, calculada amb la funció cor, podem visualitzar de manera fàcil aquelles variables més correlacionades, que corresponen a una intensitat major de color.

```
heat <- ds[,numeric_features_names]
qplot(x=Var1, y=Var2, data=melt(cor(heat, use="p")), fill=value, geom="tile") + theme(axis.text.x = ele
```



Test estadístics

¿Quines variables quantitatives influeixen més a les valoracions?

En primer lloc, procedim a realitzar una anàlisi de correlació entre les diferents variables per determinar quines d'elles exerceixen una major influència sobre el preu diari del lloguer del vehicle. Per a això, s'utilitzarà el coeficient de correlació de Spearman, ja que hem vist que tenim dades que no segueixen una distribució normal.

```
corr_matrix <- matrix(nc = 2, nr = 0)
colnames(corr_matrix) <- c("estimate", "p-value")

# Calcular el coeficiente de correlación para cada variable cuantitativa
# con respecto al campo "precio"
for (i in 1:(ncol(ds) - 1)) {
  if (is.integer(ds[,i]) | is.numeric(ds[,i])) {
    spearman_test = cor.test(ds[,i], ds[,'rate.daily'], method = "spearman")
    corr_coef = spearman_test$estimate
    p_val = spearman_test$p.value

    # Add row to matrix
    pair = matrix(ncol = 2, nrow = 1)
    pair[1][1] = corr_coef
    pair[2][1] = p_val
    corr_matrix <- rbind(corr_matrix, pair)
    rownames(corr_matrix)[nrow(corr_matrix)] <- colnames(ds)[i]
  }
```

```
}

print(corr_matrix)
```

```
##                    estimate       p-value
## rating            0.2315008  4.931486e-72
## renterTripsTaken -0.1348053  3.905934e-25
## reviewCount      -0.1255619  5.388263e-22
## rate.daily        1.0000000  0.000000e+00
## vehicle.year      0.3014508 3.439955e-123
## age              -0.3014508 3.439955e-123
## income            0.3280650 7.504998e-147
```

A partir del resultat obtingut podem identificar quines són les variables més correlacionades amb el preu diari de lloguer en funció de la seva proximitat amb els valors -1 i +1. Tenint en compte això, queda palès com la variable més rellevant en la fixació del preu és l'antigitat del vehicle (age) seguida de les valoracions.

Nota. Per a cada coeficient de correlació es mostra també el seu *p-valor* associat, ja que aquest pot donar informació sobre el pes estadístic de la correlació obtinguda.

¿Els cotxes elèctric tenen un preu més elevat que els cotxes de benzina?

Per a avaluar si els cotxes elèctics tenen un preu de lloguer diari més elevat que els cotxes de benzina, podem aplicar un test d'hipòtesis de dues mostres. Tal i com veurem a continuació.

Hipòtesi nul · la i alternativa

Comencem amb la definició de la hipòtesi nul · la i de la hipòtesi alternativa.

- Hipòtesi nul · la:

$$H_0 : \mu_1 = \mu_2$$

- Hipòtesi alternativa:

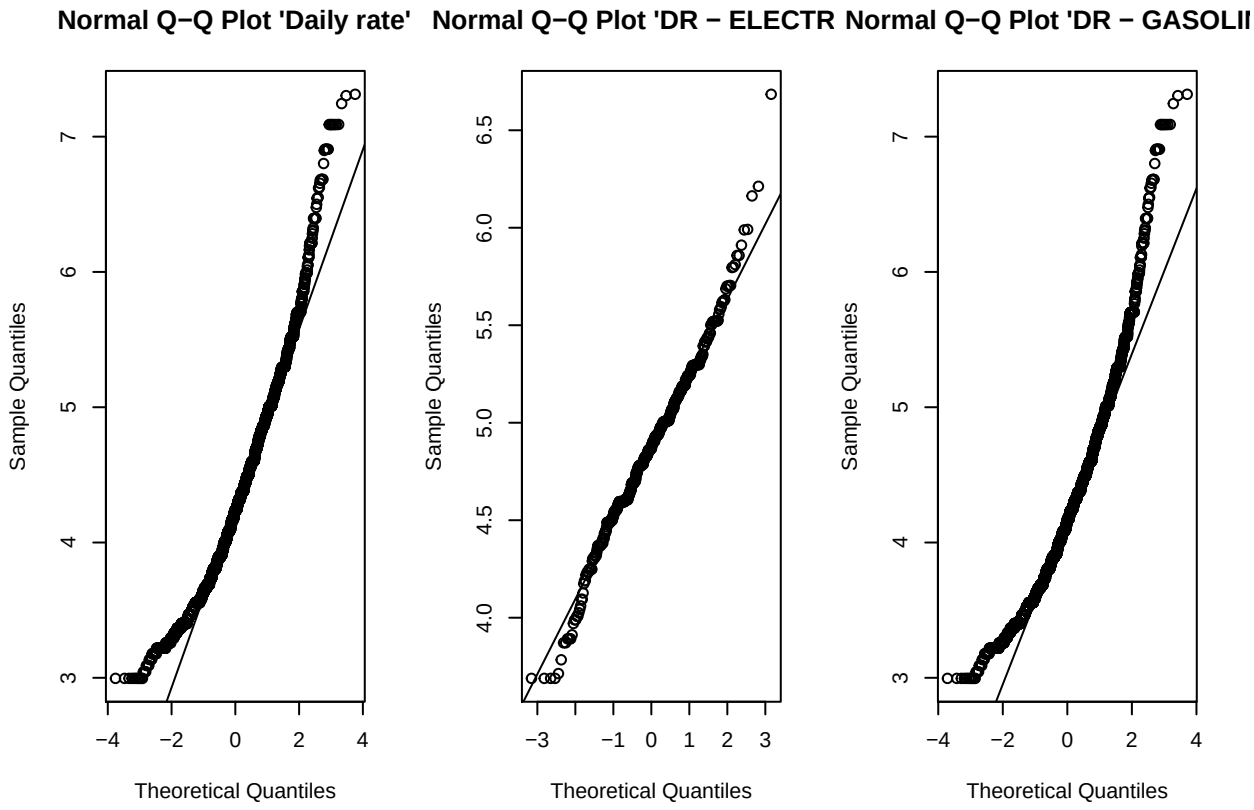
$$H_1 : \mu_1 > \mu_2$$

A continuació revisem si es compleix l'assumpció de normalitat per a la variable 'rate.daily' i a partir d'això, podrem explicar quin test podem aplicar per al test d'hipòtesis de dues mostres.

Test de normalitat

En primer lloc avaluem l'assumpció de normalitat per a la variable 'rate.daily'

```
par(mfrow=c(1,3))
x <- ds$rate.daily
qq1 <- qqnorm(x, main = "Normal Q-Q Plot 'Daily rate'")
qqline(x)

x <- ds$rate.daily[ds$fuelType == 'ELECTRIC']
qq2 <- qqnorm(x, main = "Normal Q-Q Plot 'DR - ELECTRIC'")
qqline(x)

x <- ds$rate.daily[ds$fuelType == 'GASOLINE']
qq3 <- qqnorm(x, main = "Normal Q-Q Plot 'DR - GASOLINE'")
qqline(x)
```



En aquest cas, els punts estan pràcticament sobre la línia, i, per tant, es pot assumir normalitat,

Per tant podem dir que totes dues poblacions es distribueixen normalment, però ens faltaria saber, donat que la variància poblacional és desconeguda, si aquestes poblacions presenten variàncies iguals o variàncies differents.

Test d'igualtat de variàncies

Per a aplicar l'estadístic adequat, cal comprovar si les variàncies de les dues poblacions són iguals. Per això, apliquem primer el test d'igualtat de variàncies.

Per això podem realitzar un altre test, tal que:

- Hipòtesi nul · la:

$$H_0 : \sigma_1^2 = \sigma_2^2$$

- Hipòtesi alternativa:

$$H_1 : \sigma_1^2 \neq \sigma_2^2$$

```
# Obtenim les dades
ELECTRIC <- ds$rate.daily[ds$fuelType == 'ELECTRIC']
GASOLINE <- ds$rate.daily[ds$fuelType == 'GASOLINE']

# Calculem el test d'igualtat de variàncies, amb la funció var.test d'R
var.test(ELECTRIC, GASOLINE)
```

```
##
##  F test to compare two variances
##
```

```
## data:  ELECTRIC and GASOLINE
## F = 0.39604, num df = 621, denom df = 4884, p-value < 2.2e-16
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  0.3528624 0.4469429
## sample estimates:
## ratio of variances
##            0.39604
```

Donat que el valor *p-value* obtingut és menor que el nivell de significació (0.05) podem rebutjar la hipòtesi nul · la i per tant podem dir que la variància de ambdues poblacions és diferent amb un nivell de confiança del 95%.

A partir dels resultats anteriors podem dir que el test a aplicar serà un test d'hipòtesis de dues mostres de poblacions independents amb distribucions normals i variàncies desconegudes i diferents.

Test de la mitjana de dues mostres independents amb variància desconeguda i diferents

Arribats aquest punt, amb la funció *t.test* d'R que ens permet realitzar el contrast d'hipòtesis directament, realitzem el test de mitjanes.

```
# variàncies diferents
t.test( ds$rate.daily[ds$fuelType == 'ELECTRIC'],  ds$rate.daily[ds$fuelType == 'GASOLINE'], alternativ
```

```
##
##  Welch Two Sample t-test
##
## data:  ds$rate.daily[ds$fuelType == "ELECTRIC"] and ds$rate.daily[ds$fuelType == "GASOLINE"]
## t = 35.363, df = 1070.4, p-value < 2.2e-16
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
##  0.6179538       Inf
## sample estimates:
## mean of x mean of y
##  4.874673  4.226547
```

El valor *p-value* obtingut és menor que el nivell de significació (0.05) i, per tant, podem rebutjar la hipòtesi nul · la d'igualtat de mitjanes de preu de lloguer diari entre els cotxes elèctris i els cotxes de benzina. Per tant podem dir que els cotxes électrics tenen un lloguer més elevat que els de benzina, amb un nivell de confiança del 95%.

¿La proporció de furgonetes és més petit que la d'utilitaris?

Ara ens preguntem si la proporció de furgonetes de lloguer és més petit que la d'utilitaris o vehicles convencionals.

Hipòtesi nul · la i alternativa

Per tal de donar resposta a la pregunta formulada, comencem amb la definició de la hipòtesi nul · la i de la hipòtesi alternativa.

- Hipòtesi nul · la:

$$H_0 : p = 0.5$$

- Hipòtesi alternativa:

$$H_1 : p_1 < 0.5$$

En aquest cas específicament, ens preguntem si la proporció de furgonetes és més petita que la d'utilitaris, o el que és el mateix, si la proporció de furgonetes és igual a 0.5.

Test unilateral d'una mostra sobre la proporció

A continuació podem realitzar els càlculs pertinents que ens permetin decidir si podem rebutjar la hipòtest nul·la o no.

Podem fer servir, la funció prop.test pròpia d'R que ens permet realitzar directament contrastos d'hipòtesis sobre proporcions.

```
n <- length(ds$vehicle.type)

prop.test(x=sum(ds$vehicle.type == "van"  | ds$vehicle.type == "minivan"), n=n, p=0.5, alternative="less
```

```
##
##  1-sample proportions test without continuity correction
##
## data:  sum(ds$vehicle.type == "van" | ds$vehicle.type == "minivan") out of n, null probability 0.5
## X-squared = 4759.3, df = 1, p-value < 2.2e-16
## alternative hypothesis: true p is less than 0.5
## 95 percent confidence interval:
##  0.00000000 0.05390773
## sample estimates:
##          p
## 0.04905144
```

D'altra banda el *p-valor* és inferior al nivell de significació (*p-value* $<$), per tant podem rebutjar la hipòtesi nul · la. Això ens permet dir que la proporció de furgonetes és més petita que la d'utilitàris, amb un nivell de confiança del 95%.

Model de regressió lineal múltiple per preveure el preu diari d'un vehicle

A continuació ens proposem estimar per mínims quadrats ordinaris un model lineal que expliqui la variable *rate.daily* en funció de *age i renterTripsTaken*. En aquest cas farem servir només atributs numèrics.

Model de regressió lineal múltiple 1 (Preu ~ age + renterTripsTaken)

```
model.lm1 <- lm(formula=rate.daily ~ age + renterTripsTaken, data = ds)

summary(model.lm1)
```

```
##
## Call:
## lm(formula = rate.daily ~ age + renterTripsTaken, data = ds)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.3593 -0.4833 -0.0676  0.3867  3.7207
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)       4.4620496  0.0137813 323.776  < 2e-16 ***
## age              -0.0260129  0.0020630 -12.609  < 2e-16 ***
## renterTripsTaken -0.0014895  0.0001995  -7.468 9.34e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6343 on 5848 degrees of freedom
## Multiple R-squared:  0.03969,    Adjusted R-squared:  0.03936
## F-statistic: 120.8 on 2 and 5848 DF,  p-value: < 2.2e-16
```

Així doncs l'equació de regressió és:

$$\hat{y} = -0.0260 * age - 0.0014 * renterTripsTaken + 4.46$$

Es pot observar que tant la variable *age i renterTripsTaken* són significatives perquè Pr(>|t|)< 0,05.

Finalment el coeficient de determinació ajustat per aquest model és: R2 = 0.03936. Això ens diu que el model de regressió múltiple obtingut explica el 3.936% de la variabilitat del preu del lloguer diari de vehicles. Com que és molt proper al 0%, en principi és un model bastant dolent, i per tant tindrà poc poder predictiu, gairebè nul.

Anem a veure ara si amb la introducció de noves variables al model, aconseguim un altre model que presenti una millor capacitat predictora. En aquest cas utilitzarem només atributs categòrics.

Model de regressió lineal mùltiple 2 (Preu ~ age + renterTripsTaken + fuelType + vehicle.make + vehicle.type)

```
model.lm2 <- lm(formula=rate.daily ~ fuelType + vehicle.make + vehicle.type, data = ds)

summary(model.lm2)
```

```
##
## Call:
## lm(formula = rate.daily ~ fuelType + vehicle.make + vehicle.type,
##     data = ds)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.5043 -0.2911 -0.0468  0.2355  3.2924
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 4.07880    0.08580  47.540  < 2e-16 ***
## fuelTypeELECTRIC            0.12832    0.08978   1.429 0.152980
## fuelTypeGASOLINE            0.06053    0.05669   1.068 0.285669
## fuelTypeHYBRID              0.14308    0.06396   2.237 0.025320 *
## vehicle.makeAlfa Romeo      0.51476    0.10366   4.966 7.04e-07 ***
## vehicle.makeAston Martin    1.85325    0.21065   8.798  < 2e-16 ***
## vehicle.makeAudi            0.24858    0.07236   3.435 0.000596 ***
## vehicle.makeBentley         1.75557    0.16270  10.790  < 2e-16 ***
## vehicle.makeBMW             0.33870    0.06708   5.049 4.57e-07 ***
## vehicle.makeBuick          -0.35931    0.12094  -2.971 0.002982 **
## vehicle.makeCadillac        0.37953    0.09277   4.091 4.35e-05 ***
## vehicle.makeChevrolet       0.07421    0.06731   1.103 0.270272
## vehicle.makeChrysler       -0.26114    0.08874  -2.943 0.003265 **
## vehicle.makeDodge          -0.12008    0.07281  -1.649 0.099182 .
## vehicle.makeFerrari         2.34652    0.13996  16.766  < 2e-16 ***
## vehicle.makeFIAT           -0.37665    0.09428  -3.995 6.55e-05 ***
## vehicle.makeFord           -0.12857    0.06738  -1.908 0.056436 .
## vehicle.makeGenesis         0.09885    0.32370   0.305 0.760086
## vehicle.makeGMC             0.07148    0.08465   0.844 0.398501
## vehicle.makeHonda          -0.34826    0.06981  -4.988 6.27e-07 ***
## vehicle.makeHUMMER          0.61251    0.26683   2.295 0.021741 *
## vehicle.makeHyundai        -0.47109    0.07182  -6.559 5.89e-11 ***
## vehicle.makeJaguar          0.76613    0.08684   8.822  < 2e-16 ***
## vehicle.makeJeep           -0.05524    0.06924  -0.798 0.425006
```

```
## vehicle.makeKia           -0.47586    0.07336  -6.486 9.53e-11 ***
## vehicle.makeLamborghini    2.65598    0.13226  20.081  < 2e-16 ***
## vehicle.makeLand Rover     0.64097    0.07904   8.109 6.17e-16 ***
## vehicle.makeLexus          0.01466    0.07567   0.194 0.846362
## vehicle.makeLincoln        0.01365    0.11673   0.117 0.906936
## vehicle.makeLotus          1.44270    0.23333   6.183 6.71e-10 ***
## vehicle.makeMaserati       0.85090    0.08414  10.113  < 2e-16 ***
## vehicle.makeMazda         -0.31932    0.08298  -3.848 0.000120 ***
## vehicle.makeMcLaren        2.54403    0.45329   5.612 2.09e-08 ***
## vehicle.makeMercedes-Benz  0.40793    0.06757   6.037 1.66e-09 ***
## vehicle.makeMercury       -0.78003    0.32367  -2.410 0.015985 *
## vehicle.makeMINI          -0.30321    0.10293  -2.946 0.003233 **
## vehicle.makeMitsubishi    -0.55365    0.10477  -5.284 1.31e-07 ***
## vehicle.makeNissan        -0.40413    0.06880  -5.874 4.50e-09 ***
## vehicle.makePolaris        0.84097    0.09206   9.135  < 2e-16 ***
## vehicle.makePontiac       -0.63323    0.21065  -3.006 0.002658 **
## vehicle.makePorsche        0.83392    0.07156  11.654  < 2e-16 ***
## vehicle.makeRam           -0.11564    0.12440  -0.930 0.352612
## vehicle.makeRolls Royce    2.61386    0.26688   9.794  < 2e-16 ***
## vehicle.makeSaab          -0.40166    0.45329  -0.886 0.375598
## vehicle.makeSaturn        -0.86237    0.32370  -2.664 0.007742 **
## vehicle.makeScion         -0.57237    0.13996  -4.090 4.38e-05 ***
## vehicle.makesmart         -0.61897    0.13603  -4.550 5.47e-06 ***
## vehicle.makeSubaru        -0.32551    0.08039  -4.049 5.20e-05 ***
## vehicle.makeSuzuki        -0.53814    0.32360  -1.663 0.096369 .
## vehicle.makeTesla          0.66797    0.09555   6.991 3.04e-12 ***
## vehicle.makeToyota        -0.40603    0.06649  -6.107 1.08e-09 ***
## vehicle.makeVolkswagen    -0.33645    0.07479  -4.499 6.97e-06 ***
## vehicle.makeVolvo         -0.11212    0.11481  -0.977 0.328814
## vehicle.typeminivan        0.18388    0.03400   5.408 6.64e-08 ***
## vehicle.typesuv            0.16533    0.01511  10.943  < 2e-16 ***
## vehicle.typetruck          0.34776    0.03641   9.552  < 2e-16 ***
## vehicle.typevan            0.46716    0.06271   7.450 1.07e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4487 on 5748 degrees of freedom
##   (46 observations deleted due to missingness)
## Multiple R-squared:  0.5269, Adjusted R-squared:  0.5222
## F-statistic: 114.3 on 56 and 5748 DF,  p-value: < 2.2e-16
```

Si examinem ara el coeficient de determinació ajustat per aquest nou model és: R2 = 0.5222. Això ens diu que el model de regressió múltiple obtingut explica el 52.22% de la variabilitat del preu del lloguer diari de vehicles. Com que és molt proper al 50%, en principi no és un model gaire bo, i per tant tindrà poc poder predictiu. Tot i així podem veure com hi ha un increment considerable sobre la variabilitat explicada respecte al primer model.

Model de regressió lineal mùltiple 3 (Preu ~ age + renterTripsTaken + fuelType + vehicle.make + vehicle.type)

Ara probarem a generar un nou model, però aquest cop combinarem atributs categòrics i atributs numèrics com a variables predictores.

```
model.lm3 <- lm(formula=rate.daily ~ age + renterTripsTaken + fuelType + vehicle.make + vehicle.type, da

summary(model.lm3)
```

```
##
## Call:
## lm(formula = rate.daily ~ age + renterTripsTaken + fuelType +
##     vehicle.make + vehicle.type, data = ds)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.4680 -0.2859 -0.0455  0.2229  3.5232
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 4.2129771  0.0852347  49.428  < 2e-16 ***
## age                        -0.0180194  0.0015282 -11.791  < 2e-16 ***
## renterTripsTaken           -0.0005818  0.0001425  -4.084 4.49e-05 ***
## fuelTypeELECTRIC            0.0669464  0.0886435   0.755 0.450140
## fuelTypeGASOLINE            0.0366118  0.0559475   0.654 0.512882
## fuelTypeHYBRID              0.1161880  0.0631006   1.841 0.065627 .
## vehicle.makeAlfa Romeo      0.4754594  0.1023439   4.646 3.46e-06 ***
## vehicle.makeAston Martin    1.9063379  0.2077669   9.175  < 2e-16 ***
## vehicle.makeAudi            0.2443705  0.0713470   3.425 0.000619 ***
## vehicle.makeBentley         1.8213785  0.1605432  11.345  < 2e-16 ***
## vehicle.makeBMW             0.3425522  0.0661572   5.178 2.32e-07 ***
## vehicle.makeBuick          -0.3516906  0.1192445  -2.949 0.003198 **
## vehicle.makeCadillac        0.3618207  0.0914731   3.955 7.73e-05 ***
## vehicle.makeChevrolet       0.0760052  0.0664153   1.144 0.252509
## vehicle.makeChrysler       -0.2578170  0.0874902  -2.947 0.003224 **
## vehicle.makeDodge          -0.1382037  0.0718340  -1.924 0.054412 .
## vehicle.makeFerrari         2.3946067  0.1380574  17.345  < 2e-16 ***
## vehicle.makeFIAT           -0.3429167  0.0930031  -3.687 0.000229 ***
## vehicle.makeFord           -0.1208009  0.0664552  -1.818 0.069150 .
## vehicle.makeGenesis         0.0496436  0.3191696   0.156 0.876401
## vehicle.makeGMC             0.0802021  0.0834647   0.961 0.336638
## vehicle.makeHonda          -0.3292382  0.0688887  -4.779 1.80e-06 ***
## vehicle.makeHUMMER          0.7483759  0.2633272   2.842 0.004499 **
## vehicle.makeHyundai        -0.4941164  0.0708627  -6.973 3.45e-12 ***
## vehicle.makeJaguar          0.7527355  0.0856408   8.789  < 2e-16 ***
## vehicle.makeJeep           -0.0629590  0.0683324  -0.921 0.356899
## vehicle.makeKia            -0.4912196  0.0723641  -6.788 1.25e-11 ***
## vehicle.makeLamborghini     2.6388388  0.1304074  20.235  < 2e-16 ***
## vehicle.makeLand Rover      0.6378828  0.0779297   8.185 3.31e-16 ***
## vehicle.makeLexus           0.0323550  0.0746250   0.434 0.664619
## vehicle.makeLincoln         0.0019300  0.1150908   0.017 0.986621
## vehicle.makeLotus           1.4907056  0.2300863   6.479 1.00e-10 ***
## vehicle.makeMaserati        0.8271252  0.0829773   9.968  < 2e-16 ***
## vehicle.makeMazda          -0.3129651  0.0818431  -3.824 0.000133 ***
## vehicle.makeMcLaren         2.5081768  0.4469113   5.612 2.09e-08 ***
## vehicle.makeMercedes-Benz   0.4044302  0.0666228   6.070 1.36e-09 ***
## vehicle.makeMercury        -0.6450329  0.3192964  -2.020 0.043412 *
## vehicle.makeMINI           -0.2902223  0.1014796  -2.860 0.004253 **
## vehicle.makeMitsubishi     -0.5716103  0.1033372  -5.532 3.32e-08 ***
## vehicle.makeNissan         -0.4106705  0.0678904  -6.049 1.55e-09 ***
## vehicle.makePolaris         0.7878410  0.0908630   8.671  < 2e-16 ***
## vehicle.makePontiac        -0.5043263  0.2079205  -2.426 0.015315 *
## vehicle.makePorsche         0.8700409  0.0706357  12.317  < 2e-16 ***
```

```
## vehicle.makeRam           -0.1206824  0.1226488  -0.984 0.325173
## vehicle.makeRolls Royce    2.8818177  0.2641864  10.908  < 2e-16 ***
## vehicle.makeSaab          -0.3142571  0.4469561  -0.703 0.482019
## vehicle.makeSaturn        -0.7235348  0.3193566  -2.266 0.023513 *
## vehicle.makeScion         -0.4898361  0.1381880  -3.545 0.000396 ***
## vehicle.makesmart         -0.5852443  0.1341391  -4.363 1.31e-05 ***
## vehicle.makeSubaru        -0.3391753  0.0792953  -4.277 1.92e-05 ***
## vehicle.makeSuzuki        -0.4026357  0.3193619  -1.261 0.207450
## vehicle.makeTesla          0.6539141  0.0943228   6.933 4.58e-12 ***
## vehicle.makeToyota        -0.3732875  0.0656584  -5.685 1.37e-08 ***
## vehicle.makeVolkswagen    -0.3321377  0.0737345  -4.505 6.78e-06 ***
## vehicle.makeVolvo         -0.1356465  0.1132064  -1.198 0.230880
## vehicle.typeminivan        0.1906928  0.0335688   5.681 1.41e-08 ***
## vehicle.typesuv            0.1505576  0.0149564  10.066  < 2e-16 ***
## vehicle.typetruck          0.3159204  0.0360137   8.772  < 2e-16 ***
## vehicle.typevan            0.4352117  0.0619785   7.022 2.44e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4424 on 5746 degrees of freedom
##   (46 observations deleted due to missingness)
## Multiple R-squared:  0.5403, Adjusted R-squared:  0.5356
## F-statistic: 116.4 on 58 and 5746 DF,  p-value: < 2.2e-16
```

Si examinem ara el coeficient de determinació ajustat per aquest nou model és: R2 = 0.5356. Això ens diu que el model de regressió múltiple obtingut explica el 53.56% de la variabilitat del preu del lloguer diari de vehicles. Com que és molt proper al 50%, en principi no és un model gaire bo, i per tant tindrà poc poder predictiu. Tot i així podem veure com hi ha un increment considerable sobre la variabilitat explicada respecte al primer model, i un increment poc significatiu respecte el segon model. La qual cosa ens indica que a priori els atributs que millor expliquen el preu són les característiques del vehicle: el combustible, la marca i el tipus de vehicle.

Model d'arbre de regressió per preveure el preu diari d'un vehicle

Comencem triant el subconjunt d'entrenament i el de prova. Nosaltres crearem dos conjunts de dades directament amb un rang.

```
# Creem el conjunt d'entrenament i el de prova
set.seed(555)
indexes = sample(1:nrow(ds), size=floor((2/3)*nrow(ds)))
train <- ds[indexes,]
test  <- ds[-indexes,]
```

Després d'una extracció aleatòria de casos cal realitzar una anàlisi de dades mínim per assegurar-nos de no obtenir valors esbiaixats pels valors que conté cada mostra.

```
# Verifiquem les dimensions del conjunt d'entrenament
dim(train)
```

```
## [1] 3900   18
```

```
# Verifiquem les dimensions del conjunt de prova
dim(test)
```

```
## [1] 1951   18
```

```
# Obtenim l'atribut de classe de la resta
trainY <- train[,c('rate.daily')]
```

```
testY <- test[,c('rate.daily')]
```

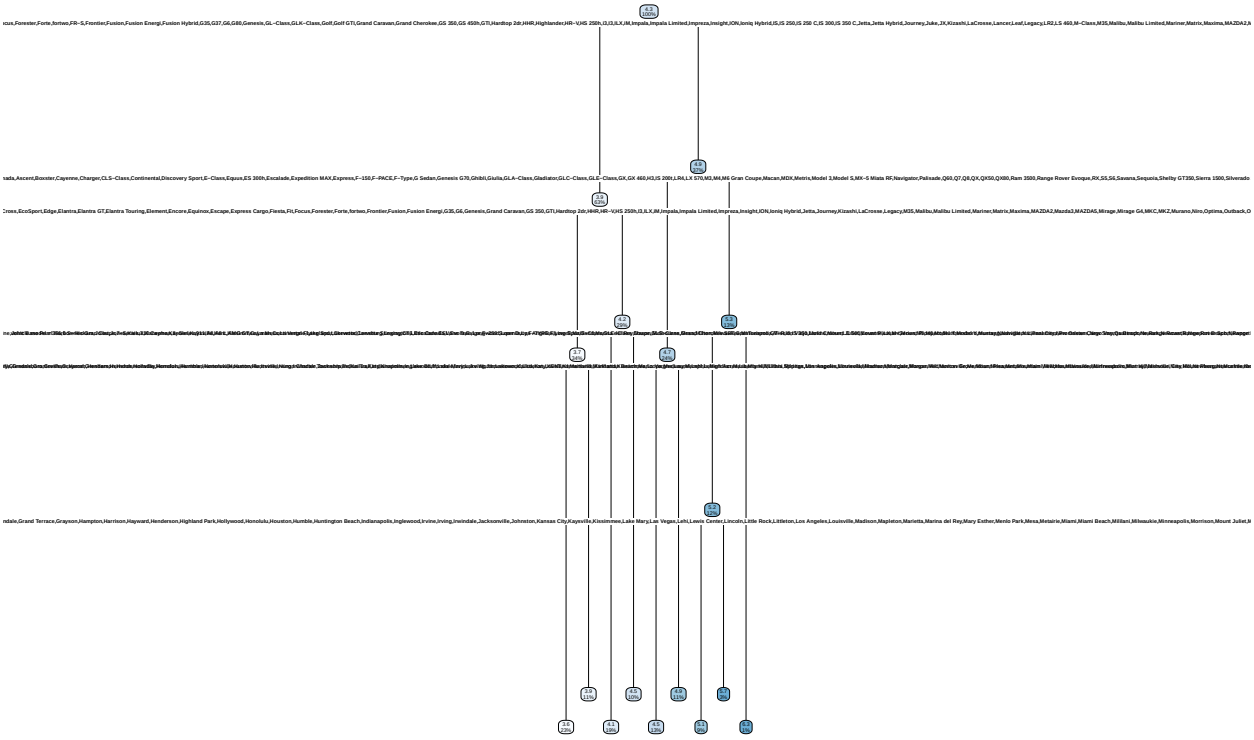
Model d'Arbre de regressió

A continuació ens proposem generar un model d'arbre de regressió que expliqui la variable *rate.daily* en funció de les característiques del vehicle.

```
# Generem el model d'arbre
model_cart_rate <- rpart(rate.daily ~ fuelType + location.city + location.state + vehicle.make + vehicl

# Mostrem un resum de la generació de l'arbre
#summary(model_cart)

printcp(model_cart_rate)
```

```
##
## Regression tree:
## rpart(formula = rate.daily ~ fuelType + location.city + location.state +
##     vehicle.make + vehicle.model + vehicle.type, data = train,
##     method = "anova")
##
## Variables actually used in tree construction:
## [1] location.city vehicle.model
##
## Root node error: 1630.5/3900 = 0.41807
##
## n= 3900
##
##         CP nsplit rel error  xerror     xstd
## 1 0.527308      0   1.00000 1.00043 0.025917
## 2 0.108160      1   0.47269 0.54113 0.018111
## 3 0.066292      2   0.36453 0.43993 0.017114
## 4 0.027720      3   0.29824 0.39122 0.015361
## 5 0.023440      4   0.27052 0.38733 0.015530
## 6 0.023387      5   0.24708 0.38461 0.015569
## 7 0.021177      6   0.22369 0.38456 0.015605
## 8 0.017006      7   0.20252 0.37918 0.015551
## 9 0.010000      8   0.18551 0.37992 0.015469
```

```
# Mostrem en un gràfic l'arbre obtingut
rpart.plot(model_cart_rate)
```



Un cop generat el model, podem comprovar la seva qualitat predint la classe per a les dades de prova que hem reservat al principi.

```
predicted_model_cart_rate <- predict(model_cart_rate, test, type = "vector")

# test RMSE
rmse_cart_rate <- rmse(predicted_model_cart_rate, testY)

# test MAE
mae_cart_rate <- mae(predicted_model_cart_rate, testY)

print(sprintf("RMSE: %.4f", rmse_cart_rate))
```

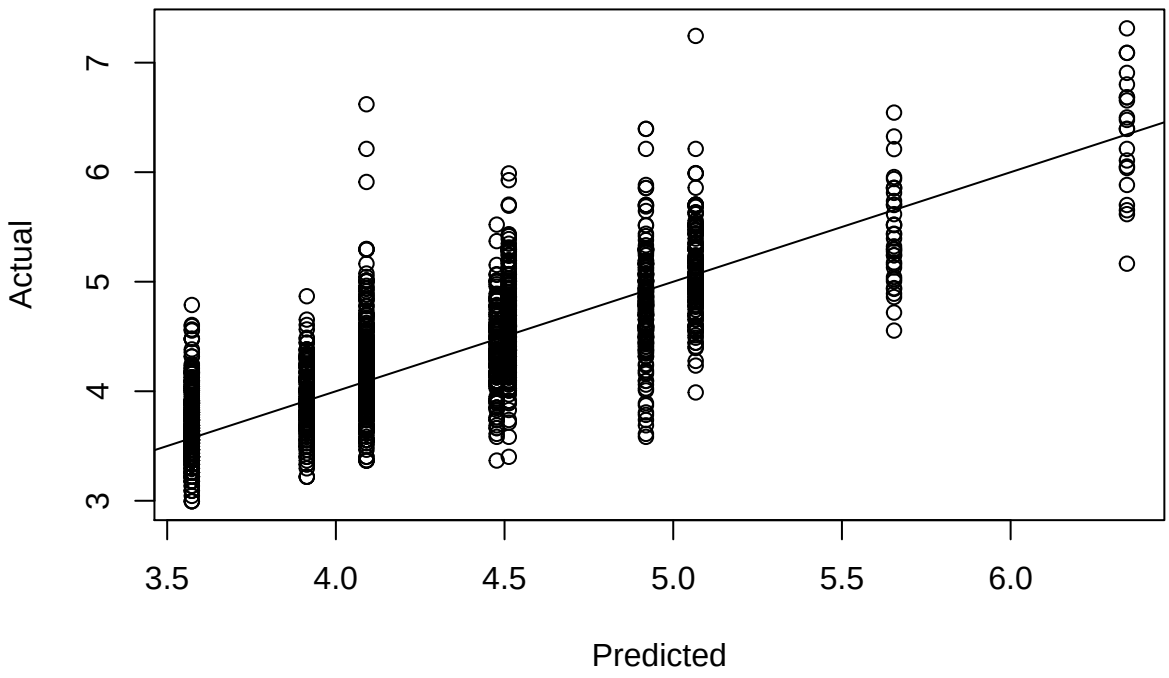
```
## [1] "RMSE: 0.3830"
```

```
print(sprintf("MAE: %.4f", rmse_cart_rate))
```

```
## [1] "MAE: 0.3830"
```

Examinem els valors predits envers els valors reals.

```
plot(predicted_model_cart_rate, testY, xlab = "Predicted", ylab = "Actual")
abline(0, 1)
```



Examinem la importància de cada una dels atributs predictors en el model.

```
model_cart_rate$variable.importance
```

```
##  vehicle.model   vehicle.make  location.city       fuelType location.state
##     1211.96274      615.77939      545.53745      239.29902      127.07061
##   vehicle.type
##       49.25973
```

A partir del resultat obtingut podem veure que l'atribut que més influeix en el preu del lloguer dels vehicles és el model de cotxe.

Model d'arbre de classificació per preveure si un vehicle serà llogat

A continuació ens proposem generar un model d'arbre de classificació que expliqui la variable *rent* en funció de les característiques del vehicle. És a dir que ens permeti donades les característiques d'un vehicle predir si aquest serà llogat o no.

Model d'Arbre de classificació

Obtenim la classe objectiu

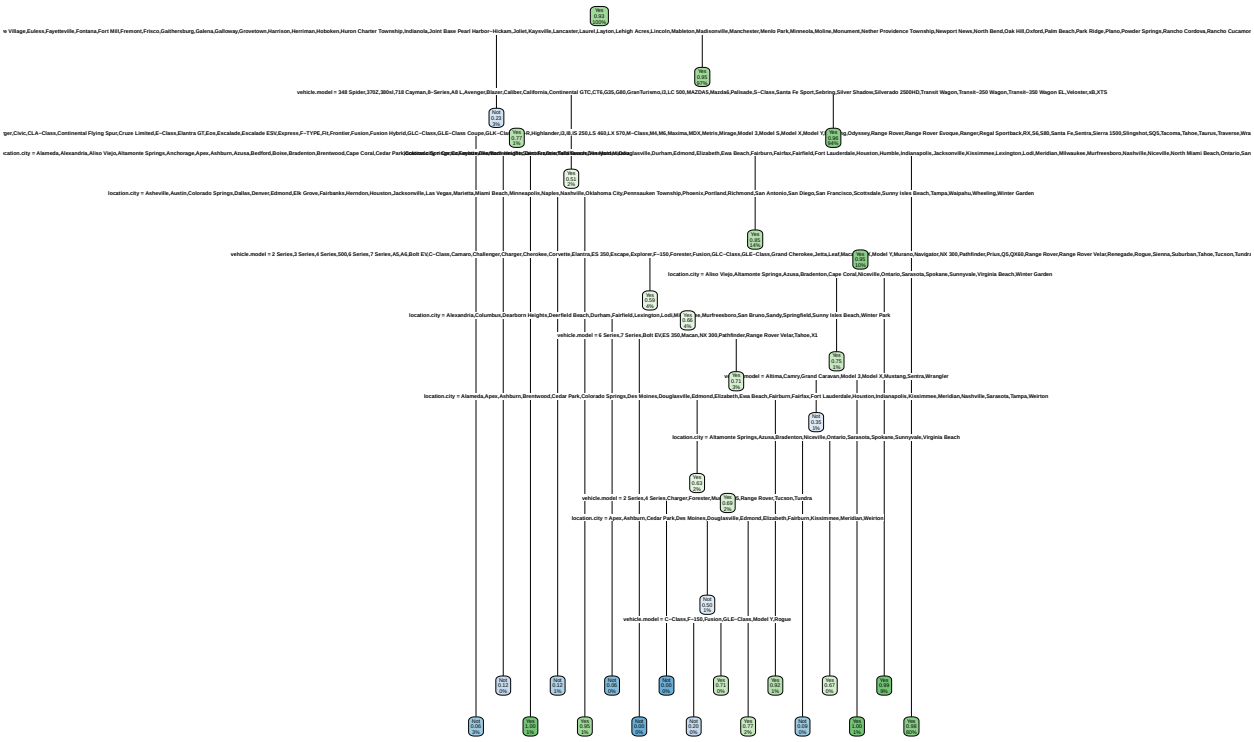
```
# Obtenim l'atribut de classe de la resta
trainY <- train[,c('rent')]
testY <- test[,c('rent')]
```

```
# Generem el model d'arbre
model_cart_rent <- rpart(rent ~ fuelType + location.city + location.state + vehicle.make + vehicle.model
```

```r
# Mostrem un resum de la generació de l'arbre
#summary(model_cart)

printcp(model_cart_rent)
```

```
##
## Classification tree:
## rpart(formula = rent ~ fuelType + location.city + location.state +
##     vehicle.make + vehicle.model + vehicle.type, data = train,
##     method = "class")
##
## Variables actually used in tree construction:
## [1] location.city vehicle.model
##
## Root node error: 291/3900 = 0.074615
##
## n= 3900
##
##         CP nsplit rel error xerror     xstd
## 1 0.240550      0   1.00000 1.0000 0.056392
## 2 0.063574      1   0.75945 1.1134 0.059231
## 3 0.058419      3   0.63230 1.1787 0.060780
## 4 0.020619      4   0.57388 1.1718 0.060620
## 5 0.018328      5   0.55326 1.2818 0.063115
## 6 0.015464      9   0.46048 1.2990 0.063491
## 7 0.010309     11   0.42955 1.3196 0.063939
## 8 0.010000     16   0.37801 1.3162 0.063865
```

```r
# Mostrem en un gràfic l'arbre obtingut
rpart.plot(model_cart_rent)
```



Un cop generat el model, podem comprovar la seva qualitat predint la classe per a les dades de prova que hem reservat al principi.

```
predicted_model_cart_rent <- predict(model_cart_rent, test, type = "class")

print(sprintf("La precisió de l'arbre és: %.4f %%", 100 * sum(predicted_model_cart_rent == testY) / leng
```

```
## [1] "La precisió de l'arbre és: 90.0051 %"
```

Quan hi ha poques classes, la qualitat de la predicció també es pot analitzar mitjançant una matriu de confusió que identifica els tipus d'errors comesos.

```
mat_conf_cart_rent <- table(testY, Predicted=predicted_model_cart_rent)

mat_conf_cart_rent
```

```
##      Predicted
## testY  Not  Yes
##   Not    6  134
##   Yes   61 1750
```

Una altra manera de calcular el percentatge de registres correctament classificats és utilitzant la matriu de confusió:

```
porcentaje_correct <- 100 * sum(diag(mat_conf_cart_rent)) / sum(mat_conf_cart_rent)
print(sprintf("El %% de registres correctament classificats és: %.4f %%", porcentaje_correct))
```

```
## [1] "El % de registres correctament classificats és: 90.0051 %"
```

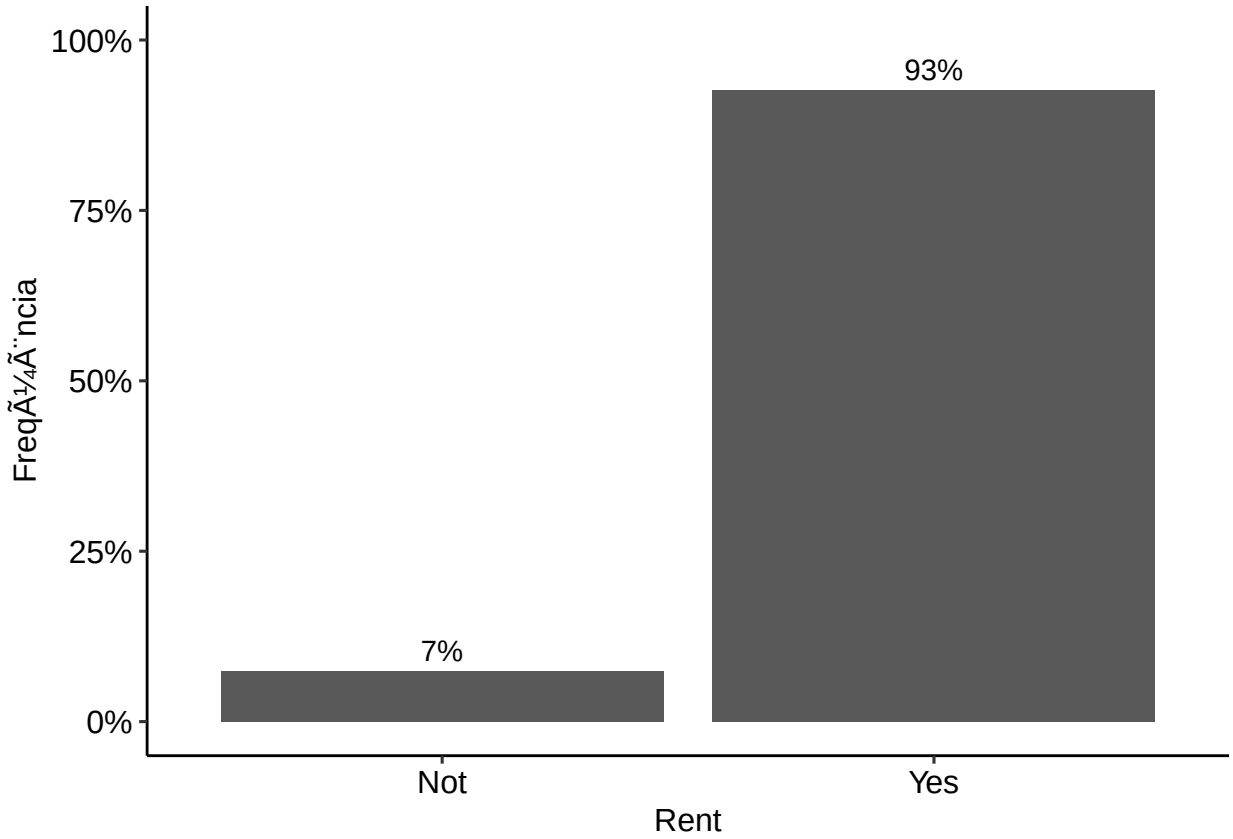
A més, tenim a la nostra disposició el paquet gmodels per a obtenir informació més completa

```
CrossTable(testY, predicted_model_cart_rent, prop.chisq  = FALSE, prop.c = FALSE, prop.r =FALSE, dnn = c
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  1951
##
##
##              | Prediction
##      Reality |       Not |       Yes | Row Total |
## -------------|-----------|-----------|-----------|
##          Not |         6 |       134 |       140 |
##              |     0.003 |     0.069 |           |
## -------------|-----------|-----------|-----------|
##          Yes |        61 |      1750 |      1811 |
##              |     0.031 |     0.897 |           |
## -------------|-----------|-----------|-----------|
## Column Total |        67 |      1884 |      1951 |
## -------------|-----------|-----------|-----------|
##
##
```

Durant tot l'estudi previ em deixat de banda un fet important que es va veure durant la fase d'anàlisis del joc de dades. Si recordem el nombre d'observacions pertanyents a una classe respecte a l'altre és força diferent per a l'atribut objectiu 'rent'.

```
# Calculem histogrames de les variables de classe
ggplot(data=ds, aes(x = rent)) + geom_bar(aes(y = (..count..)/sum(..count..))) + geom_text(aes(y = ((..c
```

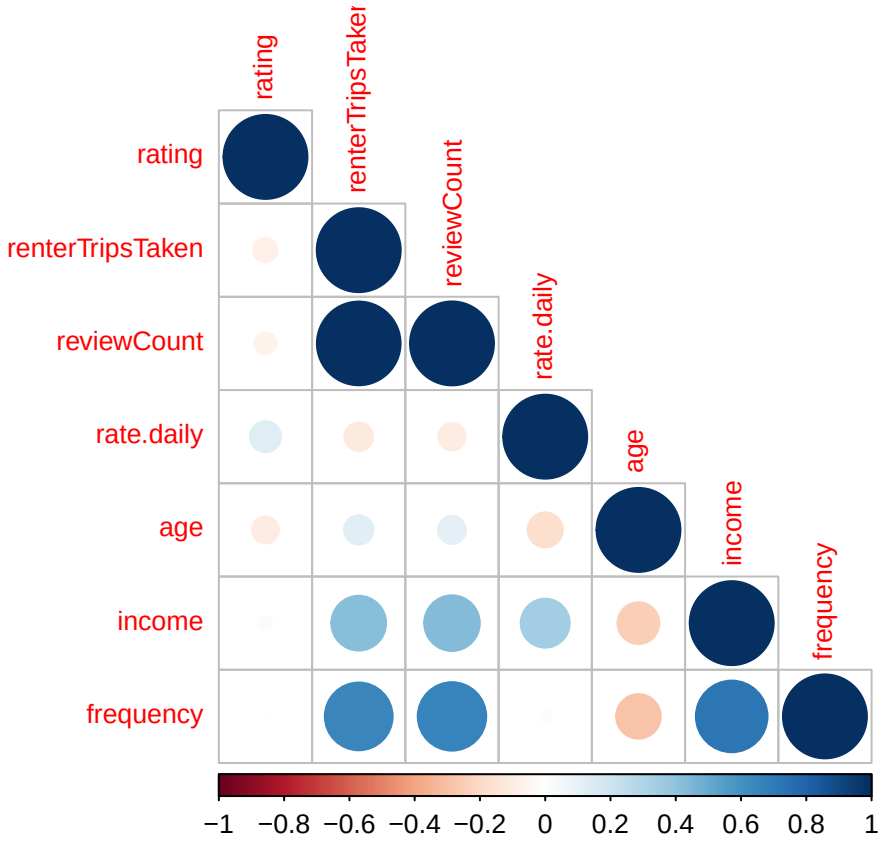


Les classificacions desequilibrades suposen un desafiament per a la generació de models de predicció, ja que la majoria dels algorismes utilitzats per a la classificació van ser dissenyats entorn l'assumpció d'un nombre igual d'observacions per a cada classe. Això produeix que els models obtinguts presentin un rendiment predictiu deficient, en concret per a la classe minoritària. A més això sol ser un problema perqué normalment, la classe minoritària és la més important i per tant el model és més sensible als errors de classificació de la classe minoritària que la classe majoritària.

Generar un model que pugui preveure la freqüencia amb que es lloga un automòbil de lloguer (frequency).

En primer lloc fem una visualització de les correlacions entre les variables númeriques.

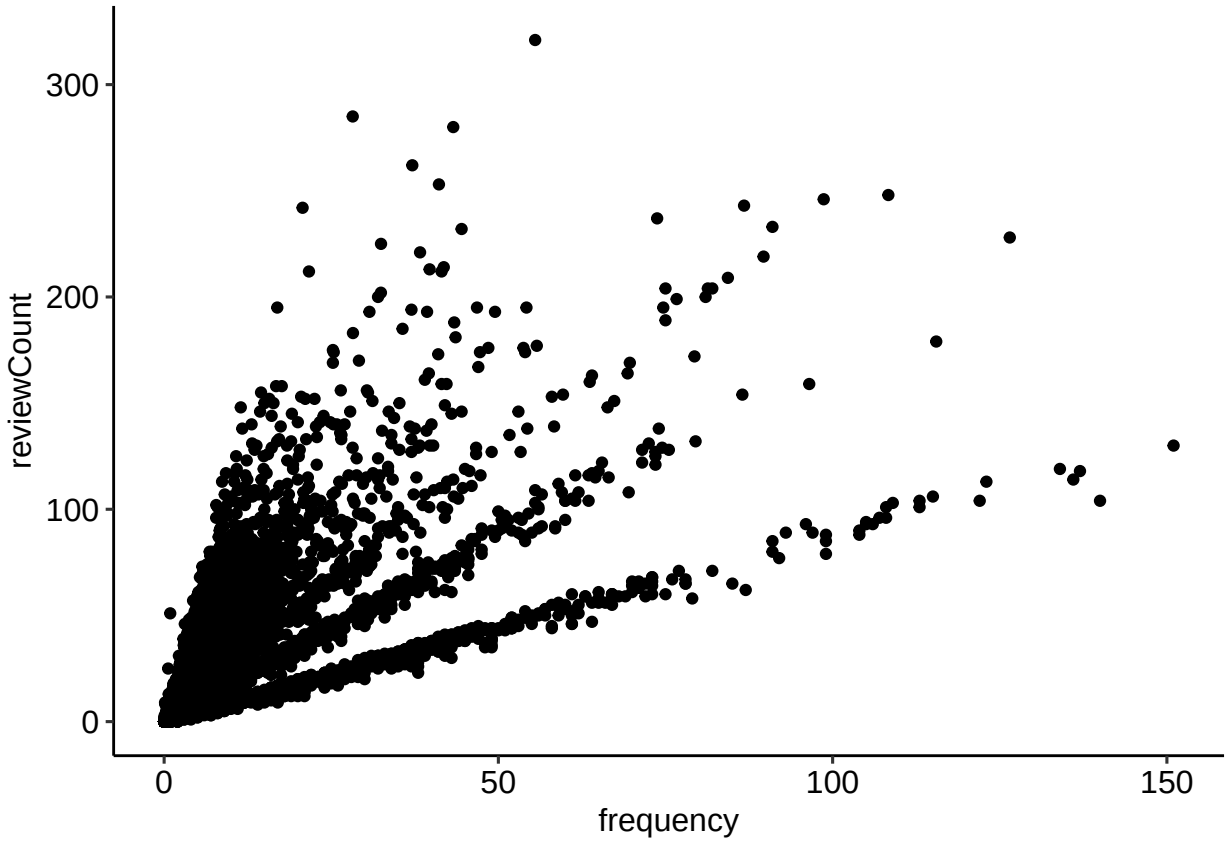
```
ds.num <- ds[numeric_features_names]
corr.res<-cor(ds.num)
corrplot(corr.res, type="lower", number.cex=.7,tl.cex=.8)
```



S'observa clarament que **frequency** mostra correlació amb **income**, **reviewCount**, **renterTripsTaken** i en menor mesura amb **age**.

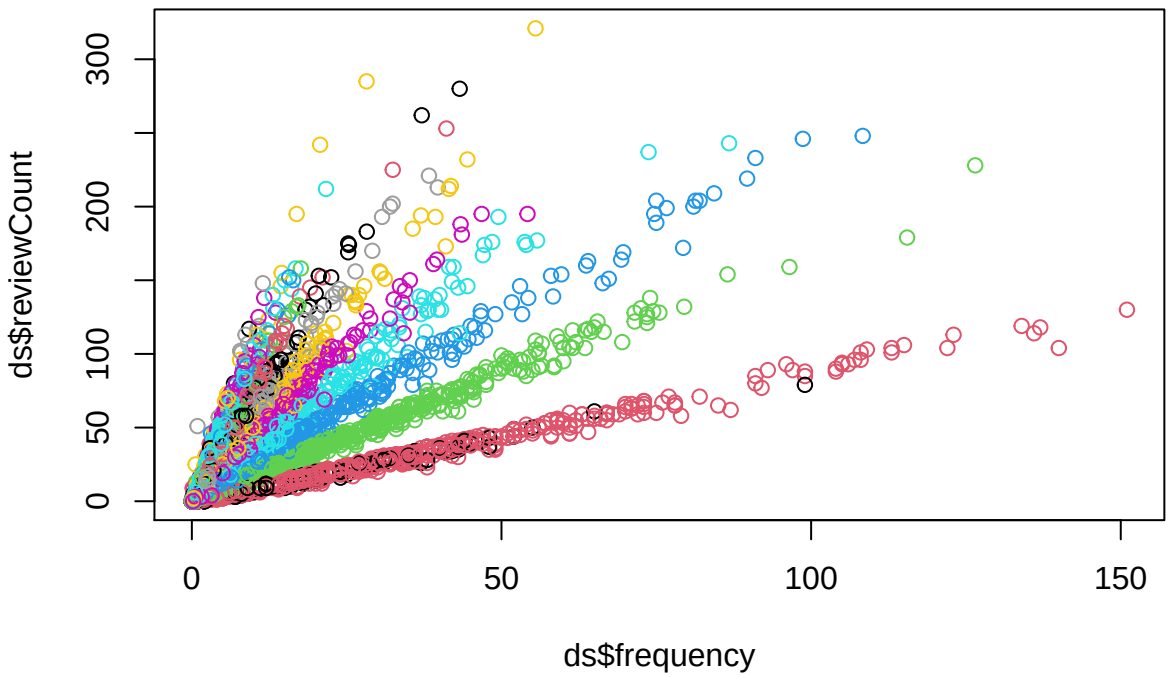
Que hi hagi correlació amb **renterTripsTaken** i amb **age** no aporta novetat ja que la mateixa variable frequency ha estat creada en una operació entre aquestes dues. EL mateix passa amb la variable **income**, la qual és el resultat de multiplicar **frequency** per **rate.daily**. Realment l'única relació interessant d'aquest gràfic seria la que pot haver-hi entre **frequency** i **reviewCount**. No obstant es possible que la relació de causalitat lògica sigui que a més **frequency** s'esperi trobar més **reviewCount** com a conseqüencia de que el vehicle és usat per més usuaris, i no al revés.

```
# Relació entre frequency i reviewCount
ggplot(ds, aes(x=frequency, y=reviewCount)) + geom_point()
```



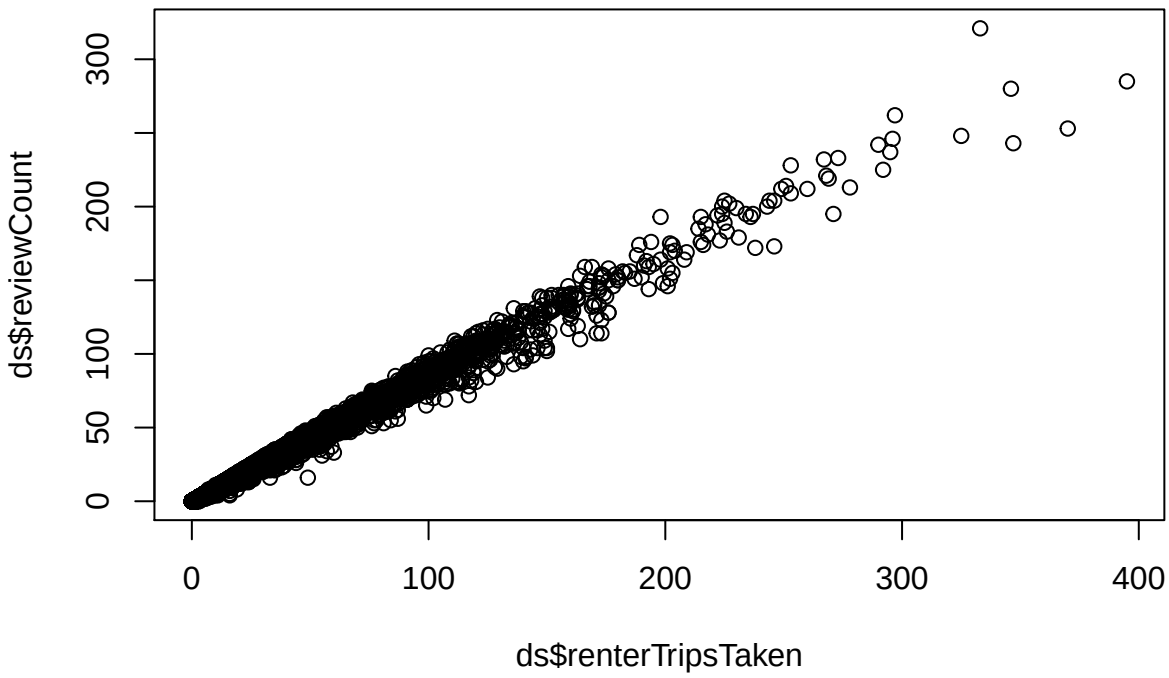
Això ens mostra un gràfic interessantissim, en que es pot distingir grups d'observacions que es comporten diferent (vaja, clusters). On hi ha grups que individualment mostres una correlació quasi perfecte entre **frequency** i **reviewCount** i la diferencia entre els grups ve donada pel pendent de la relació.

```
plot(ds$frequency,ds$reviewCount, col=as.factor(ds$age))
```



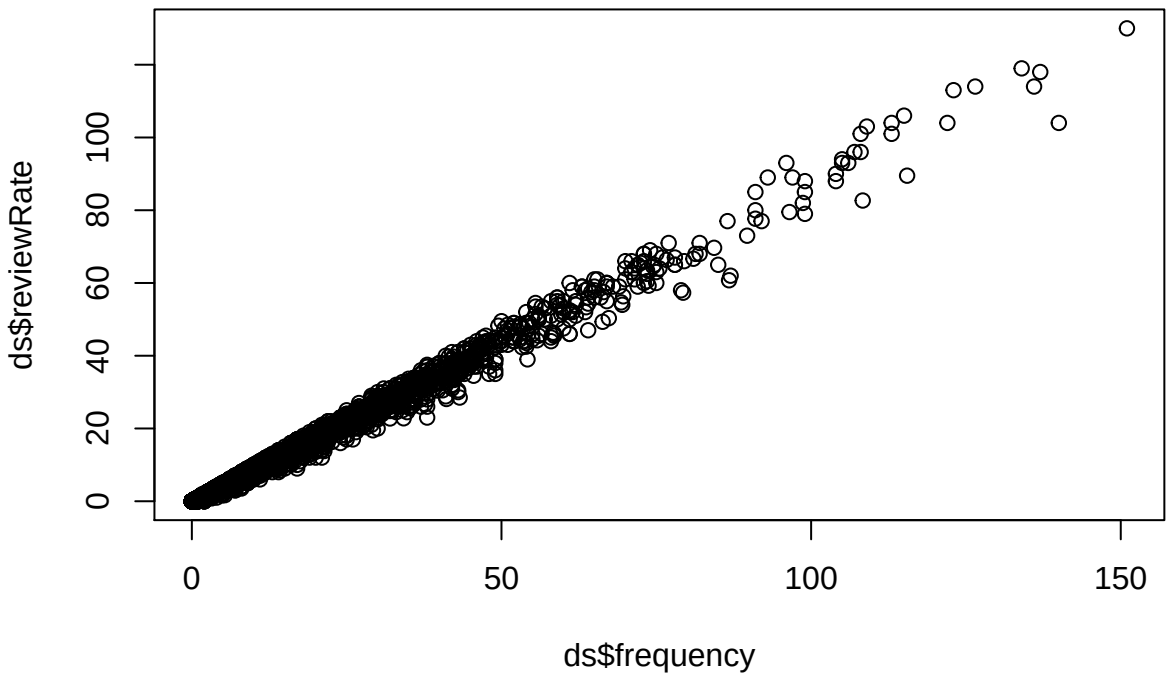
Aclarim ràpidament que la diferencia de pendents ve explicada per la variable **age**, i doncs **frequency** es una variable ja normalitzada amb **age** mentre que **reviewCount** és el valor absolut del recompte de reviews i no depen de la finestrra temporal.

```
plot(ds$renterTripsTaken,ds$reviewCount)
```



Així si mirem l'scaterplot entre **renterTripsTaken** i **reviewCount** trobem una correlació molt bona.

```
# Creem un vector reviewRate que normalitza el reviewCOunt en funció de l'edat del vehicle.
age_sense_0 <- ds$age
age_sense_0[age_sense_0 == 0] = 1
ds$reviewRate = ds$reviewCount/age_sense_0
numeric_features_names <- c(numeric_features_names,"reviewRate")

# Visualitzem la relació entre le freqüència d'us del vehicle i la freqüència de reviews.
plot(ds$frequency,ds$reviewRate)
```



Si mirem l'scaterplot de les dues variables normalitzades amb l'edat del vehicles (per tant **frequency** i freqüència de reviews) trobem que hi pot haver una correlació encara més bona.

```
# Comprovem la relació entre frequecny i reviewRate (variables normalitzade sper age)
model.lm4 <- lm(formula=frequency ~ reviewRate, data = ds)
summary(model.lm4)
```

```
##
## Call:
## lm(formula = frequency ~ reviewRate, data = ds)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10.8317  -0.4030  -0.1441   0.2369  20.5494
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.144143   0.023164   6.223 5.23e-10 ***
## reviewRate  1.147178   0.001392 824.027  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.462 on 5849 degrees of freedom
## Multiple R-squared:  0.9915, Adjusted R-squared:  0.9915
## F-statistic: 6.79e+05 on 1 and 5849 DF,  p-value: < 2.2e-16
```

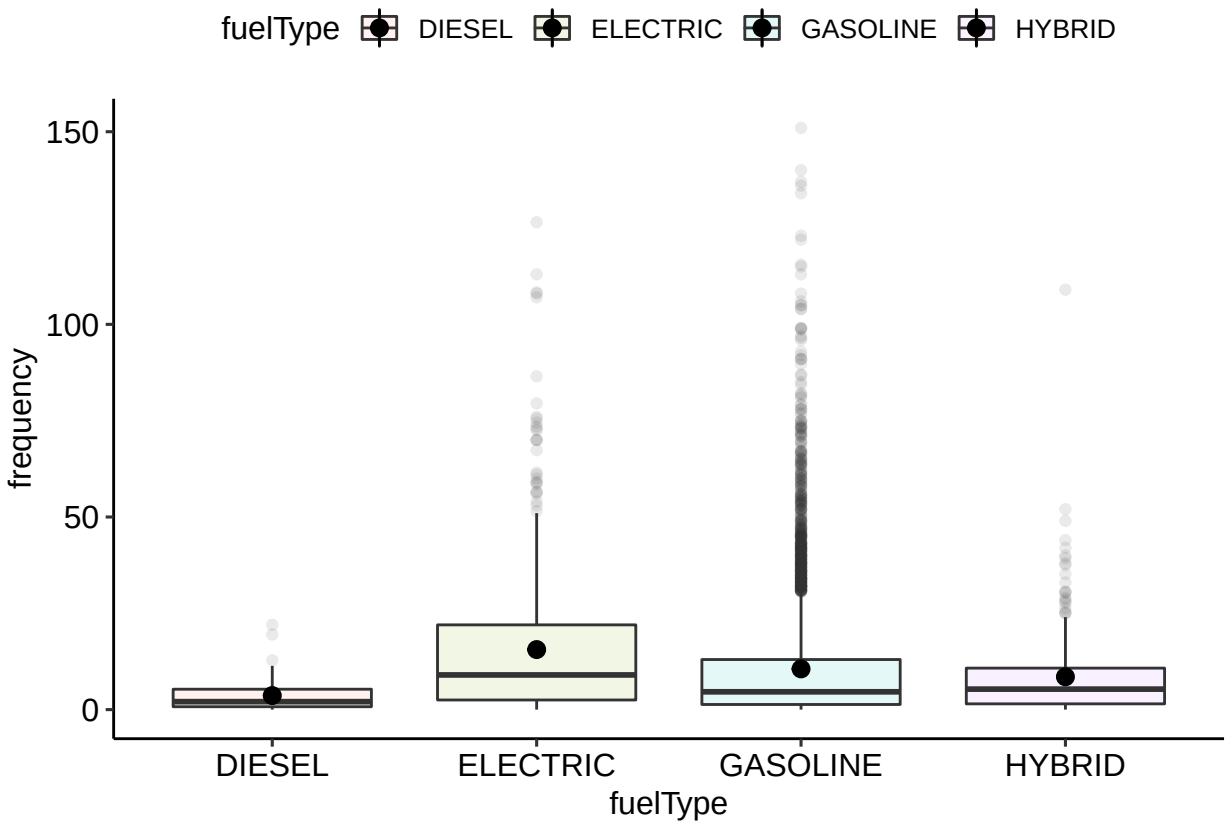
```r
# Comprovem la relació entre renterTripsTaken i reviewCount (variables sense normalitzar per age)
model.lm5 <- lm(formula=renterTripsTaken ~ reviewCount, data = ds)
summary(model.lm5)
```

```
##
## Call:
## lm(formula = renterTripsTaken ~ reviewCount, data = ds)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -47.299  -1.526   0.071   1.071  70.317
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.25672    0.07596   -3.38  0.00073 ***
## reviewCount  1.18553    0.00168  705.61  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.515 on 5849 degrees of freedom
## Multiple R-squared:  0.9884, Adjusted R-squared:  0.9884
## F-statistic: 4.979e+05 on 1 and 5849 DF,  p-value: < 2.2e-16
```

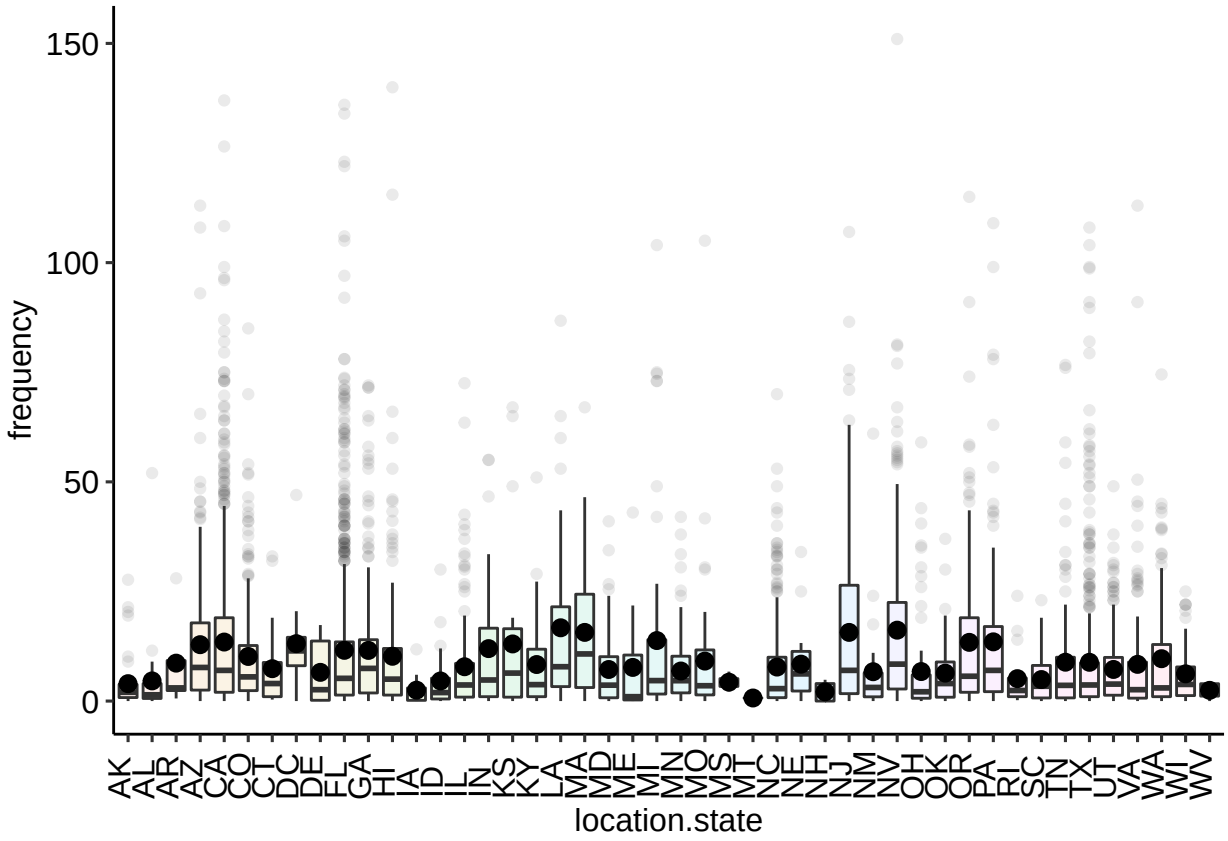
Veiem que el nombre de viatges fets i el nombre de reviews normalitzades per **age** (**frequency** i **reviewRate**) ofereixen entre elles una correlació molt bona, amb R-quadrat del 0,9915, mentre que les variables equivalents sense normalitzar amb **age** (**renterTripsTaken** i **reviewCount**) ofereixen una correlació amb un R-quadrat lleugerament inferior, del 0,9884.

Tot i amb la interessant relació entre **frequency** i **reviewRate** trobada, no hem resolt el model que expliqui **frequency** encara. Llavors exoplorem la seva relació amb les variables categoriques.

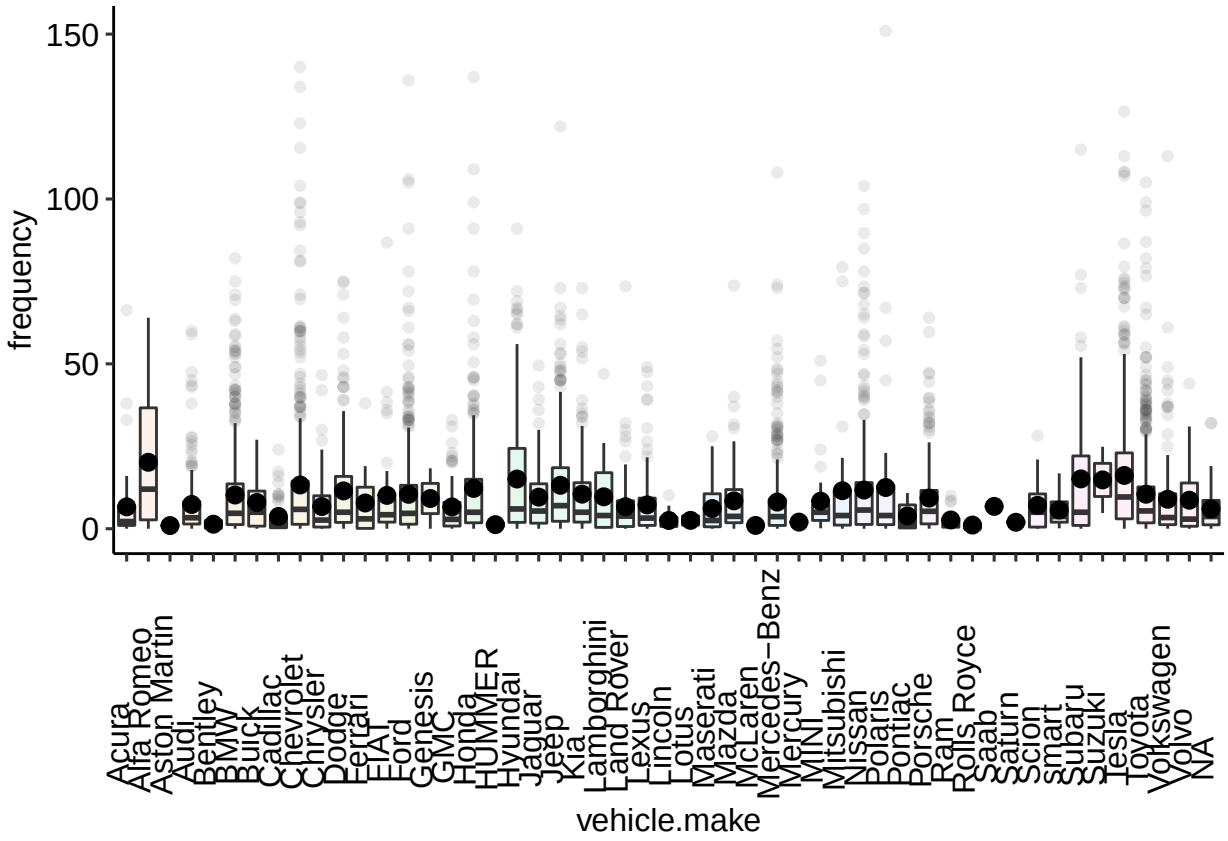
```r
# Relació entre la freqüència d'us del vehicle i el tipus de combustible
ggplot(ds, aes(x=fuelType, y=frequency, fill=fuelType)) + geom_boxplot(alpha=0.1) + stat_summary(fun.y=
```



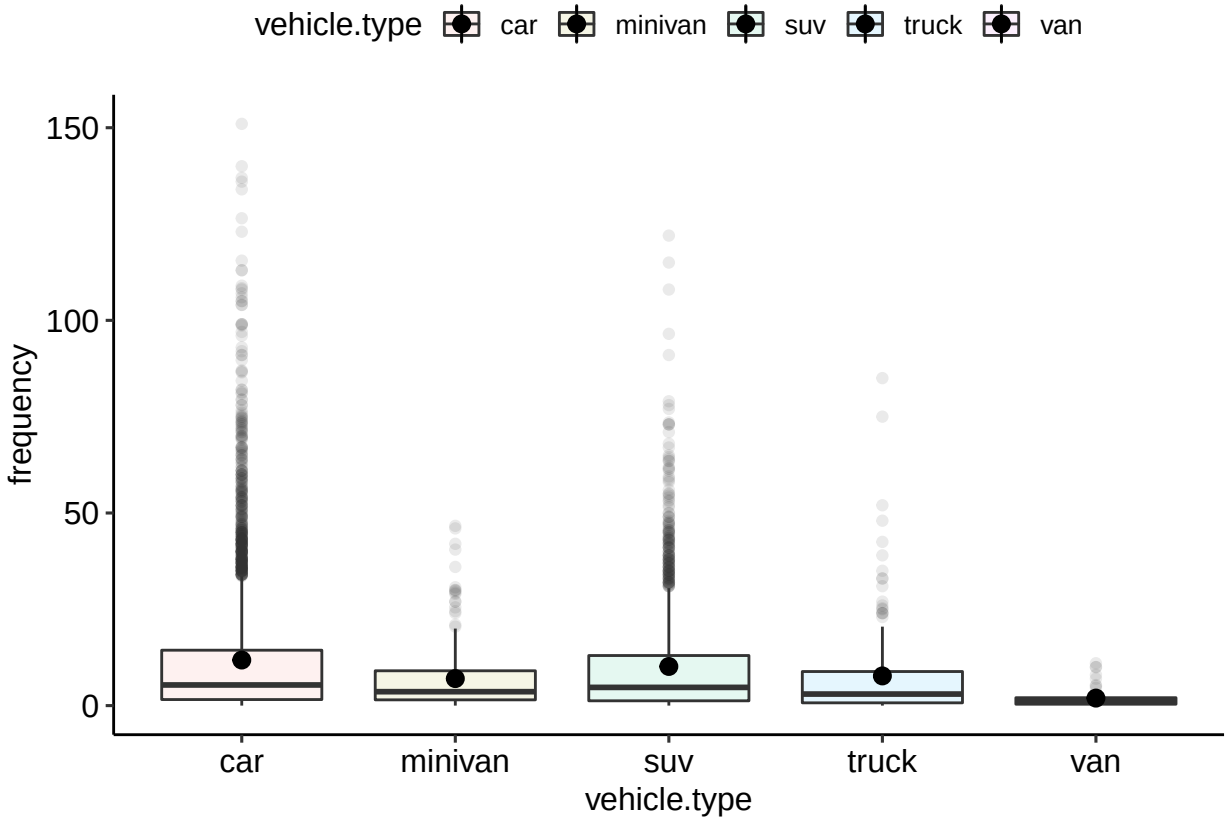

```
# Relació entre la freqüència d'us del vehicle  i l'estat on es localitza el vehicle
ggplot(ds, aes(x=location.state, y=frequency, fill=location.state)) + geom_boxplot(alpha=0.1) + stat_sum
```




```
# Relació entre la freqüència d'us del vehicle  i la marca del vehicle
ggplot(ds, aes(x=vehicle.make, y=frequency, fill=vehicle.make)) + geom_boxplot(alpha=0.1) + stat_summary
```




```
# Relació entre la freqüència d'us del vehicle i el tipus de vehicle
ggplot(ds, aes(x=vehicle.type, y=frequency, fill=vehicle.type)) + geom_boxplot(alpha=0.1) + stat_summary
```



De les anteriors visualitzacions veiem clar que:

- **frequency** depèn notablement de **fuelType**: els vehicles Eleèctrics son molt més usats que la resta, seguit dels de benzina i hibrids (independentment del nombre que n'hi ha de cada un)
- **frequency** depèn notablement de **location.state**, havent-hi diferencies notables entres estats que apreciablement afecten l'ordre de magnitud de **frequency**
- **frequency** depèn de **vehicle.make**
- **frequency** depèn en menor mesura de **vehicle.type**. Almenys es pot dir que pel tipus *van* la variable **frequency** pendrà amb seguretat valors molt més baixos que per a la resta.

Cal doncs buscar un model que pugui preveure frequency en funció de les variables categòriques observades

INICI A REVISAR

```
# Generem el model d'arbre
model_cart_frequency <- rpart(frequency ~ fuelType + location.city + location.state + vehicle.make + vel

# Mostrem un resum de la generació de l'arbre
summary(model_cart_frequency)
```

```
## Call:
## rpart(formula = frequency ~ fuelType + location.city + location.state +
##     vehicle.make + vehicle.model + vehicle.type, data = train,
##     method = "anova")
##   n= 3900
```

```
##
##            CP nsplit rel error    xerror       xstd
## 1  0.13080938      0 1.0000000 1.0004692 0.06211321
## 2  0.10869933      1 0.8691906 0.9962589 0.05961659
## 3  0.03269633      2 0.7604913 1.0004536 0.05817292
## 4  0.02621786      3 0.7277950 0.9950384 0.05650398
## 5  0.02254686      4 0.7015771 1.0048886 0.05684110
## 6  0.02204816      5 0.6790302 0.9940861 0.05625417
## 7  0.01837275      7 0.6349339 0.9985149 0.05541581
## 8  0.01784612      8 0.6165611 1.0183214 0.05549739
## 9  0.01691688      9 0.5987150 1.0159576 0.05544226
## 10 0.01234852     10 0.5817982 1.0424994 0.05647156
## 11 0.01162225     12 0.5571011 1.0563987 0.05747497
## 12 0.01114709     13 0.5454789 1.0572498 0.05765909
## 13 0.01000000     14 0.5343318 1.0605820 0.05785273
##
## Variable importance
##  location.city  vehicle.model location.state   vehicle.make       fuelType
##             41             36             12              9              3
##
## Node number 1: 3900 observations,    complexity param=0.1308094
##   mean=10.89061, MSE=259.4574
##   left son=2 (2400 obs) right son=3 (1500 obs)
##   Primary splits:
##       location.city  splits as  LLL-LRLLLLLLLLRLLLL-LLLLLL--LLLLLLLLLRLLR-RLLLL-L-LR-RLL-LLLL-LLLL-RL
##       vehicle.model  splits as  LLRLLR-LLLLLRLLLL-LLRLRL---RLLLRLLLLLRLLLR-RLLLLLLLLLLLLLLRLLRLRLL-RL
##       location.state splits as  LLLRRLLRLRRRLLLRRLRRLLRLRL-LLLRLRLLRRLLLLLLLLL, improve=0.030589680,
##       vehicle.make   splits as  LRLLLLLLRLLLLLLLRLRLRLRLLLLLLLLLLLLRRLLLLLLLLRRRLLL, improve=0.024314
##       fuelType       splits as  LRLL, improve=0.008141621, (0 missing)
##   Surrogate splits:
##       location.state splits as  LLLRRLLRLRLRLLLLLLRRLRLLLL-LLLRLRLLRRLLLLLLLLL, agree=0.721, adj=0.2
##       vehicle.model  splits as  LLRLLR-LLLLLLLRLL-LRRLRR---RLRLRLLLLLRLLRR-RLLLLLLLRRLLLLLRLLLLRRL-LL
##
## Node number 2: 2400 observations,    complexity param=0.02621786
##   mean=6.284942, MSE=90.57753
##   left son=4 (1974 obs) right son=5 (426 obs)
##   Primary splits:
##       vehicle.model  splits as  LLLLRL-LLLLLLL-LL-L-LL-----LL-L-LLRLLLLLLR-LLLLLLLRL-LLLLLRLRRLRLL-RL
##       location.city  splits as  LLL-L-LLLLLRRL-RRRL-LLLRLL--RRRLRLLLR-RL---LRRL-L-L---RL-LLRR-LRLL--L
##       vehicle.make   splits as  LRLLLLLLRLLLLLRLRLRLRRLLLLLLLLLLLLLLRLLLLLRLLLRLRRLL, improve=0.031672
##       location.state splits as  LLLLRRL-RRRRLLLLLRLLLLLRRL-RRLLLRLLLLLLRRLLLRL, improve=0.01694637,
##       fuelType       splits as  LRLL, improve=0.01444323, (0 missing)
##   Surrogate splits:
##       fuelType      splits as  LRLL, agree=0.858, adj=0.197, (0 split)
##       vehicle.make  splits as  LRLLLLLLLLLLLLRLLLLLLLLLLLLLLLLLLLLLLLLLLLLLRLLL, agree=0.854, adj=0
##       location.city splits as  LLL-R-LLRLRLLR-LLLL-LLLLLL--LLLLLLLLL-LL---LLLL-L-L---RL-LLLL-LLLL--RL
##
## Node number 3: 1500 observations,    complexity param=0.1086993
##   mean=18.25968, MSE=441.4227
##   left son=6 (942 obs) right son=7 (558 obs)
##   Primary splits:
##       vehicle.model  splits as  L-RL-L-L-LLLR-LL--LLRLRL---RLLLRL----RL-LR-LL---LLLLLLL--LRL-RLLLL--L
##       location.city  splits as  -----R--------R----------------------L--R-R--------L-L-------------L-
##       vehicle.make   splits as  LRLLLLLLRRLLLLLLL-RLRLLLLLLLL-L-LLLR-LL---LLLRRLRR, improve=0.027666
```

```
##       location.state splits as  LLRLLLLLLLLL--RRR-LRRLRRL--LR-R-LLLLL--LLLRLL-, improve=0.019036040,
##       vehicle.type   splits as  RLRRL, improve=0.009624081, (0 missing)
##   Surrogate splits:
##       vehicle.make   splits as  LRLLLLLLRLLLLRLLL-RLLLLLLLLLL-L-LLLR-LL---LLRRRLLR, agree=0.694, adj=
##       location.city  splits as  -----L--------R----------------------L--L-L--------L-R-------------R-
##       fuelType       splits as  LRLL, agree=0.641, adj=0.034, (0 split)
##       location.state splits as  LLRLLLLLLLLL--RLL-LLLLRRL--LL-L-LRLLL--LLLLLR-, agree=0.638, adj=0.0
##
## Node number 4: 1974 observations,    complexity param=0.01162225
##   mean=4.740435, MSE=41.03321
##   left son=8 (956 obs) right son=9 (1018 obs)
##   Primary splits:
##       location.city  splits as  LLR---LL-L-LR--RRRL-LLLRLL--RRLLRLLLR-RL---LRRL-L-L---LL-LRRR-LRLL---
##       vehicle.model  splits as  LLLL-L-LLRLRRL-RL-L-RL-----RR-L-LL-RLLLRL--RLLLRLL-L-LRLLR-R--L-RL---
##       vehicle.make   splits as  LLLLLLLLRLLLLR-LRLRRRRLLLLLRRLLLLLRRLLLLRLLRRRRRLL, improve=0.022999
##       location.state splits as  LLLLRRL-LLRLLLRLLRLLRLLRLR-RRLLRRLRLRLLRRLLLRL, improve=0.018493430,
##       vehicle.type   splits as  RLRLL, improve=0.003177832, (0 missing)
##   Surrogate splits:
##       location.state splits as  LLLLRRL-RLRLLLRLLRLLRLLRRL-RRLLLRLRLRLLRRRLLRL, agree=0.679, adj=0.33
##       vehicle.model  splits as  RLRL-R-RLLRRRL-RR-R-RR-----RR-R-RR-RRRLRR--RRLRLLL-L-LRRRR-R--L-RR---
##       vehicle.make   splits as  LLRRRRLLRLLRRL-LRLRLLRRLRLLRRRRLLRLRRLLLLLRRRLLRLL, agree=0.566, adj=
##       vehicle.type   splits as  RLRLL, agree=0.535, adj=0.040, (0 split)
##       fuelType       splits as  RLRR, agree=0.517, adj=0.003, (0 split)
##
## Node number 5: 426 observations,    complexity param=0.02254686
##   mean=13.44188, MSE=257.8805
##   left son=10 (222 obs) right son=11 (204 obs)
##   Primary splits:
##       location.city  splits as  ----L-L-LLLR-L-L--L---------L-R-R---L-R------R--------L-------------
##       vehicle.model  splits as  ----L-----------------------------L------R--------L-------L-LL-L---RL
##       location.state splits as  LLLLRRR-RRRRLLLLRRLLLLLRR--RR-LLRRLLLRLRRLLLRL, improve=0.062764080,
##       vehicle.make   splits as  RL-L-RLLL-L-LLLLR-LLLR-L---LR-L-L-L---L-----R-LLL-, improve=0.0448368
##       fuelType       splits as  LLRL, improve=0.005553613, (0 missing)
##   Surrogate splits:
##       location.state splits as  LLLLRRL-RRRRLLLLLRLLLLLRR--RR-LLLLLLLRLRRLLLLL, agree=0.744, adj=0.4
##       vehicle.model  splits as  ----R-----------------------------L------R--------L-------L-LL-R---RL
##       vehicle.make   splits as  LL-L-RLLR-R-LLRLL-LRRL-L---LR-R-R-R---L-----R-LRR-, agree=0.577, adj=
##       fuelType       splits as  LLRL, agree=0.540, adj=0.039, (0 split)
##       vehicle.type   splits as  L-LRL, agree=0.523, adj=0.005, (0 split)
##
## Node number 6: 942 observations,    complexity param=0.02204816
##   mean=11.66908, MSE=189.7491
##   left son=12 (385 obs) right son=13 (557 obs)
##   Primary splits:
##       vehicle.model  splits as  L--L-R-R-RRR--LL--LL-R-R----RRL-L-----L-L--RL---RLLRLRL--R-R--LRLL---
##       location.city  splits as  -----L---------------------------------L--R-L--------L----------------
##       location.state splits as  LL-LRRLRRLRL--LRR-LRRLR-L--RR-R-RLLRR--LLLRR--, improve=0.037141810,
##       vehicle.make   splits as  LRLRLRLLRRLLLRLRR-RRRRRLLLLLR-R-LLR--RL---LLL-RRRL, improve=0.025390
##       fuelType       splits as  LRRL, improve=0.008451485, (0 missing)
##   Surrogate splits:
##       vehicle.make   splits as  LRLLLRLLRRLLLRLRR-RRRRRLLLLLL-R-LLR--RL---LLL-RRRL, agree=0.725, adj=
##       location.city  splits as  -----R---------------------------------R--R-R--------L----------------
##       vehicle.type   splits as  RLRRL, agree=0.631, adj=0.096, (0 split)
##       fuelType       splits as  LRRL, agree=0.616, adj=0.060, (0 split)
```

```
##       location.state splits as  LR-RRRLRRRRL--RRR-RRRRR-L--RL-R-RLRRR--RLRRL--, agree=0.614, adj=0.05
##
## Node number 7: 558 observations,    complexity param=0.03269633
##   mean=29.38574, MSE=669.1743
##   left son=14 (485 obs) right son=15 (73 obs)
##   Primary splits:
##       location.city  splits as  -----R--------R----------------------L--L-R----------L-------------L-
##       vehicle.model  splits as  --L---------R-------L-R----L---L-----L---L----------------L--R------R
##       location.state splits as  LRLLLLL--LLL--RLR-LLR-RRR--LL-L-LLLLL--LLLLLL-, improve=0.021564770,
##       vehicle.make   splits as  LL-R-LL-RRL-RL-LL-LLLL-RR---L-L-RRLL-L------LLLRRR, improve=0.018866
##       fuelType       splits as  -LLR, improve=0.005692756, (0 missing)
##   Surrogate splits:
##       location.state splits as  LRLLLLL--LRL--RLR-LLR-RRR--LL-L-LLLLL--LLLLLL-, agree=0.898, adj=0.2
##       vehicle.model  splits as  --L---------L-------L-L----L---L-----L---L----------------L--L------R
##       vehicle.make   splits as  LL-L-LL-LLL-LL-LL-LLLL-LL---L-L-RLLL-L------LLLLLL, agree=0.871, adj=
##
## Node number 8: 956 observations
##   mean=2.221704, MSE=9.09083
##
## Node number 9: 1018 observations
##   mean=7.105767, MSE=59.47774
##
## Node number 10: 222 observations
##   mean=6.426643, MSE=40.66512
##
## Node number 11: 204 observations,    complexity param=0.01691688
##   mean=21.0761, MSE=382.4246
##   left son=22 (197 obs) right son=23 (7 obs)
##   Primary splits:
##       vehicle.model  splits as  ----L-----------------------------------R--------L-------L-LL-L---RL
##       vehicle.make   splits as  RL-L-LLLL-L---L-R-LLL--L----L-L-L-L---------R-LLL-, improve=0.119822
##       location.city  splits as  -----------R------------------R-L-----R------L----------------------
##       location.state splits as  ---LRRL-LRLL----LL--L-LRR--RR-L-LR---L-LLLR-L-, improve=0.016615250,
##       vehicle.type   splits as  R-LL-, improve=0.001537907, (0 missing)
##   Surrogate splits:
##       vehicle.make  splits as  LL-L-LLLL-L---L-L-LLL--L----L-L-L-L---------R-LLL-, agree=0.975, adj=0
##       location.city splits as  -----------L------------------L-L-----L------L----------------------
##
## Node number 12: 385 observations
##   mean=5.87672, MSE=43.09431
##
## Node number 13: 557 observations,    complexity param=0.02204816
##   mean=15.67277, MSE=251.8968
##   left son=26 (505 obs) right son=27 (52 obs)
##   Primary splits:
##       location.city  splits as  -----L-------------------------------L--R-L------------------------
##       location.state splits as  LL-LRL-LRLRL--LRL-LRRLR-L--R--R-R-LRR--LRLRR--, improve=0.047457910,
##       vehicle.model  splits as  -----R-R-RRR---------L-L----RL-------------R----L--L-R---L-R---R-----
##       fuelType       splits as  LRRL, improve=0.004255521, (0 missing)
##       vehicle.make   splits as  -L-R-R-LRRLL-L-LR-RRRLR-L---L-R---R--R------R-RLR-, improve=0.002426
##   Surrogate splits:
##       location.state splits as  LL-LLL-LLLLL--LRL-LRLLL-L--L--L-L-LLL--LLLLL--, agree=0.923, adj=0.1
##
## Node number 14: 485 observations,    complexity param=0.01837275
```

```
##   mean=26.39837, MSE=595.5184
##   left son=28 (407 obs) right son=29 (78 obs)
##   Primary splits:
##       vehicle.model  splits as  --R---------R-------L-R----L---L-----L---L---------------L--R------
##       location.city  splits as  -------------------------------------L--L------------L-------------L
##       vehicle.make   splits as  LR-R-LL-RRL-RL-LR-LLLL-RR---R-L--RLL-L------RLLRRL, improve=0.021395
##       location.state splits as  L-RRLLR--LLR--RL--RL--L----LR-R-RLRLR--RLLLRL-, improve=0.014917250,
##       fuelType       splits as  -LLR, improve=0.007332867, (0 missing)
##   Surrogate splits:
##       location.city  splits as  -------------------------------------L--L------------L-------------L
##       vehicle.make   splits as  LR-L-LL-LLL-RL-LL-LLLL-RL---L-L--LLL-L------LLLLRL, agree=0.860, adj=
##       location.state splits as  L-LLLLL--LLL--LL--LL--L----LL-L-LLRLL--LLLLLL-, agree=0.841, adj=0.0
##       vehicle.type   splits as  LLLR-, agree=0.841, adj=0.013, (0 split)
##
## Node number 15: 73 observations,    complexity param=0.01114709
##   mean=49.2333, MSE=705.3146
##   left son=30 (63 obs) right son=31 (10 obs)
##   Primary splits:
##       vehicle.model  splits as  --------------------L-L------------------L----------------L--L------
##       vehicle.make   splits as  -----L--LLL--L--L-L-LL----------LRRL-L--------LR-L, improve=0.168325
##       location.city  splits as  -----R--------R---------------------------R-------------------------
##       location.state splits as  -R-LR----LL---LRL---L-RLR-----R-----L---L-LL--, improve=0.058678150,
##       fuelType       splits as  -LRR, improve=0.009211748, (0 missing)
##   Surrogate splits:
##       vehicle.make   splits as  -----L--LLL--L--L-L-LL----------LRRL-L--------LR-L, agree=0.918, adj=
##       location.city  splits as  -----L--------R---------------------------L-------------------------
##       location.state splits as  -L-LL----LL---LLL---L-LLR-----L-----L---L-LL--, agree=0.877, adj=0.1
##
## Node number 22: 197 observations
##   mean=19.34937, MSE=263.0334
##
## Node number 23: 7 observations
##   mean=69.67143, MSE=1297.019
##
## Node number 26: 505 observations
##   mean=13.62088, MSE=184.6053
##
## Node number 27: 52 observations
##   mean=35.59977, MSE=467.4282
##
## Node number 28: 407 observations,    complexity param=0.01234852
##   mean=23.68798, MSE=517.9797
##   left son=56 (232 obs) right son=57 (175 obs)
##   Primary splits:
##       location.city  splits as  -------------------------------------L--L------------L-------------L
##       location.state splits as  L-RRLLR--LLR--RL--RL--L----LR-R-RL-LL--RLLLRL-, improve=0.028928930,
##       vehicle.model  splits as  --------------------L------R---R-----R---L----------------L----------
##       vehicle.make   splits as  LR-R-LL-RRL--L-LL-LRLL--R---R-R--RLL-L------RRRRRL, improve=0.024272
##       fuelType       splits as  -RLR, improve=0.007795553, (0 missing)
##   Surrogate splits:
##       location.state splits as  R-RRLLR--LLR--RR--RL--L----LR-R-RL-LL--RLLLRL-, agree=0.838, adj=0.6
##       vehicle.model  splits as  --------------------R------R---L-----R---L----------------L----------
##       vehicle.make   splits as  LR-L-LL-LRR--L-LL-LRLL--R---L-L--RRL-R------LRLLLL, agree=0.649, adj=
##       vehicle.type   splits as  LRLL-, agree=0.572, adj=0.006, (0 split)
```

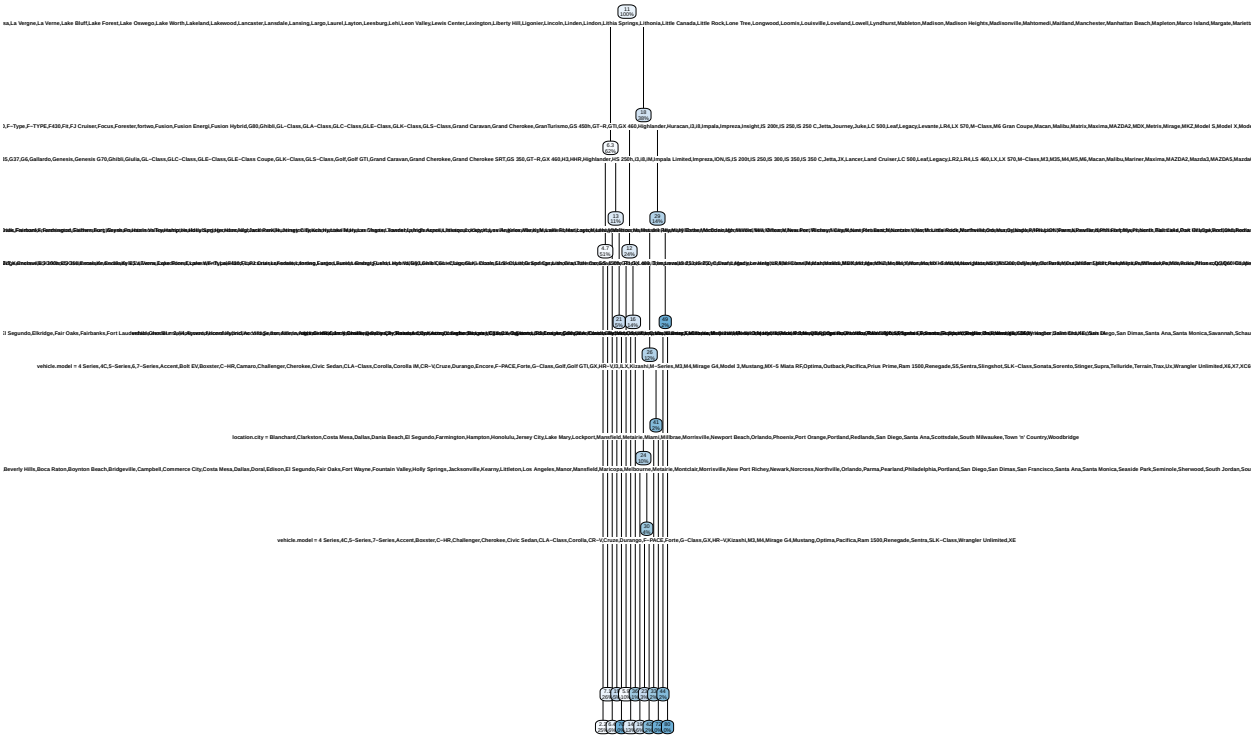
```
##
## Node number 29: 78 observations,    complexity param=0.01784612
##   mean=40.54105, MSE=761.7639
##   left son=58 (63 obs) right son=59 (15 obs)
##   Primary splits:
##       location.city  splits as  ------------------------------------------------------------------------
##       location.state splits as  ---LLRL--LLL--L---LL-------L--R-R-LLR---L-L-L-, improve=0.120090200,
##       vehicle.model  splits as  --L---------L---------R--------------------------------------R------
##       vehicle.make   splits as  -L-R-L--L---LL--R-L-L--LR---------R--L----------L-, improve=0.049407
##       vehicle.type   splits as  L-RR-, improve=0.004099288, (0 missing)
##   Surrogate splits:
##       location.state splits as  ---LLRL--LLL--L---LL-------L--L-R-LLR---L-L-L-, agree=0.859, adj=0.2
##       vehicle.model  splits as  --L---------R---------L--------------------------------------L------
##
## Node number 30: 63 observations
##   mean=44.28092, MSE=497.6561
##
## Node number 31: 10 observations
##   mean=80.43333, MSE=885.6067
##
## Node number 56: 232 observations
##   mean=19.0894, MSE=344.7987
##
## Node number 57: 175 observations,    complexity param=0.01234852
##   mean=29.78439, MSE=682.3672
##   left son=114 (113 obs) right son=115 (62 obs)
##   Primary splits:
##       vehicle.model  splits as  --------------------L------L---L-----R---L----------------L----------
##       vehicle.make   splits as  RL-R-L--RLL--L--L-RLLR--L---R-L--LLR-L------RLRL--, improve=0.097393
##       fuelType       splits as  -RL-, improve=0.019531010, (0 missing)
##       location.city  splits as  ------------------------------------------------------------------------
##       location.state splits as  L-LRLRR--L-R--RL--RR-------RR-R-L------RL--L--, improve=0.007088564,
##   Surrogate splits:
##       vehicle.make   splits as  RL-R-L--RLL--L--L-RLLL--L---R-L--LLR-L------RLRL--, agree=0.954, adj
##       fuelType       splits as  -RL-, agree=0.846, adj=0.565, (0 split)
##       location.city  splits as  ------------------------------------------------------------------------
##       location.state splits as  R-LLLRL--L-L--RR--RR-------RR-L-L------LR--L--, agree=0.720, adj=0.2
##
## Node number 58: 63 observations
##   mean=33.11658, MSE=337.1433
##
## Node number 59: 15 observations
##   mean=71.72381, MSE=1341.29
##
## Node number 114: 113 observations
##   mean=23.25921, MSE=290.8227
##
## Node number 115: 62 observations
##   mean=41.67707, MSE=1176.951
```

```
printcp(model_cart_frequency)
```

```
##
## Regression tree:
## rpart(formula = frequency ~ fuelType + location.city + location.state +
```

```
##     vehicle.make + vehicle.model + vehicle.type, data = train,
##     method = "anova")
##
## Variables actually used in tree construction:
## [1] location.city vehicle.model
##
## Root node error: 1011884/3900 = 259.46
##
## n= 3900
##
##          CP nsplit rel error  xerror     xstd
## 1  0.130809      0   1.00000 1.00047 0.062113
## 2  0.108699      1   0.86919 0.99626 0.059617
## 3  0.032696      2   0.76049 1.00045 0.058173
## 4  0.026218      3   0.72779 0.99504 0.056504
## 5  0.022547      4   0.70158 1.00489 0.056841
## 6  0.022048      5   0.67903 0.99409 0.056254
## 7  0.018373      7   0.63493 0.99851 0.055416
## 8  0.017846      8   0.61656 1.01832 0.055497
## 9  0.016917      9   0.59872 1.01596 0.055442
## 10 0.012349     10   0.58180 1.04250 0.056472
## 11 0.011622     12   0.55710 1.05640 0.057475
## 12 0.011147     13   0.54548 1.05725 0.057659
## 13 0.010000     14   0.53433 1.06058 0.057853
```

```r
# Mostrem en un gràfic l'arbre obtingut
rpart.plot(model_cart_frequency)
```



Provem l'ús de *desicion tree*

En primer lloc, realitzem una barreja de les dades per si en alguna de les variables hi hagués cert ordre.

```
set.seed(1)
data_random <- ds[sample(nrow(ds)),]
```

Agafem com a **y** la vairable **frequency**, i les categoriques com a **X**.

```
set.seed(666)
y <- data_random["frequency"]
X <- data_random[c("fuelType","location.state","vehicle.make","vehicle.type")]
```

```
# Triem un subconjunt per a l'entrenament del model i un subconjunt per a provar-lo
# El subconjunt d'entenament tindra 2/3 de les observacions: per tant 3901
# El subconjunt de prova tindra 1/3 de les observacion: per tant 1950
trainX <- X[1:3901,]
trainy <- y[1:3901,]
testX  <- X[3902:5851,]
testy  <- y[3902:5851,]
```

Creem el model, en visualitzem la qualitat i n'extraiem les regles

```
trainy = as.factor(trainy)
model <- C50::C5.0(trainX, trainy, rules=TRUE )
summary(model)
```

```
##
## Call:
```

```
## C5.0.default(x = trainX, y = trainy, rules = TRUE)
##
##
## C5.0 [Release 2.07 GPL Edition]      Wed Dec 23 19:52:47 2020
## -------------------------------
##
## Class specified by attribute `outcome'
##
## Read 3901 cases (5 attributes) from undefined.data
##
## Rules:
##
## Rule 1: (1, lift 9.3)
##  fuelType = ELECTRIC
##  location.state = DE
##  ->  class 0  [0.667]
##
## Rule 2: (1, lift 9.3)
##  fuelType = HYBRID
##  location.state = MO
##  vehicle.make = Ford
##  ->  class 0  [0.667]
##
## Rule 3: (1, lift 9.3)
##  location.state = MO
##  vehicle.make = Toyota
##  vehicle.type = truck
##  ->  class 0  [0.667]
##
## Rule 4: (1, lift 9.3)
##  fuelType = ELECTRIC
##  location.state = NV
##  vehicle.make = BMW
##  ->  class 0  [0.667]
##
## Rule 5: (1, lift 9.3)
##  location.state = VA
##  vehicle.make = Mazda
##  vehicle.type = minivan
##  ->  class 0  [0.667]
##
## Rule 6: (1, lift 9.3)
##  location.state = WI
##  vehicle.type = minivan
##  ->  class 0  [0.667]
##
## Rule 7: (6/3, lift 7.0)
##  fuelType = HYBRID
##  location.state = CA
##  vehicle.make = Ford
##  ->  class 0  [0.500]
##
## Rule 8: (14/7, lift 7.0)
##  location.state = TN
```

```
##  vehicle.make in {BMW, Buick, Dodge, Volkswagen}
##  ->  class 0  [0.500]
##
## Rule 9: (9/6, lift 5.1)
##  location.state = UT
##  vehicle.make = Nissan
##  ->  class 0  [0.364]
##
## Rule 10: (273/240, lift 1.7)
##  vehicle.make = Mercedes-Benz
##  ->  class 0  [0.124]
##
## Rule 11: (428/379, lift 1.6)
##  location.state in {AK, NH, OH, TX}
##  ->  class 0  [0.116]
##
## Rule 12: (1298/1179, lift 1.3)
##  vehicle.type in {suv, truck, van}
##  ->  class 0  [0.092]
##
## Rule 13: (1, lift 2600.7)
##  location.state = NV
##  vehicle.make = Ferrari
##  ->  class 0.0769230769230769  [0.667]
##
## Rule 14: (1, lift 371.5)
##  location.state = HI
##  vehicle.make = HUMMER
##  ->  class 0.0833333333333333  [0.667]
##
## Rule 15: (2/1, lift 278.6)
##  location.state = NC
##  vehicle.make = Pontiac
##  ->  class 0.0833333333333333  [0.500]
##
## Rule 16: (4/3, lift 185.8)
##  location.state = FL
##  vehicle.make = BMW
##  vehicle.type = suv
##  ->  class 0.0833333333333333  [0.333]
##
## Rule 17: (8/7, lift 111.5)
##  location.state = FL
##  vehicle.make = Honda
##  ->  class 0.0833333333333333  [0.200]
##
## Rule 18: (3/2, lift 520.1)
##  location.state = WA
##  vehicle.make = Mercedes-Benz
##  vehicle.type = suv
##  ->  class 0.0909090909090909  [0.400]
##
## Rule 19: (4/3, lift 433.4)
##  location.state = CO
```

```
##  vehicle.make = Jeep
##  ->  class 0.0909090909090909  [0.333]
##
## Rule 20: (5/4, lift 371.5)
##  location.state = IL
##  vehicle.make = Lexus
##  ->  class 0.0909090909090909  [0.286]
##
## Rule 21: (1, lift 1300.3)
##  fuelType = HYBRID
##  location.state = MN
##  vehicle.make = Toyota
##  ->  class 0.1  [0.667]
##
## Rule 22: (1, lift 433.4)
##  location.state = FL
##  vehicle.make = Rolls Royce
##  ->  class 0.111111111111111  [0.667]
##
## Rule 23: (2/1, lift 325.1)
##  location.state = OR
##  vehicle.make = Toyota
##  vehicle.type = truck
##  ->  class 0.111111111111111  [0.500]
##
## Rule 24: (2/1, lift 325.1)
##  location.state = TN
##  vehicle.make = Toyota
##  vehicle.type = suv
##  ->  class 0.111111111111111  [0.500]
##
## Rule 25: (3/2, lift 260.1)
##  location.state = IL
##  vehicle.make = Kia
##  ->  class 0.111111111111111  [0.400]
##
## Rule 26: (2/1, lift 216.7)
##  location.state = HI
##  vehicle.make = FIAT
##  ->  class 0.125  [0.500]
##
## Rule 27: (4/3, lift 144.5)
##  location.state = PA
##  vehicle.make = BMW
##  ->  class 0.125  [0.333]
##
## Rule 28: (5/4, lift 123.8)
##  location.state = OR
##  vehicle.make = Porsche
##  ->  class 0.125  [0.286]
##
## Rule 29: (12/10, lift 92.9)
##  location.state = CA
##  vehicle.make = Hyundai
```

```
##  ->  class 0.125  [0.214]
##
## Rule 30: (1, lift 185.8)
##  location.state = NJ
##  vehicle.make = Bentley
##  ->  class 0.142857142857143  [0.667]
##
## Rule 31: (1, lift 185.8)
##  location.state = OK
##  vehicle.make = Chrysler
##  ->  class 0.142857142857143  [0.667]
##
## Rule 32: (2/1, lift 139.3)
##  location.state = VA
##  vehicle.make = Porsche
##  ->  class 0.142857142857143  [0.500]
##
## Rule 33: (4/3, lift 92.9)
##  location.state = NV
##  vehicle.make = Jaguar
##  ->  class 0.142857142857143  [0.333]
##
## Rule 34: (4/3, lift 92.9)
##  location.state = PA
##  vehicle.make = Toyota
##  ->  class 0.142857142857143  [0.333]
##
## Rule 35: (4/3, lift 92.9)
##  location.state = SC
##  vehicle.make = Audi
##  ->  class 0.142857142857143  [0.333]
##
## Rule 36: (11/10, lift 42.9)
##  location.state = NJ
##  vehicle.make = Porsche
##  ->  class 0.142857142857143  [0.154]
##
## Rule 37: (2, lift 112.5)
##  location.state = IA
##  vehicle.make in {Dodge, GMC}
##  ->  class 0.166666666666667  [0.750]
##
## Rule 38: (1, lift 100.0)
##  location.state = NV
##  vehicle.make = Lamborghini
##  ->  class 0.166666666666667  [0.667]
##
## Rule 39: (2/1, lift 75.0)
##  location.state = ID
##  vehicle.make = Ford
##  ->  class 0.166666666666667  [0.500]
##
## Rule 40: (2/1, lift 75.0)
##  location.state = IL
```

```
##  vehicle.make = Mercedes-Benz
##  vehicle.type = car
##  ->  class 0.166666666666667  [0.500]
##
## Rule 41: (2/1, lift 75.0)
##  location.state = OK
##  vehicle.make = Chevrolet
##  vehicle.type = suv
##  ->  class 0.166666666666667  [0.500]
##
## Rule 42: (4/3, lift 50.0)
##  location.state = CA
##  vehicle.make = Audi
##  vehicle.type = suv
##  ->  class 0.166666666666667  [0.333]
##
## Rule 43: (4/3, lift 50.0)
##  fuelType = GASOLINE
##  location.state = DE
##  ->  class 0.166666666666667  [0.333]
##
## Rule 44: (5/4, lift 42.9)
##  location.state = UT
##  vehicle.make = Dodge
##  ->  class 0.166666666666667  [0.286]
##
## Rule 45: (6/5, lift 37.5)
##  location.state = CA
##  vehicle.make = Honda
##  vehicle.type = minivan
##  ->  class 0.166666666666667  [0.250]
##
## Rule 46: (6/5, lift 37.5)
##  location.state = UT
##  vehicle.make = Toyota
##  vehicle.type = truck
##  ->  class 0.166666666666667  [0.250]
##
## Rule 47: (14/12, lift 28.1)
##  location.state = AZ
##  vehicle.make = BMW
##  ->  class 0.166666666666667  [0.188]
##
## Rule 48: (1, lift 866.9)
##  location.state = CA
##  vehicle.make = Hyundai
##  vehicle.type = suv
##  ->  class 0.181818181818182  [0.667]
##
## Rule 49: (3/2, lift 520.1)
##  location.state = HI
##  vehicle.make = Lexus
##  ->  class 0.181818181818182  [0.400]
##
```

```
## Rule 50: (4/3, lift 433.4)
##  location.state = WA
##  vehicle.make = Subaru
##  ->  class 0.181818181818182  [0.333]
##
## Rule 51: (1, lift 113.1)
##  location.state = GA
##  vehicle.make = Cadillac
##  ->  class 0.2  [0.667]
##
## Rule 52: (1, lift 113.1)
##  location.state = HI
##  vehicle.make = GMC
##  ->  class 0.2  [0.667]
##
## Rule 53: (2/1, lift 84.8)
##  location.state = UT
##  vehicle.make = Dodge
##  vehicle.type = car
##  ->  class 0.2  [0.500]
##
## Rule 54: (3/2, lift 67.8)
##  location.state = NJ
##  vehicle.make = Volkswagen
##  ->  class 0.2  [0.400]
##
## Rule 55: (6/5, lift 42.4)
##  location.state = NJ
##  vehicle.make = Mercedes-Benz
##  vehicle.type = suv
##  ->  class 0.2  [0.250]
##
## Rule 56: (8/7, lift 33.9)
##  location.state = FL
##  vehicle.make = Polaris
##  ->  class 0.2  [0.200]
##
## Rule 57: (86/84, lift 5.8)
##  vehicle.make = Lexus
##  ->  class 0.2  [0.034]
##
## Rule 58: (2/1, lift 975.2)
##  location.state = ID
##  vehicle.make = BMW
##  ->  class 0.222222222222222  [0.500]
##
## Rule 59: (8/7, lift 390.1)
##  location.state = PA
##  vehicle.make = Volkswagen
##  ->  class 0.222222222222222  [0.200]
##
## Rule 60: (1, lift 2600.7)
##  location.state = CA
##  vehicle.make = Pontiac
```

```
##  ->  class 0.230769230769231  [0.667]
##
## Rule 61: (2, lift 91.4)
##  location.state = PA
##  vehicle.make in {Chrysler, Mitsubishi}
##  ->  class 0.25  [0.750]
##
## Rule 62: (1, lift 81.3)
##  location.state = ME
##  vehicle.make = Jeep
##  ->  class 0.25  [0.667]
##
## Rule 63: (1, lift 81.3)
##  location.state = OK
##  vehicle.make = Mitsubishi
##  ->  class 0.25  [0.667]
##
## Rule 64: (1, lift 81.3)
##  location.state = TN
##  vehicle.make = Porsche
##  ->  class 0.25  [0.667]
##
## Rule 65: (2/1, lift 61.0)
##  location.state = ID
##  vehicle.make = Dodge
##  ->  class 0.25  [0.500]
##
## Rule 66: (3/2, lift 48.8)
##  fuelType = HYBRID
##  location.state = FL
##  vehicle.make = Ford
##  ->  class 0.25  [0.400]
##
## Rule 67: (4/3, lift 40.6)
##  location.state = AZ
##  vehicle.make = Ford
##  vehicle.type = truck
##  ->  class 0.25  [0.333]
##
## Rule 68: (4/3, lift 40.6)
##  location.state = FL
##  vehicle.make = Tesla
##  vehicle.type = suv
##  ->  class 0.25  [0.333]
##
## Rule 69: (4/3, lift 40.6)
##  location.state = WA
##  vehicle.make = Tesla
##  vehicle.type = suv
##  ->  class 0.25  [0.333]
##
## Rule 70: (5/4, lift 34.8)
##  location.state = GA
##  vehicle.make = Maserati
```

```
##  ->  class 0.25  [0.286]
##
## Rule 71: (5/4, lift 34.8)
##  location.state = NC
##  vehicle.make = Toyota
##  vehicle.type = car
##  ->  class 0.25  [0.286]
##
## Rule 72: (7/6, lift 27.1)
##  location.state = NC
##  vehicle.make = Hyundai
##  ->  class 0.25  [0.222]
##
## Rule 73: (291/286, lift 2.5)
##  vehicle.make = Ford
##  ->  class 0.25  [0.020]
##
## Rule 74: (1, lift 371.5)
##  location.state = AZ
##  vehicle.make = Ferrari
##  ->  class 0.285714285714286  [0.667]
##
## Rule 75: (1, lift 371.5)
##  location.state = KS
##  vehicle.make = Bentley
##  ->  class 0.285714285714286  [0.667]
##
## Rule 76: (1, lift 371.5)
##  location.state = NJ
##  vehicle.make = Chrysler
##  ->  class 0.285714285714286  [0.667]
##
## Rule 77: (7/6, lift 123.8)
##  location.state = MD
##  vehicle.make = Toyota
##  ->  class 0.285714285714286  [0.222]
##
## Rule 78: (2/1, lift 650.2)
##  location.state = NC
##  vehicle.make = Audi
##  ->  class 0.3  [0.500]
##
## Rule 79: (2/1, lift 650.2)
##  location.state = TN
##  vehicle.make = Lexus
##  ->  class 0.3  [0.500]
##
## Rule 80: (4/3, lift 433.4)
##  location.state = MN
##  vehicle.make = Honda
##  ->  class 0.3  [0.333]
##
## Rule 81: (1, lift 2600.7)
##  location.state = HI
```

```
##  vehicle.make = Lamborghini
##  ->  class 0.3125  [0.667]
##
## Rule 82: (1, lift 61.9)
##  location.state = FL
##  vehicle.make = Maserati
##  vehicle.type = suv
##  ->  class 0.333333333333333  [0.667]
##
## Rule 83: (1, lift 61.9)
##  fuelType = ELECTRIC
##  vehicle.make = FIAT
##  ->  class 0.333333333333333  [0.667]
##
## Rule 84: (1, lift 61.9)
##  location.state = IL
##  vehicle.make = Subaru
##  ->  class 0.333333333333333  [0.667]
##
## Rule 85: (1, lift 61.9)
##  location.state = OK
##  vehicle.make = Cadillac
##  ->  class 0.333333333333333  [0.667]
##
## Rule 86: (1, lift 61.9)
##  location.state = TN
##  vehicle.make = Polaris
##  ->  class 0.333333333333333  [0.667]
##
## Rule 87: (2/1, lift 46.4)
##  location.state = NJ
##  vehicle.make = Maserati
##  ->  class 0.333333333333333  [0.500]
##
## Rule 88: (2/1, lift 46.4)
##  location.state = UT
##  vehicle.make = Land Rover
##  ->  class 0.333333333333333  [0.500]
##
## Rule 89: (2/1, lift 46.4)
##  location.state = WA
##  vehicle.make = BMW
##  ->  class 0.333333333333333  [0.500]
##
## Rule 90: (3/2, lift 37.2)
##  location.state = NC
##  vehicle.make = GMC
##  vehicle.type = suv
##  ->  class 0.333333333333333  [0.400]
##
## Rule 91: (3/2, lift 37.2)
##  location.state = NV
##  vehicle.make = Volkswagen
##  ->  class 0.333333333333333  [0.400]
```

```
##
## Rule 92: (4/3, lift 31.0)
##  location.state = NJ
##  vehicle.make = Ford
##  ->  class 0.333333333333333  [0.333]
##
## Rule 93: (4/3, lift 31.0)
##  location.state = NV
##  vehicle.make = Cadillac
##  ->  class 0.333333333333333  [0.333]
##
## Rule 94: (4/3, lift 31.0)
##  location.state = PA
##  vehicle.make = Jeep
##  ->  class 0.333333333333333  [0.333]
##
## Rule 95: (8/6, lift 27.9)
##  location.state = NC
##  vehicle.make = Mercedes-Benz
##  ->  class 0.333333333333333  [0.300]
##
## Rule 96: (5/4, lift 26.5)
##  location.state = UT
##  vehicle.make = Chevrolet
##  ->  class 0.333333333333333  [0.286]
##
## Rule 97: (9/7, lift 25.3)
##  location.state = NC
##  vehicle.make = BMW
##  ->  class 0.333333333333333  [0.273]
##
## Rule 98: (9/7, lift 25.3)
##  location.state = VA
##  vehicle.make = Nissan
##  ->  class 0.333333333333333  [0.273]
##
## Rule 99: (13/11, lift 18.6)
##  fuelType = GASOLINE
##  location.state = CT
##  ->  class 0.333333333333333  [0.200]
##
## Rule 100: (15/13, lift 16.4)
##  location.state = CT
##  ->  class 0.333333333333333  [0.176]
##
## Rule 101: (2/1, lift 487.6)
##  location.state = IL
##  vehicle.make = Porsche
##  ->  class 0.375  [0.500]
##
## Rule 102: (2/1, lift 487.6)
##  location.state = UT
##  vehicle.make = Toyota
##  vehicle.type = minivan
```

```
##  ->  class 0.375  [0.500]
##
## Rule 103: (13/12, lift 520.1)
##  location.state = HI
##  vehicle.make = Toyota
##  ->  class 0.384615384615385  [0.133]
##
## Rule 104: (1, lift 173.4)
##  location.state = FL
##  vehicle.make = Ford
##  vehicle.type = van
##  ->  class 0.4  [0.667]
##
## Rule 105: (1, lift 173.4)
##  fuelType = ELECTRIC
##  location.state = HI
##  vehicle.make = Nissan
##  ->  class 0.4  [0.667]
##
## Rule 106: (1, lift 173.4)
##  location.state = NC
##  vehicle.make = Maserati
##  ->  class 0.4  [0.667]
##
## Rule 107: (2/1, lift 130.0)
##  location.state = HI
##  vehicle.make = Acura
##  ->  class 0.4  [0.500]
##
## Rule 108: (2/1, lift 130.0)
##  location.state = ID
##  vehicle.make = Mercedes-Benz
##  ->  class 0.4  [0.500]
##
## Rule 109: (6/5, lift 65.0)
##  location.state = NV
##  vehicle.make = Polaris
##  ->  class 0.4  [0.250]
##
## Rule 110: (7/6, lift 57.8)
##  location.state = IL
##  vehicle.make = Toyota
##  ->  class 0.4  [0.222]
##
## Rule 111: (8/7, lift 52.0)
##  location.state = UT
##  vehicle.make = Audi
##  ->  class 0.4  [0.200]
##
## Rule 112: (4/3, lift 433.4)
##  location.state = MN
##  vehicle.make = Dodge
##  ->  class 0.416666666666667  [0.333]
##
```

```
## Rule 113: (1, lift 325.1)
##  location.state = TN
##  vehicle.make = Land Rover
##  ->  class 0.428571428571429  [0.667]
##
## Rule 114: (2/1, lift 243.8)
##  location.state = HI
##  vehicle.make = Toyota
##  vehicle.type = suv
##  ->  class 0.428571428571429  [0.500]
##
## Rule 115: (3/2, lift 195.1)
##  location.state = UT
##  vehicle.make = Chrysler
##  ->  class 0.428571428571429  [0.400]
##
## Rule 116: (5/4, lift 139.3)
##  location.state = MN
##  vehicle.make = Chevrolet
##  ->  class 0.428571428571429  [0.286]
##
## Rule 117: (7/6, lift 108.4)
##  location.state = GA
##  vehicle.make = Audi
##  ->  class 0.428571428571429  [0.222]
##
## Rule 118: (9/8, lift 141.9)
##  location.state = CO
##  vehicle.make = Chevrolet
##  ->  class 0.444444444444444  [0.182]
##
## Rule 119: (9/8, lift 141.9)
##  location.state = CA
##  vehicle.make = Nissan
##  ->  class 0.444444444444444  [0.182]
##
## Rule 120: (11/10, lift 120.0)
##  location.state = FL
##  vehicle.make = Kia
##  vehicle.type = car
##  ->  class 0.444444444444444  [0.154]
##
## Rule 121: (1, lift 47.3)
##  location.state = FL
##  vehicle.make = Scion
##  ->  class 0.5  [0.667]
##
## Rule 122: (1, lift 47.3)
##  fuelType = DIESEL
##  location.state = GA
##  vehicle.make = Land Rover
##  ->  class 0.5  [0.667]
##
## Rule 123: (1, lift 47.3)
```

```
##  location.state = ID
##  vehicle.make = Alfa Romeo
##  ->  class 0.5  [0.667]
##
## Rule 124: (1, lift 47.3)
##  fuelType = HYBRID
##  location.state = IL
##  vehicle.make = Lexus
##  ->  class 0.5  [0.667]
##
## Rule 125: (1, lift 47.3)
##  location.state = KS
##  vehicle.make = BMW
##  ->  class 0.5  [0.667]
##
## Rule 126: (1, lift 47.3)
##  location.state = OK
##  vehicle.make = Toyota
##  ->  class 0.5  [0.667]
##
## Rule 127: (1, lift 47.3)
##  location.state = TN
##  vehicle.make = GMC
##  ->  class 0.5  [0.667]
##
## Rule 128: (1, lift 47.3)
##  location.state = WI
##  vehicle.make = Scion
##  ->  class 0.5  [0.667]
##
## Rule 129: (2/1, lift 35.5)
##  location.state = GA
##  vehicle.make = Dodge
##  vehicle.type = minivan
##  ->  class 0.5  [0.500]
##
## Rule 130: (2/1, lift 35.5)
##  location.state = TN
##  vehicle.make = Chevrolet
##  vehicle.type = suv
##  ->  class 0.5  [0.500]
##
## Rule 131: (3/2, lift 28.4)
##  location.state = CO
##  vehicle.make = Porsche
##  ->  class 0.5  [0.400]
##
## Rule 132: (3/2, lift 28.4)
##  location.state = FL
##  vehicle.make = Mitsubishi
##  vehicle.type = suv
##  ->  class 0.5  [0.400]
##
## Rule 133: (3/2, lift 28.4)
```

```
##  location.state = NM
##  vehicle.make = Ford
##  ->  class 0.5  [0.400]
##
## Rule 134: (3/2, lift 28.4)
##  location.state = NV
##  vehicle.make = Lexus
##  ->  class 0.5  [0.400]
##
## Rule 135: (7/5, lift 23.6)
##  location.state = VA
##  vehicle.make = Nissan
##  vehicle.type = car
##  ->  class 0.5  [0.333]
##
## Rule 136: (82/78, lift 4.2)
##  location.state = TN
##  ->  class 0.5  [0.060]
##
## Rule 137: (66/63, lift 4.2)
##  vehicle.make = Land Rover
##  ->  class 0.5  [0.059]
##
## Rule 138: (173/168, lift 2.4)
##  vehicle.make = Nissan
##  ->  class 0.5  [0.034]
##
## Rule 139: (1, lift 260.1)
##  location.state = IL
##  vehicle.make = Land Rover
##  ->  class 0.571428571428571  [0.667]
##
## Rule 140: (1, lift 260.1)
##  location.state = IL
##  vehicle.make = Nissan
##  vehicle.type = suv
##  ->  class 0.571428571428571  [0.667]
##
## Rule 141: (1, lift 260.1)
##  location.state = WI
##  vehicle.make = FIAT
##  ->  class 0.571428571428571  [0.667]
##
## Rule 142: (4/3, lift 130.0)
##  location.state = CA
##  vehicle.make = Subaru
##  ->  class 0.571428571428571  [0.333]
##
## Rule 143: (11/10, lift 60.0)
##  location.state = NJ
##  vehicle.make = Toyota
##  ->  class 0.571428571428571  [0.154]
##
## Rule 144: (23/22, lift 31.2)
```

```
##  location.state = MN
##  vehicle.make = Toyota
##  ->  class 0.571428571428571  [0.080]
##
## Rule 145: (1, lift 2600.7)
##  location.state = UT
##  vehicle.make = MINI
##  ->  class 0.583333333333333  [0.667]
##
## Rule 146: (1, lift 123.8)
##  location.state = IN
##  vehicle.make = Volkswagen
##  ->  class 0.6  [0.667]
##
## Rule 147: (1, lift 123.8)
##  location.state = SC
##  vehicle.make = Lexus
##  ->  class 0.6  [0.667]
##
## Rule 148: (1, lift 123.8)
##  location.state = WI
##  vehicle.make = Dodge
##  ->  class 0.6  [0.667]
##
## Rule 149: (2/1, lift 92.9)
##  location.state = HI
##  vehicle.make = Ford
##  vehicle.type = van
##  ->  class 0.6  [0.500]
##
## Rule 150: (3/2, lift 74.3)
##  location.state = LA
##  vehicle.make = Ford
##  ->  class 0.6  [0.400]
##
## Rule 151: (4/3, lift 61.9)
##  location.state = UT
##  vehicle.make = Audi
##  vehicle.type = suv
##  ->  class 0.6  [0.333]
##
## Rule 152: (5/4, lift 53.1)
##  location.state = FL
##  vehicle.make = Porsche
##  vehicle.type = suv
##  ->  class 0.6  [0.286]
##
## Rule 153: (6/5, lift 46.4)
##  location.state = GA
##  vehicle.make = Jeep
##  ->  class 0.6  [0.250]
##
## Rule 154: (292/287, lift 3.8)
##  vehicle.make = Chevrolet
```

```
##  ->  class 0.6  [0.020]
##
## Rule 155: (1, lift 866.9)
##  fuelType = HYBRID
##  location.state = CO
##  vehicle.make = Toyota
##  ->  class 0.625  [0.667]
##
## Rule 156: (1, lift 866.9)
##  location.state = VA
##  vehicle.make = Mercedes-Benz
##  vehicle.type = suv
##  ->  class 0.625  [0.667]
##
## Rule 157: (1, lift 92.9)
##  location.state = IL
##  vehicle.make = Buick
##  ->  class 0.666666666666667  [0.667]
##
## Rule 158: (1, lift 92.9)
##  location.state = SC
##  vehicle.make = Buick
##  ->  class 0.666666666666667  [0.667]
##
## Rule 159: (2/1, lift 69.7)
##  location.state = NJ
##  vehicle.make = Toyota
##  vehicle.type = minivan
##  ->  class 0.666666666666667  [0.500]
##
## Rule 160: (2/1, lift 69.7)
##  location.state = WI
##  vehicle.make = Hyundai
##  ->  class 0.666666666666667  [0.500]
##
## Rule 161: (3/2, lift 55.7)
##  location.state = FL
##  vehicle.make = GMC
##  ->  class 0.666666666666667  [0.400]
##
## Rule 162: (5/4, lift 39.8)
##  location.state = KY
##  vehicle.make = Ford
##  vehicle.type = car
##  ->  class 0.666666666666667  [0.286]
##
## Rule 163: (5/4, lift 39.8)
##  location.state = NC
##  vehicle.make = Kia
##  ->  class 0.666666666666667  [0.286]
##
## Rule 164: (5/4, lift 39.8)
##  location.state = UT
##  vehicle.make = Subaru
```

```
##  ->  class 0.666666666666667  [0.286]
##
## Rule 165: (1, lift 866.9)
##  location.state = MA
##  vehicle.make = Lotus
##  ->  class 0.7  [0.667]
##
## Rule 166: (3/1, lift 260.1)
##  location.state = AZ
##  vehicle.make = Porsche
##  ->  class 0.714285714285714  [0.600]
##
## Rule 167: (2/1, lift 216.7)
##  location.state = TN
##  vehicle.make = Audi
##  ->  class 0.714285714285714  [0.500]
##
## Rule 168: (3/2, lift 173.4)
##  location.state = CA
##  vehicle.make = smart
##  ->  class 0.714285714285714  [0.400]
##
## Rule 169: (3/2, lift 173.4)
##  location.state = NJ
##  vehicle.make = Porsche
##  vehicle.type = suv
##  ->  class 0.714285714285714  [0.400]
##
## Rule 170: (5/4, lift 123.8)
##  location.state = CA
##  vehicle.make = Nissan
##  vehicle.type = car
##  ->  class 0.714285714285714  [0.286]
##
## Rule 171: (2, lift 139.3)
##  location.state = OK
##  vehicle.make in {Land Rover, Nissan}
##  ->  class 0.75  [0.750]
##
## Rule 172: (1, lift 123.8)
##  fuelType = DIESEL
##  location.state = FL
##  vehicle.make = Mercedes-Benz
##  ->  class 0.75  [0.667]
##
## Rule 173: (5/3, lift 79.6)
##  location.state = HI
##  vehicle.make in {Audi, Porsche}
##  ->  class 0.75  [0.429]
##
## Rule 174: (3/2, lift 74.3)
##  location.state = MD
##  vehicle.make = BMW
##  ->  class 0.75  [0.400]
```

```
##
## Rule 175: (3/2, lift 74.3)
##  location.state = UT
##  vehicle.make = Volkswagen
##  ->  class 0.75  [0.400]
##
## Rule 176: (4/3, lift 61.9)
##  location.state = OR
##  vehicle.make = Ford
##  ->  class 0.75  [0.333]
##
## Rule 177: (5/4, lift 53.1)
##  location.state = WI
##  vehicle.make = Toyota
##  ->  class 0.75  [0.286]
##
## Rule 178: (7/6, lift 41.3)
##  location.state = CA
##  vehicle.make = Mazda
##  ->  class 0.75  [0.222]
##
## Rule 179: (1, lift 866.9)
##  location.state = AZ
##  vehicle.make = Lincoln
##  ->  class 0.777777777777778  [0.667]
##
## Rule 180: (2/1, lift 650.2)
##  location.state = VA
##  vehicle.make = Mazda
##  vehicle.type = car
##  ->  class 0.777777777777778  [0.500]
##
## Rule 181: (1, lift 162.5)
##  location.state = NE
##  vehicle.make = BMW
##  ->  class 0.8  [0.667]
##
## Rule 182: (3/2, lift 97.5)
##  location.state = IL
##  vehicle.make = Jeep
##  ->  class 0.8  [0.400]
##
## Rule 183: (3/2, lift 97.5)
##  location.state = MD
##  vehicle.make = Land Rover
##  ->  class 0.8  [0.400]
##
## Rule 184: (3/2, lift 97.5)
##  location.state = VA
##  vehicle.make = Dodge
##  ->  class 0.8  [0.400]
##
## Rule 185: (4/3, lift 81.3)
##  location.state = NC
```

```
##  vehicle.make = Nissan
##  vehicle.type = car
##  ->  class 0.8  [0.333]
##
## Rule 186: (4/3, lift 81.3)
##  location.state = OR
##  vehicle.make = Honda
##  ->  class 0.8  [0.333]
##
## Rule 187: (4/3, lift 81.3)
##  location.state = WA
##  vehicle.make = Volkswagen
##  ->  class 0.8  [0.333]
##
## Rule 188: (5/4, lift 69.7)
##  location.state = CA
##  vehicle.make = Jaguar
##  ->  class 0.8  [0.286]
##
## Rule 189: (7/6, lift 54.2)
##  location.state = TN
##  vehicle.make = Jeep
##  ->  class 0.8  [0.222]
##
## Rule 190: (1, lift 866.9)
##  location.state = WA
##  vehicle.make = Chevrolet
##  vehicle.type = truck
##  ->  class 0.818181818181818  [0.667]
##
## Rule 191: (3/2, lift 520.1)
##  location.state = PA
##  vehicle.make = Hyundai
##  ->  class 0.818181818181818  [0.400]
##
## Rule 192: (6/5, lift 325.1)
##  location.state = NJ
##  vehicle.make = Honda
##  ->  class 0.818181818181818  [0.250]
##
## Rule 193: (1, lift 162.5)
##  location.state = MI
##  vehicle.make = Mercedes-Benz
##  ->  class 0.833333333333333  [0.667]
##
## Rule 194: (2/1, lift 121.9)
##  location.state = LA
##  vehicle.make = Nissan
##  ->  class 0.833333333333333  [0.500]
##
## Rule 195: (2/1, lift 121.9)
##  location.state = TN
##  vehicle.make = Chevrolet
##  vehicle.type = van
```

```
##  ->  class 0.833333333333333  [0.500]
##
## Rule 196: (4/3, lift 81.3)
##  location.state = MI
##  vehicle.make = Dodge
##  ->  class 0.833333333333333  [0.333]
##
## Rule 197: (4/3, lift 81.3)
##  location.state = MO
##  vehicle.make = Toyota
##  vehicle.type = car
##  ->  class 0.833333333333333  [0.333]
##
## Rule 198: (5/4, lift 69.7)
##  location.state = VA
##  vehicle.make = Kia
##  ->  class 0.833333333333333  [0.286]
##
## Rule 199: (9/7, lift 66.5)
##  location.state = CA
##  vehicle.make = Kia
##  vehicle.type = car
##  ->  class 0.833333333333333  [0.273]
##
## Rule 200: (11/10, lift 37.5)
##  location.state = NC
##  vehicle.make = Honda
##  ->  class 0.833333333333333  [0.154]
##
## Rule 201: (1, lift 2600.7)
##  location.state = NM
##  vehicle.make = GMC
##  ->  class 0.866666666666667  [0.667]
##
## Rule 202: (4/3, lift 433.4)
##  location.state = NC
##  vehicle.make = Kia
##  vehicle.type = car
##  ->  class 0.875  [0.333]
##
## Rule 203: (1, lift 1300.3)
##  fuelType = DIESEL
##  location.state = SC
##  vehicle.make = Audi
##  ->  class 0.9  [0.667]
##
## Rule 204: (2/1, lift 975.2)
##  location.state = MD
##  vehicle.make = Honda
##  ->  class 0.916666666666667  [0.500]
##
## Rule 205: (3/2, lift 1560.4)
##  location.state = HI
##  vehicle.make = Toyota
```

```
##  vehicle.type = minivan
##  ->  class 0.928571428571429  [0.400]
##
## Rule 206: (3, lift 24.4)
##  location.state = VA
##  vehicle.make in {Alfa Romeo, MINI}
##  ->  class 1  [0.800]
##
## Rule 207: (2, lift 22.9)
##  location.state = NJ
##  vehicle.make in {Lincoln, McLaren}
##  ->  class 1  [0.750]
##
## Rule 208: (1, lift 20.3)
##  location.state = FL
##  vehicle.make = Hyundai
##  vehicle.type = suv
##  ->  class 1  [0.667]
##
## Rule 209: (4/1, lift 20.3)
##  location.state = KY
##  vehicle.make in {Mercedes-Benz, Volkswagen, Volvo}
##  ->  class 1  [0.667]
##
## Rule 210: (1, lift 20.3)
##  fuelType = DIESEL
##  location.state = CA
##  vehicle.make = Porsche
##  ->  class 1  [0.667]
##
## Rule 211: (1, lift 20.3)
##  location.state = MO
##  vehicle.make = Subaru
##  ->  class 1  [0.667]
##
## Rule 212: (1, lift 20.3)
##  location.state = RI
##  vehicle.type = minivan
##  ->  class 1  [0.667]
##
## Rule 213: (1, lift 20.3)
##  location.state = TN
##  vehicle.make = Chevrolet
##  vehicle.type = truck
##  ->  class 1  [0.667]
##
## Rule 214: (1, lift 20.3)
##  vehicle.make = Lamborghini
##  vehicle.type = suv
##  ->  class 1  [0.667]
##
## Rule 215: (4/2, lift 15.2)
##  location.state = NC
##  vehicle.make in {Acura, FIAT}
```

```
##  ->  class 1  [0.500]
##
## Rule 216: (2/1, lift 15.2)
##  location.state = SC
##  vehicle.make = MINI
##  ->  class 1  [0.500]
##
## Rule 217: (2/1, lift 15.2)
##  fuelType = HYBRID
##  location.state = CT
##  ->  class 1  [0.500]
##
## Rule 218: (6/4, lift 11.4)
##  location.state = FL
##  vehicle.make = Chrysler
##  ->  class 1  [0.375]
##
## Rule 219: (6/4, lift 11.4)
##  location.state = CA
##  vehicle.make in {Alfa Romeo, GMC}
##  ->  class 1  [0.375]
##
## Rule 220: (8/6, lift 9.1)
##  location.state = FL
##  vehicle.make = Dodge
##  vehicle.type = car
##  ->  class 1  [0.300]
##
## Rule 221: (8/6, lift 9.1)
##  location.state = FL
##  vehicle.make = Mitsubishi
##  ->  class 1  [0.300]
##
## Rule 222: (8/6, lift 9.1)
##  location.state = FL
##  vehicle.make = Kia
##  vehicle.type = minivan
##  ->  class 1  [0.300]
##
## Rule 223: (12/9, lift 8.7)
##  location.state = WA
##  vehicle.make in {Dodge, Nissan, Porsche}
##  ->  class 1  [0.286]
##
## Rule 224: (16/12, lift 8.5)
##  location.state = AL
##  ->  class 1  [0.278]
##
## Rule 225: (9/7, lift 8.3)
##  location.state = IL
##  vehicle.make = Tesla
##  ->  class 1  [0.273]
##
## Rule 226: (11/9, lift 7.0)
```

```
##  location.state = FL
##  vehicle.make = Audi
##  ->  class 1  [0.231]
##
## Rule 227: (36/33, lift 3.2)
##  location.state = MA
##  ->  class 1  [0.105]
##
## Rule 228: (138/129, lift 2.2)
##  location.state = NC
##  ->  class 1  [0.071]
##
## Rule 229: (2/1, lift 975.2)
##  location.state = HI
##  vehicle.make = Mercedes-Benz
##  ->  class 1.11111111111111  [0.500]
##
## Rule 230: (1, lift 866.9)
##  vehicle.make = Ford
##  vehicle.type = minivan
##  ->  class 1.125  [0.667]
##
## Rule 231: (2/1, lift 390.1)
##  location.state = HI
##  vehicle.make = Ford
##  vehicle.type = suv
##  ->  class 1.14285714285714  [0.500]
##
## Rule 232: (7/6, lift 216.7)
##  location.state = OR
##  vehicle.make = Mercedes-Benz
##  vehicle.type = suv
##  ->  class 1.16666666666667  [0.222]
##
## Rule 233: (1, lift 325.1)
##  location.state = NJ
##  vehicle.make = GMC
##  ->  class 1.2  [0.667]
##
## Rule 234: (2/1, lift 243.8)
##  location.state = PA
##  vehicle.make = Mazda
##  ->  class 1.2  [0.500]
##
## Rule 235: (2/1, lift 278.6)
##  location.state = MO
##  vehicle.make = Kia
##  ->  class 1.22222222222222  [0.500]
##
## Rule 236: (1, lift 2600.7)
##  location.state = KS
##  vehicle.make = Lotus
##  ->  class 1.23076923076923  [0.667]
##
```

```
## Rule 237: (1, lift 130.0)
##  location.state = MO
##  vehicle.make = GMC
##  ->  class 1.25  [0.667]
##
## Rule 238: (2/1, lift 97.5)
##  location.state = TN
##  vehicle.make = Honda
##  ->  class 1.25  [0.500]
##
## Rule 239: (2/1, lift 97.5)
##  location.state = CO
##  vehicle.make = Cadillac
##  ->  class 1.25  [0.500]
##
## Rule 240: (5/4, lift 55.7)
##  location.state = CA
##  vehicle.make = Ferrari
##  ->  class 1.25  [0.286]
##
## Rule 241: (5/4, lift 55.7)
##  location.state = WA
##  vehicle.make = Honda
##  ->  class 1.25  [0.286]
##
## Rule 242: (6/5, lift 48.8)
##  location.state = IL
##  vehicle.make = Honda
##  ->  class 1.25  [0.250]
##
## Rule 243: (1, lift 2600.7)
##  location.state = MI
##  vehicle.make = Chevrolet
##  vehicle.type = car
##  ->  class 1.27272727272727  [0.667]
##
## Rule 244: (3/2, lift 222.9)
##  fuelType = GASOLINE
##  location.state = MN
##  vehicle.make = Chevrolet
##  ->  class 1.28571428571429  [0.400]
##
## Rule 245: (3/2, lift 222.9)
##  location.state = OK
##  vehicle.make = Volkswagen
##  ->  class 1.28571428571429  [0.400]
##
## Rule 246: (4/3, lift 185.8)
##  location.state = RI
##  vehicle.type = car
##  ->  class 1.28571428571429  [0.333]
##
## Rule 247: (13/12, lift 74.3)
##  location.state = AZ
```

```
##  vehicle.make = Tesla
##  vehicle.type = car
##  ->  class 1.28571428571429  [0.133]
##
## Rule 248: (5/4, lift 1114.6)
##  location.state = TN
##  vehicle.make = Mercedes-Benz
##  ->  class 1.3  [0.286]
##
## Rule 249: (1, lift 89.7)
##  location.state = GA
##  vehicle.make = Polaris
##  ->  class 1.33333333333333  [0.667]
##
## Rule 250: (1, lift 89.7)
##  location.state = MO
##  vehicle.make = Maserati
##  ->  class 1.33333333333333  [0.667]
##
## Rule 251: (1, lift 89.7)
##  location.state = OK
##  vehicle.make = Porsche
##  ->  class 1.33333333333333  [0.667]
##
## Rule 252: (1, lift 89.7)
##  location.state = OR
##  vehicle.make = Porsche
##  vehicle.type = suv
##  ->  class 1.33333333333333  [0.667]
##
## Rule 253: (1, lift 89.7)
##  location.state = SC
##  vehicle.make = BMW
##  ->  class 1.33333333333333  [0.667]
##
## Rule 254: (2/1, lift 67.3)
##  location.state = OR
##  vehicle.make = Subaru
##  vehicle.type = car
##  ->  class 1.33333333333333  [0.500]
##
## Rule 255: (4/2, lift 67.3)
##  location.state = PA
##  vehicle.make = Nissan
##  ->  class 1.33333333333333  [0.500]
##
## Rule 256: (5/3, lift 57.7)
##  location.state = NC
##  vehicle.make = Chevrolet
##  vehicle.type = car
##  ->  class 1.33333333333333  [0.429]
##
## Rule 257: (5/3, lift 57.7)
##  location.state = AZ
```

```
##  vehicle.make = Honda
##  ->  class 1.33333333333333  [0.429]
##
## Rule 258: (3/2, lift 53.8)
##  fuelType = HYBRID
##  location.state = WI
##  vehicle.make = Toyota
##  ->  class 1.33333333333333  [0.400]
##
## Rule 259: (4/3, lift 44.8)
##  location.state = NC
##  vehicle.make = Toyota
##  vehicle.type = suv
##  ->  class 1.33333333333333  [0.333]
##
## Rule 260: (6/5, lift 33.6)
##  location.state = NV
##  vehicle.make = Ford
##  ->  class 1.33333333333333  [0.250]
##
## Rule 261: (2/1, lift 650.2)
##  location.state = HI
##  vehicle.make = Volkswagen
##  ->  class 1.375  [0.500]
##
## Rule 262: (3/2, lift 520.1)
##  location.state = NC
##  vehicle.make = Honda
##  vehicle.type = minivan
##  ->  class 1.375  [0.400]
##
## Rule 263: (1, lift 130.0)
##  location.state = IL
##  vehicle.type = minivan
##  ->  class 1.4  [0.667]
##
## Rule 264: (1, lift 130.0)
##  location.state = AZ
##  vehicle.make = Ram
##  ->  class 1.4  [0.667]
##
## Rule 265: (1, lift 130.0)
##  location.state = MN
##  vehicle.make = Lexus
##  ->  class 1.4  [0.667]
##
## Rule 266: (1, lift 130.0)
##  location.state = NC
##  vehicle.make = Aston Martin
##  ->  class 1.4  [0.667]
##
## Rule 267: (1, lift 130.0)
##  location.state = NJ
##  vehicle.make = Cadillac
```

```
##  ->  class 1.4  [0.667]
##
## Rule 268: (2/1, lift 97.5)
##  location.state = IL
##  vehicle.make = BMW
##  vehicle.type = car
##  ->  class 1.4  [0.500]
##
## Rule 269: (5/4, lift 55.7)
##  location.state = HI
##  vehicle.make = Dodge
##  ->  class 1.4  [0.286]
##
## Rule 270: (1, lift 650.2)
##  fuelType = HYBRID
##  location.state = NJ
##  vehicle.make = Toyota
##  ->  class 1.41666666666667  [0.667]
##
## Rule 271: (2/1, lift 487.6)
##  location.state = IN
##  vehicle.make = Chevrolet
##  vehicle.type = truck
##  ->  class 1.41666666666667  [0.500]
##
## Rule 272: (5/4, lift 278.6)
##  location.state = NV
##  vehicle.make = Chevrolet
##  ->  class 1.41666666666667  [0.286]
##
## Rule 273: (1, lift 520.1)
##  location.state = MI
##  vehicle.make = Land Rover
##  ->  class 1.42857142857143  [0.667]
##
## Rule 274: (2/1, lift 390.1)
##  location.state = SC
##  vehicle.make = Volkswagen
##  ->  class 1.42857142857143  [0.500]
##
## Rule 275: (6/5, lift 195.0)
##  location.state = MD
##  vehicle.make = Tesla
##  ->  class 1.42857142857143  [0.250]
##
## Rule 276: (2/1, lift 650.2)
##  location.state = FL
##  vehicle.make = Cadillac
##  ->  class 1.44444444444444  [0.500]
##
## Rule 277: (1, lift 1300.3)
##  location.state = AZ
##  vehicle.make = Cadillac
##  ->  class 1.45454545454545  [0.667]
```

```
##
## Rule 278: (8/7, lift 390.1)
##  location.state = PA
##  vehicle.make = Honda
##  ->  class 1.45454545454545  [0.200]
##
## Rule 279: (1, lift 52.0)
##  location.state = IL
##  vehicle.make = Dodge
##  ->  class 1.5  [0.667]
##
## Rule 280: (1, lift 52.0)
##  location.state = IN
##  vehicle.make = Audi
##  ->  class 1.5  [0.667]
##
## Rule 281: (1, lift 52.0)
##  location.state = OR
##  vehicle.make = Land Rover
##  ->  class 1.5  [0.667]
##
## Rule 282: (2/1, lift 39.0)
##  location.state = MA
##  vehicle.make = Honda
##  ->  class 1.5  [0.500]
##
## Rule 283: (2/1, lift 39.0)
##  location.state = WV
##  vehicle.make = Audi
##  ->  class 1.5  [0.500]
##
## Rule 284: (6/4, lift 29.3)
##  location.state = IL
##  vehicle.make = Tesla
##  vehicle.type = car
##  ->  class 1.5  [0.375]
##
## Rule 285: (6/4, lift 29.3)
##  location.state = GA
##  vehicle.make in {Hyundai, Jaguar}
##  ->  class 1.5  [0.375]
##
## Rule 286: (7/5, lift 26.0)
##  location.state = HI
##  vehicle.make = Nissan
##  ->  class 1.5  [0.333]
##
## Rule 287: (8/6, lift 23.4)
##  fuelType = HYBRID
##  location.state = CA
##  vehicle.make = BMW
##  ->  class 1.5  [0.300]
##
## Rule 288: (12/10, lift 16.7)
```

```
##  location.state = GA
##  vehicle.make = Toyota
##  ->  class 1.5  [0.214]
##
## Rule 289: (138/134, lift 2.8)
##  location.state = HI
##  ->  class 1.5  [0.036]
##
## Rule 290: (4/3, lift 260.1)
##  location.state = IL
##  vehicle.make = Honda
##  vehicle.type = car
##  ->  class 1.57142857142857  [0.333]
##
## Rule 291: (4/3, lift 260.1)
##  location.state = WA
##  vehicle.make = Hyundai
##  ->  class 1.57142857142857  [0.333]
##
## Rule 292: (1, lift 289.0)
##  location.state = NC
##  vehicle.make = Chevrolet
##  vehicle.type = suv
##  ->  class 1.6  [0.667]
##
## Rule 293: (1, lift 289.0)
##  location.state = HI
##  vehicle.make = Chevrolet
##  vehicle.type = suv
##  ->  class 1.6  [0.667]
##
## Rule 294: (3/2, lift 173.4)
##  location.state = MD
##  vehicle.make = Volkswagen
##  ->  class 1.6  [0.400]
##
## Rule 295: (2/1, lift 390.1)
##  location.state = FL
##  vehicle.make = FIAT
##  ->  class 1.625  [0.500]
##
## Rule 296: (20/18, lift 106.4)
##  location.state = MN
##  vehicle.make = Ford
##  ->  class 1.625  [0.136]
##
## Rule 297: (1, lift 104.0)
##  location.state = GA
##  vehicle.type = truck
##  ->  class 1.66666666666667  [0.667]
##
## Rule 298: (1, lift 104.0)
##  location.state = UT
##  vehicle.make = Lexus
```

```
##  ->  class 1.66666666666667  [0.667]
##
## Rule 299: (2/1, lift 78.0)
##  fuelType = ELECTRIC
##  location.state = CA
##  vehicle.make = BMW
##  ->  class 1.66666666666667  [0.500]
##
## Rule 300: (2/1, lift 78.0)
##  location.state = WA
##  vehicle.make = Maserati
##  ->  class 1.66666666666667  [0.500]
##
## Rule 301: (4/3, lift 52.0)
##  location.state = FL
##  vehicle.make = Volkswagen
##  vehicle.type = suv
##  ->  class 1.66666666666667  [0.333]
##
## Rule 302: (1, lift 371.5)
##  location.state = NJ
##  vehicle.make = Scion
##  ->  class 1.71428571428571  [0.667]
##
## Rule 303: (3/2, lift 222.9)
##  location.state = PA
##  vehicle.make = Chevrolet
##  ->  class 1.71428571428571  [0.400]
##
## Rule 304: (3/2, lift 222.9)
##  location.state = UT
##  vehicle.make = Ram
##  ->  class 1.71428571428571  [0.400]
##
## Rule 305: (1, lift 216.7)
##  location.state = OR
##  vehicle.make = Maserati
##  ->  class 1.75  [0.667]
##
## Rule 306: (2/1, lift 162.5)
##  location.state = HI
##  vehicle.make = Toyota
##  vehicle.type = truck
##  ->  class 1.75  [0.500]
##
## Rule 307: (2/1, lift 162.5)
##  location.state = CA
##  vehicle.make = Acura
##  vehicle.type = car
##  ->  class 1.75  [0.500]
##
## Rule 308: (2/1, lift 162.5)
##  location.state = WA
##  vehicle.make = Kia
```

```
##  ->  class 1.75  [0.500]
##
## Rule 309: (6/5, lift 81.3)
##  location.state = NJ
##  vehicle.make = Mercedes-Benz
##  vehicle.type = car
##  ->  class 1.75  [0.250]
##
## Rule 310: (1, lift 1300.3)
##  location.state = NM
##  vehicle.make = Porsche
##  ->  class 1.76923076923077  [0.667]
##
## Rule 311: (3/2, lift 780.2)
##  location.state = AZ
##  vehicle.make = Chevrolet
##  vehicle.type = suv
##  ->  class 1.76923076923077  [0.400]
##
## Rule 312: (1, lift 144.5)
##  location.state = VA
##  vehicle.make = Mercury
##  ->  class 1.8  [0.667]
##
## Rule 313: (2/1, lift 108.4)
##  location.state = MD
##  vehicle.make = Kia
##  ->  class 1.8  [0.500]
##
## Rule 314: (2/1, lift 108.4)
##  location.state = OR
##  vehicle.make = Kia
##  ->  class 1.8  [0.500]
##
## Rule 315: (5/4, lift 61.9)
##  location.state = MN
##  vehicle.make = Tesla
##  ->  class 1.8  [0.286]
##
## Rule 316: (5/4, lift 222.9)
##  location.state = VA
##  vehicle.make = Honda
##  ->  class 1.83333333333333  [0.286]
##
## Rule 317: (1, lift 520.1)
##  location.state = ME
##  vehicle.make = Honda
##  ->  class 1.875  [0.667]
##
## Rule 318: (4/3, lift 433.4)
##  fuelType = HYBRID
##  location.state = FL
##  vehicle.make = Toyota
##  ->  class 1.90909090909091  [0.333]
```

```
##
## Rule 319: (2/1, lift 1950.5)
##  location.state = WA
##  vehicle.make = Honda
##  vehicle.type = minivan
##  ->  class 1.92307692307692  [0.500]
##
## Rule 320: (2, lift 29.6)
##  location.state = KS
##  vehicle.make in {Lexus, Nissan}
##  ->  class 2  [0.750]
##
## Rule 321: (2, lift 29.6)
##  location.state = AZ
##  vehicle.make in {Audi, FIAT}
##  ->  class 2  [0.750]
##
## Rule 322: (1, lift 26.3)
##  location.state = LA
##  vehicle.make = Polaris
##  ->  class 2  [0.667]
##
## Rule 323: (1, lift 26.3)
##  fuelType = HYBRID
##  location.state = CA
##  vehicle.make = Toyota
##  vehicle.type = suv
##  ->  class 2  [0.667]
##
## Rule 324: (1, lift 26.3)
##  location.state = HI
##  vehicle.make = Dodge
##  vehicle.type = car
##  ->  class 2  [0.667]
##
## Rule 325: (1, lift 26.3)
##  location.state = MN
##  vehicle.make = Ram
##  ->  class 2  [0.667]
##
## Rule 326: (1, lift 26.3)
##  location.state = MN
##  vehicle.make = Tesla
##  vehicle.type = suv
##  ->  class 2  [0.667]
##
## Rule 327: (1, lift 26.3)
##  location.state = NJ
##  vehicle.make = Acura
##  ->  class 2  [0.667]
##
## Rule 328: (1, lift 26.3)
##  location.state = VA
##  vehicle.make = Honda
```

```
##  vehicle.type = minivan
##  ->  class 2  [0.667]
##
## Rule 329: (1, lift 26.3)
##  location.state = WA
##  vehicle.make = Ford
##  vehicle.type = suv
##  ->  class 2  [0.667]
##
## Rule 330: (3/1, lift 23.6)
##  location.state = KY
##  vehicle.make in {Chevrolet, Toyota}
##  ->  class 2  [0.600]
##
## Rule 331: (3/1, lift 23.6)
##  location.state = FL
##  vehicle.make in {Lincoln, Subaru}
##  ->  class 2  [0.600]
##
## Rule 332: (5/2, lift 22.5)
##  location.state = UT
##  vehicle.make in {Jaguar, Lincoln, Mazda}
##  ->  class 2  [0.571]
##
## Rule 333: (2/1, lift 19.7)
##  location.state = MI
##  vehicle.make = Ford
##  vehicle.type = suv
##  ->  class 2  [0.500]
##
## Rule 334: (2/1, lift 19.7)
##  location.state = FL
##  vehicle.make = Ford
##  vehicle.type = suv
##  ->  class 2  [0.500]
##
## Rule 335: (3/2, lift 15.8)
##  location.state = RI
##  vehicle.type = suv
##  ->  class 2  [0.400]
##
## Rule 336: (6/4, lift 14.8)
##  vehicle.make = Jaguar
##  vehicle.type = suv
##  ->  class 2  [0.375]
##
## Rule 337: (7/5, lift 13.1)
##  location.state = VA
##  vehicle.make = Toyota
##  ->  class 2  [0.333]
##
## Rule 338: (9/7, lift 10.7)
##  location.state = CO
##  vehicle.make = Honda
```

```
##  ->  class 2  [0.273]
##
## Rule 339: (13/11, lift 7.9)
##  location.state = OR
##  vehicle.make = BMW
##  ->  class 2  [0.200]
##
## Rule 340: (36/33, lift 4.1)
##  location.state = MO
##  ->  class 2  [0.105]
##
## Rule 341: (46/43, lift 3.3)
##  vehicle.make = Mazda
##  ->  class 2  [0.083]
##
## Rule 342: (5/4, lift 371.5)
##  fuelType = GASOLINE
##  location.state = CA
##  vehicle.make = Lexus
##  vehicle.type = car
##  ->  class 2.1  [0.286]
##
## Rule 343: (1, lift 650.2)
##  location.state = MO
##  vehicle.make = Chrysler
##  ->  class 2.125  [0.667]
##
## Rule 344: (2/1, lift 487.6)
##  location.state = CO
##  vehicle.make = Mercedes-Benz
##  vehicle.type = suv
##  ->  class 2.125  [0.500]
##
## Rule 345: (4/3, lift 325.1)
##  location.state = CA
##  vehicle.make = MINI
##  ->  class 2.125  [0.333]
##
## Rule 346: (1, lift 866.9)
##  location.state = MO
##  vehicle.make = Land Rover
##  ->  class 2.14285714285714  [0.667]
##
## Rule 347: (4/3, lift 433.4)
##  location.state = PA
##  vehicle.make = Mercedes-Benz
##  vehicle.type = suv
##  ->  class 2.14285714285714  [0.333]
##
## Rule 348: (1, lift 371.5)
##  location.state = IA
##  vehicle.make = Ford
##  ->  class 2.16666666666667  [0.667]
##
```

```
## Rule 349: (3/2, lift 222.9)
##  location.state = HI
##  vehicle.make = Honda
##  vehicle.type = car
##  ->  class 2.16666666666667  [0.400]
##
## Rule 350: (1, lift 260.1)
##  location.state = IN
##  vehicle.make = Jeep
##  ->  class 2.2  [0.667]
##
## Rule 351: (1, lift 260.1)
##  location.state = MO
##  vehicle.make = Lincoln
##  ->  class 2.2  [0.667]
##
## Rule 352: (7/6, lift 86.7)
##  location.state = VA
##  vehicle.make = Tesla
##  ->  class 2.2  [0.222]
##
## Rule 353: (1, lift 2600.7)
##  location.state = IL
##  vehicle.make = Mercury
##  ->  class 2.21428571428571  [0.667]
##
## Rule 354: (2/1, lift 390.1)
##  location.state = ID
##  vehicle.make = Volkswagen
##  ->  class 2.22222222222222  [0.500]
##
## Rule 355: (2/1, lift 390.1)
##  location.state = NC
##  vehicle.make = Chrysler
##  ->  class 2.22222222222222  [0.500]
##
## Rule 356: (2/1, lift 139.3)
##  location.state = MN
##  vehicle.make = Mercedes-Benz
##  ->  class 2.25  [0.500]
##
## Rule 357: (2/1, lift 139.3)
##  location.state = NV
##  vehicle.make = Alfa Romeo
##  ->  class 2.25  [0.500]
##
## Rule 358: (3/2, lift 111.5)
##  location.state = PA
##  vehicle.make = Dodge
##  vehicle.type = minivan
##  ->  class 2.25  [0.400]
##
## Rule 359: (5/4, lift 79.6)
##  location.state = CA
```

```
##  vehicle.make = Acura
##  ->  class 2.25  [0.286]
##
## Rule 360: (1, lift 520.1)
##  location.state = HI
##  vehicle.make = smart
##  ->  class 2.28571428571429  [0.667]
##
## Rule 361: (1, lift 520.1)
##  fuelType = DIESEL
##  location.state = PA
##  vehicle.make = Mercedes-Benz
##  ->  class 2.28571428571429  [0.667]
##
## Rule 362: (2/1, lift 390.1)
##  location.state = MO
##  vehicle.make = Dodge
##  ->  class 2.28571428571429  [0.500]
##
## Rule 363: (2/1, lift 1950.5)
##  fuelType = HYBRID
##  location.state = GA
##  vehicle.make = Toyota
##  ->  class 2.3  [0.500]
##
## Rule 364: (1, lift 1300.3)
##  location.state = VA
##  vehicle.make = HUMMER
##  ->  class 2.30769230769231  [0.667]
##
## Rule 365: (2/1, lift 975.2)
##  location.state = PA
##  vehicle.make = Land Rover
##  ->  class 2.30769230769231  [0.500]
##
## Rule 366: (1, lift 108.4)
##  location.state = MN
##  vehicle.make = Toyota
##  vehicle.type = truck
##  ->  class 2.33333333333333  [0.667]
##
## Rule 367: (1, lift 108.4)
##  location.state = NC
##  vehicle.make = Hyundai
##  vehicle.type = suv
##  ->  class 2.33333333333333  [0.667]
##
## Rule 368: (1, lift 108.4)
##  location.state = NV
##  vehicle.make = Bentley
##  ->  class 2.33333333333333  [0.667]
##
## Rule 369: (1, lift 108.4)
##  location.state = HI
```

```
##  vehicle.make = MINI
##  ->  class 2.33333333333333  [0.667]
##
## Rule 370: (1, lift 108.4)
##  location.state = PA
##  vehicle.make = BMW
##  vehicle.type = suv
##  ->  class 2.33333333333333  [0.667]
##
## Rule 371: (1, lift 108.4)
##  location.state = TN
##  vehicle.make = Ram
##  ->  class 2.33333333333333  [0.667]
##
## Rule 372: (2/1, lift 81.3)
##  location.state = CA
##  vehicle.make = Cadillac
##  ->  class 2.33333333333333  [0.500]
##
## Rule 373: (4/3, lift 54.2)
##  fuelType = HYBRID
##  location.state = HI
##  vehicle.make = Toyota
##  ->  class 2.33333333333333  [0.333]
##
## Rule 374: (5/4, lift 46.4)
##  location.state = FL
##  vehicle.make = Mazda
##  ->  class 2.33333333333333  [0.286]
##
## Rule 375: (6/5, lift 40.6)
##  location.state = OR
##  vehicle.make = Lexus
##  ->  class 2.33333333333333  [0.250]
##
## Rule 376: (9/8, lift 29.6)
##  location.state = CO
##  vehicle.make = GMC
##  ->  class 2.33333333333333  [0.182]
##
## Rule 377: (1, lift 650.2)
##  location.state = LA
##  vehicle.make = Mercedes-Benz
##  ->  class 2.375  [0.667]
##
## Rule 378: (3/2, lift 173.4)
##  location.state = MN
##  vehicle.make = Ford
##  vehicle.type = car
##  ->  class 2.4  [0.400]
##
## Rule 379: (4/3, lift 144.5)
##  location.state = NV
##  vehicle.make = Toyota
```

```
##  ->  class 2.4  [0.333]
##
## Rule 380: (4/3, lift 144.5)
##  location.state = VA
##  vehicle.make = BMW
##  ->  class 2.4  [0.333]
##
## Rule 381: (1, lift 866.9)
##  location.state = WA
##  vehicle.make = Mazda
##  ->  class 2.42857142857143  [0.667]
##
## Rule 382: (2, lift 91.4)
##  location.state = HI
##  vehicle.make in {Polaris, Subaru}
##  ->  class 2.5  [0.750]
##
## Rule 383: (1, lift 81.3)
##  location.state = MN
##  vehicle.make = Jeep
##  ->  class 2.5  [0.667]
##
## Rule 384: (1, lift 81.3)
##  location.state = GA
##  vehicle.make = Chrysler
##  ->  class 2.5  [0.667]
##
## Rule 385: (1, lift 81.3)
##  location.state = HI
##  vehicle.make = Chevrolet
##  vehicle.type = truck
##  ->  class 2.5  [0.667]
##
## Rule 386: (1, lift 81.3)
##  location.state = TN
##  vehicle.make = Mercedes-Benz
##  vehicle.type = suv
##  ->  class 2.5  [0.667]
##
## Rule 387: (2/1, lift 61.0)
##  location.state = UT
##  vehicle.make = Dodge
##  vehicle.type = minivan
##  ->  class 2.5  [0.500]
##
## Rule 388: (20/17, lift 22.2)
##  location.state = AZ
##  vehicle.make in {Jeep, Lexus}
##  ->  class 2.5  [0.182]
##
## Rule 389: (2/1, lift 975.2)
##  location.state = PA
##  vehicle.make = Ford
##  ->  class 2.53846153846154  [0.500]
```

```
##
## Rule 390: (3/2, lift 1560.4)
##  location.state = WA
##  vehicle.make = Land Rover
##  ->  class 2.55555555555556  [0.400]
##
## Rule 391: (2/1, lift 487.6)
##  location.state = NC
##  vehicle.make = Lexus
##  ->  class 2.57142857142857  [0.500]
##
## Rule 392: (5/4, lift 278.6)
##  location.state = UT
##  vehicle.make = Honda
##  ->  class 2.57142857142857  [0.286]
##
## Rule 393: (2/1, lift 278.6)
##  location.state = MA
##  vehicle.make = Lexus
##  ->  class 2.6  [0.500]
##
## Rule 394: (5/4, lift 159.2)
##  location.state = OR
##  vehicle.make = Volkswagen
##  ->  class 2.6  [0.286]
##
## Rule 395: (1, lift 136.9)
##  location.state = CO
##  vehicle.make = Mitsubishi
##  ->  class 2.66666666666667  [0.667]
##
## Rule 396: (1, lift 136.9)
##  location.state = TN
##  vehicle.make = Mazda
##  ->  class 2.66666666666667  [0.667]
##
## Rule 397: (2/1, lift 102.7)
##  location.state = OR
##  vehicle.make = Mazda
##  ->  class 2.66666666666667  [0.500]
##
## Rule 398: (2/1, lift 102.7)
##  location.state = WI
##  vehicle.make = Kia
##  ->  class 2.66666666666667  [0.500]
##
## Rule 399: (4/3, lift 68.4)
##  location.state = CO
##  vehicle.make = Chevrolet
##  vehicle.type = car
##  ->  class 2.66666666666667  [0.333]
##
## Rule 400: (5/4, lift 58.7)
##  location.state = FL
```

```
##  vehicle.make = Honda
##  vehicle.type = car
##  ->  class 2.66666666666667  [0.286]
##
## Rule 401: (1, lift 2600.7)
##  vehicle.make = Saturn
##  ->  class 2.69230769230769  [0.667]
##
## Rule 402: (1, lift 2600.7)
##  location.state = GA
##  vehicle.make = smart
##  ->  class 2.7  [0.667]
##
## Rule 403: (1, lift 866.9)
##  location.state = NC
##  vehicle.make = MINI
##  ->  class 2.71428571428571  [0.667]
##
## Rule 404: (3/2, lift 520.1)
##  location.state = HI
##  vehicle.make = Dodge
##  vehicle.type = minivan
##  ->  class 2.71428571428571  [0.400]
##
## Rule 405: (1, lift 185.8)
##  fuelType = HYBRID
##  location.state = NC
##  vehicle.make = BMW
##  ->  class 2.75  [0.667]
##
## Rule 406: (2/1, lift 139.3)
##  location.state = LA
##  vehicle.make = Toyota
##  ->  class 2.75  [0.500]
##
## Rule 407: (2/1, lift 139.3)
##  location.state = HI
##  vehicle.make = Mazda
##  ->  class 2.75  [0.500]
##
## Rule 408: (1, lift 433.4)
##  location.state = MA
##  vehicle.make = MINI
##  ->  class 2.8  [0.667]
##
## Rule 409: (1, lift 433.4)
##  location.state = UT
##  vehicle.make = Polaris
##  ->  class 2.8  [0.667]
##
## Rule 410: (1, lift 520.1)
##  location.state = NV
##  vehicle.make = Cadillac
##  vehicle.type = car
```

```
##  ->  class 2.83333333333333  [0.667]
##
## Rule 411: (1, lift 433.4)
##  location.state = AZ
##  vehicle.make = Mitsubishi
##  ->  class 2.88888888888889  [0.667]
##
## Rule 412: (2/1, lift 325.1)
##  location.state = MN
##  vehicle.make = GMC
##  ->  class 2.88888888888889  [0.500]
##
## Rule 413: (11/10, lift 100.0)
##  location.state = AZ
##  vehicle.make = Toyota
##  ->  class 2.88888888888889  [0.154]
##
## Rule 414: (1, lift 31.0)
##  location.state = MA
##  vehicle.make = Chevrolet
##  ->  class 3  [0.667]
##
## Rule 415: (1, lift 31.0)
##  location.state = NJ
##  vehicle.make = Honda
##  vehicle.type = car
##  ->  class 3  [0.667]
##
## Rule 416: (1, lift 31.0)
##  location.state = SC
##  vehicle.make = Nissan
##  vehicle.type = suv
##  ->  class 3  [0.667]
##
## Rule 417: (1, lift 31.0)
##  location.state = WA
##  vehicle.make = FIAT
##  ->  class 3  [0.667]
##
## Rule 418: (1, lift 31.0)
##  location.state = IA
##  vehicle.make = Lincoln
##  ->  class 3  [0.667]
##
## Rule 419: (7/3, lift 25.8)
##  location.state = CO
##  vehicle.make in {Bentley, Dodge, Kia}
##  ->  class 3  [0.556]
##
## Rule 420: (2/1, lift 23.2)
##  location.state = MN
##  vehicle.make = Ford
##  vehicle.type = truck
##  ->  class 3  [0.500]
```

```
##
## Rule 421: (4/2, lift 23.2)
##  location.state = AR
##  ->  class 3  [0.500]
##
## Rule 422: (2/1, lift 23.2)
##  location.state = CA
##  vehicle.make = Volvo
##  ->  class 3  [0.500]
##
## Rule 423: (4/2, lift 23.2)
##  location.state = FL
##  vehicle.make in {Ferrari, Volvo}
##  ->  class 3  [0.500]
##
## Rule 424: (5/3, lift 19.9)
##  location.state = IL
##  vehicle.make in {Mazda, Volkswagen}
##  ->  class 3  [0.429]
##
## Rule 425: (3/2, lift 18.6)
##  location.state = SC
##  vehicle.make = Toyota
##  ->  class 3  [0.400]
##
## Rule 426: (11/8, lift 14.3)
##  location.state = NC
##  vehicle.make in {Dodge, Polaris}
##  ->  class 3  [0.308]
##
## Rule 427: (82/77, lift 3.3)
##  location.state = TN
##  ->  class 3  [0.071]
##
## Rule 428: (173/165, lift 2.4)
##  vehicle.make = Nissan
##  ->  class 3  [0.051]
##
## Rule 429: (1, lift 866.9)
##  location.state = ME
##  vehicle.make = Ford
##  ->  class 3.14285714285714  [0.667]
##
## Rule 430: (1, lift 866.9)
##  location.state = WA
##  vehicle.make = Audi
##  ->  class 3.14285714285714  [0.667]
##
## Rule 431: (1, lift 866.9)
##  location.state = NJ
##  vehicle.make = Lexus
##  ->  class 3.15384615384615  [0.667]
##
## Rule 432: (1, lift 1300.3)
```

```
##  fuelType = DIESEL
##  location.state = CA
##  vehicle.make = Volkswagen
##  ->  class 3.2  [0.667]
##
## Rule 433: (2/1, lift 243.8)
##  fuelType = HYBRID
##  location.state = OR
##  vehicle.make = Ford
##  ->  class 3.25  [0.500]
##
## Rule 434: (3/2, lift 195.1)
##  fuelType = GASOLINE
##  location.state = VA
##  vehicle.make = BMW
##  ->  class 3.25  [0.400]
##
## Rule 435: (1, lift 173.4)
##  location.state = IL
##  vehicle.make = Lexus
##  vehicle.type = suv
##  ->  class 3.33333333333333  [0.667]
##
## Rule 436: (3/2, lift 104.0)
##  location.state = MI
##  vehicle.make = Nissan
##  ->  class 3.33333333333333  [0.400]
##
## Rule 437: (3/2, lift 104.0)
##  fuelType = HYBRID
##  location.state = CA
##  vehicle.make = Mercedes-Benz
##  ->  class 3.33333333333333  [0.400]
##
## Rule 438: (6/5, lift 65.0)
##  location.state = CO
##  vehicle.make = Subaru
##  ->  class 3.33333333333333  [0.250]
##
## Rule 439: (1, lift 1300.3)
##  location.state = HI
##  vehicle.make = Lincoln
##  ->  class 3.36363636363636  [0.667]
##
## Rule 440: (1, lift 1300.3)
##  location.state = IA
##  vehicle.make = Lexus
##  ->  class 3.36363636363636  [0.667]
##
## Rule 441: (2/1, lift 195.0)
##  fuelType = DIESEL
##  location.state = CA
##  vehicle.make = BMW
##  ->  class 3.4  [0.500]
```

```
##
## Rule 442: (5/4, lift 111.5)
##  location.state = WI
##  vehicle.make = Honda
##  vehicle.type = car
##  ->  class 3.4  [0.286]
##
## Rule 443: (1, lift 866.9)
##  location.state = MI
##  vehicle.make = Hyundai
##  ->  class 3.42857142857143  [0.667]
##
## Rule 444: (1, lift 104.0)
##  location.state = CO
##  vehicle.make = FIAT
##  ->  class 3.5  [0.667]
##
## Rule 445: (2/1, lift 78.0)
##  location.state = GA
##  vehicle.make = Kia
##  ->  class 3.5  [0.500]
##
## Rule 446: (2/1, lift 78.0)
##  location.state = NM
##  vehicle.make = Toyota
##  ->  class 3.5  [0.500]
##
## Rule 447: (3/2, lift 62.4)
##  location.state = TN
##  vehicle.make = Ford
##  vehicle.type = car
##  ->  class 3.5  [0.400]
##
## Rule 448: (6/5, lift 39.0)
##  fuelType = GASOLINE
##  location.state = CA
##  vehicle.make = Tesla
##  ->  class 3.5  [0.250]
##
## Rule 449: (1, lift 2600.7)
##  location.state = GA
##  vehicle.make = Bentley
##  ->  class 3.53333333333333  [0.667]
##
## Rule 450: (3/2, lift 260.1)
##  location.state = CO
##  vehicle.make = BMW
##  vehicle.type = car
##  ->  class 3.57142857142857  [0.400]
##
## Rule 451: (26/24, lift 69.7)
##  location.state = FL
##  vehicle.make = Hyundai
##  ->  class 3.57142857142857  [0.107]
```

```
##
## Rule 452: (1, lift 325.1)
##  location.state = PA
##  vehicle.make = Honda
##  vehicle.type = minivan
##  ->  class 3.6  [0.667]
##
## Rule 453: (1, lift 1300.3)
##  location.state = UT
##  vehicle.make = Honda
##  vehicle.type = minivan
##  ->  class 3.63636363636364  [0.667]
##
## Rule 454: (1, lift 185.8)
##  location.state = NV
##  vehicle.make = Lotus
##  ->  class 3.66666666666667  [0.667]
##
## Rule 455: (1, lift 185.8)
##  location.state = SC
##  vehicle.make = Chevrolet
##  ->  class 3.66666666666667  [0.667]
##
## Rule 456: (13/11, lift 55.7)
##  fuelType = GASOLINE
##  location.state = OR
##  vehicle.make = Mercedes-Benz
##  vehicle.type = car
##  ->  class 3.66666666666667  [0.200]
##
## Rule 457: (1, lift 866.9)
##  fuelType = HYBRID
##  location.state = FL
##  vehicle.make = Lexus
##  ->  class 3.71428571428571  [0.667]
##
## Rule 458: (2/1, lift 278.6)
##  location.state = MD
##  vehicle.make = Mazda
##  ->  class 3.75  [0.500]
##
## Rule 459: (2/1, lift 278.6)
##  location.state = MO
##  vehicle.make = Mercedes-Benz
##  ->  class 3.75  [0.500]
##
## Rule 460: (3/2, lift 173.4)
##  location.state = IN
##  vehicle.make = Tesla
##  vehicle.type = car
##  ->  class 3.8  [0.400]
##
## Rule 461: (6/5, lift 108.4)
##  fuelType = GASOLINE
```

```
##  location.state = GA
##  vehicle.make = Land Rover
##  ->  class 3.8  [0.250]
##
## Rule 462: (1, lift 1300.3)
##  location.state = AZ
##  vehicle.make = GMC
##  ->  class 3.85714285714286  [0.667]
##
## Rule 463: (1, lift 866.9)
##  location.state = MD
##  vehicle.make = Acura
##  ->  class 3.90909090909091  [0.667]
##
## Rule 464: (3/2, lift 1560.4)
##  location.state = IN
##  vehicle.make = Honda
##  ->  class 3.92307692307692  [0.400]
##
## Rule 465: (1, lift 38.8)
##  location.state = MI
##  vehicle.make = Mazda
##  ->  class 4  [0.667]
##
## Rule 466: (1, lift 38.8)
##  location.state = AZ
##  vehicle.make = Alfa Romeo
##  ->  class 4  [0.667]
##
## Rule 467: (1, lift 38.8)
##  location.state = MD
##  vehicle.make = Mercedes-Benz
##  ->  class 4  [0.667]
##
## Rule 468: (1, lift 38.8)
##  location.state = CA
##  vehicle.make = Dodge
##  vehicle.type = minivan
##  ->  class 4  [0.667]
##
## Rule 469: (1, lift 38.8)
##  fuelType = HYBRID
##  location.state = GA
##  vehicle.make = Porsche
##  ->  class 4  [0.667]
##
## Rule 470: (3/1, lift 34.9)
##  location.state = VA
##  vehicle.make = Jeep
##  ->  class 4  [0.600]
##
## Rule 471: (2/1, lift 29.1)
##  location.state = TN
##  vehicle.make = Hyundai
```

```
##  ->  class 4  [0.500]
##
## Rule 472: (18/16, lift 8.7)
##  location.state = NV
##  vehicle.make = BMW
##  ->  class 4  [0.150]
##
## Rule 473: (1, lift 2600.7)
##  location.state = WA
##  vehicle.make = Ford
##  vehicle.type = truck
##  ->  class 4.08333333333333  [0.667]
##
## Rule 474: (1, lift 866.9)
##  location.state = ID
##  vehicle.make = FIAT
##  ->  class 4.14285714285714  [0.667]
##
## Rule 475: (4/3, lift 433.4)
##  location.state = FL
##  vehicle.make = MINI
##  ->  class 4.14285714285714  [0.333]
##
## Rule 476: (1, lift 650.2)
##  location.state = UT
##  vehicle.make = FIAT
##  ->  class 4.16666666666667  [0.667]
##
## Rule 477: (2/1, lift 390.1)
##  location.state = NM
##  vehicle.make = Hyundai
##  ->  class 4.2  [0.500]
##
## Rule 478: (3/2, lift 312.1)
##  location.state = NJ
##  vehicle.make = Mazda
##  ->  class 4.2  [0.400]
##
## Rule 479: (2/1, lift 1950.5)
##  location.state = FL
##  vehicle.make = Kia
##  vehicle.type = suv
##  ->  class 4.22222222222222  [0.500]
##
## Rule 480: (7/6, lift 866.9)
##  fuelType = GASOLINE
##  location.state = AZ
##  vehicle.make = Toyota
##  ->  class 4.23076923076923  [0.222]
##
## Rule 481: (1, lift 260.1)
##  location.state = NE
##  vehicle.make = Nissan
##  ->  class 4.25  [0.667]
```

```
##
## Rule 482: (1, lift 260.1)
##  location.state = OK
##  vehicle.make = Honda
##  ->  class 4.25  [0.667]
##
## Rule 483: (1, lift 260.1)
##  location.state = FL
##  vehicle.make = Mazda
##  vehicle.type = suv
##  ->  class 4.25  [0.667]
##
## Rule 484: (1, lift 1300.3)
##  location.state = AZ
##  vehicle.make = Nissan
##  vehicle.type = minivan
##  ->  class 4.27272727272727  [0.667]
##
## Rule 485: (2/1, lift 278.6)
##  location.state = IL
##  vehicle.make = FIAT
##  ->  class 4.28571428571429  [0.500]
##
## Rule 486: (1, lift 216.7)
##  location.state = MO
##  vehicle.make = Ford
##  vehicle.type = suv
##  ->  class 4.33333333333333  [0.667]
##
## Rule 487: (3/2, lift 130.0)
##  location.state = AZ
##  vehicle.make = Ford
##  vehicle.type = car
##  ->  class 4.33333333333333  [0.400]
##
## Rule 488: (1, lift 113.1)
##  location.state = MN
##  vehicle.make = Kia
##  ->  class 4.5  [0.667]
##
## Rule 489: (1, lift 113.1)
##  location.state = WV
##  vehicle.make = Toyota
##  ->  class 4.5  [0.667]
##
## Rule 490: (2/1, lift 84.8)
##  fuelType = HYBRID
##  location.state = IL
##  vehicle.make = Toyota
##  ->  class 4.5  [0.500]
##
## Rule 491: (2/1, lift 84.8)
##  fuelType = ELECTRIC
##  location.state = CA
```

```
##  vehicle.make = Chevrolet
##  ->  class 4.5  [0.500]
##
## Rule 492: (4/3, lift 56.5)
##  location.state = NV
##  vehicle.make = Honda
##  ->  class 4.5  [0.333]
##
## Rule 493: (4/3, lift 56.5)
##  location.state = PA
##  vehicle.make = Kia
##  ->  class 4.5  [0.333]
##
## Rule 494: (7/6, lift 37.7)
##  location.state = AZ
##  vehicle.make = Dodge
##  ->  class 4.5  [0.222]
##
## Rule 495: (2/1, lift 325.1)
##  location.state = MS
##  ->  class 4.6  [0.500]
##
## Rule 496: (34/32, lift 54.2)
##  fuelType = GASOLINE
##  location.state = CO
##  vehicle.make = Toyota
##  ->  class 4.6  [0.083]
##
## Rule 497: (35/33, lift 52.7)
##  location.state = CO
##  vehicle.make = Toyota
##  ->  class 4.6  [0.081]
##
## Rule 498: (1, lift 2600.7)
##  location.state = IN
##  vehicle.make = Ford
##  ->  class 4.61538461538461  [0.667]
##
## Rule 499: (4/3, lift 100.0)
##  location.state = MI
##  vehicle.make = Tesla
##  ->  class 4.66666666666667  [0.333]
##
## Rule 500: (3/2, lift 390.1)
##  location.state = WI
##  vehicle.make = Lexus
##  ->  class 4.7  [0.400]
##
## Rule 501: (4/3, lift 650.2)
##  location.state = MN
##  vehicle.make = BMW
##  ->  class 4.71428571428571  [0.333]
##
## Rule 502: (2/1, lift 177.3)
```

```
##  location.state = WI
##  vehicle.make = Ford
##  ->  class 4.75  [0.500]
##
## Rule 503: (1, lift 650.2)
##  fuelType = GASOLINE
##  location.state = AZ
##  vehicle.make = Toyota
##  vehicle.type = suv
##  ->  class 4.8  [0.667]
##
## Rule 504: (3/2, lift 390.1)
##  location.state = FL
##  vehicle.make = Dodge
##  vehicle.type = suv
##  ->  class 4.8  [0.400]
##
## Rule 505: (1, lift 38.2)
##  location.state = OK
##  vehicle.make = Kia
##  ->  class 5  [0.667]
##
## Rule 506: (1, lift 38.2)
##  location.state = HI
##  vehicle.make = BMW
##  vehicle.type = suv
##  ->  class 5  [0.667]
##
## Rule 507: (1, lift 38.2)
##  location.state = MD
##  vehicle.make = Mitsubishi
##  ->  class 5  [0.667]
##
## Rule 508: (1, lift 38.2)
##  fuelType = DIESEL
##  location.state = FL
##  vehicle.make = Volkswagen
##  ->  class 5  [0.667]
##
## Rule 509: (1, lift 38.2)
##  fuelType = DIESEL
##  location.state = CA
##  vehicle.make = Land Rover
##  ->  class 5  [0.667]
##
## Rule 510: (3/1, lift 34.4)
##  location.state = CA
##  vehicle.make in {Buick, Ram}
##  ->  class 5  [0.600]
##
## Rule 511: (2/1, lift 28.7)
##  fuelType = HYBRID
##  location.state = NV
##  ->  class 5  [0.500]
```

```
##
## Rule 512: (2/1, lift 28.7)
##  location.state = MN
##  vehicle.make = Porsche
##  ->  class 5  [0.500]
##
## Rule 513: (2/1, lift 28.7)
##  location.state = NV
##  vehicle.make = Hyundai
##  ->  class 5  [0.500]
##
## Rule 514: (10/8, lift 14.3)
##  location.state = NC
##  vehicle.make = Ford
##  ->  class 5  [0.250]
##
## Rule 515: (10/8, lift 14.3)
##  location.state = OR
##  vehicle.make = Audi
##  ->  class 5  [0.250]
##
## Rule 516: (17/15, lift 9.1)
##  location.state = LA
##  ->  class 5  [0.158]
##
## Rule 517: (19/17, lift 8.2)
##  location.state = OR
##  vehicle.make = Tesla
##  ->  class 5  [0.143]
##
## Rule 518: (1, lift 325.1)
##  location.state = KY
##  vehicle.make = Audi
##  ->  class 5.2  [0.667]
##
## Rule 519: (3/2, lift 195.1)
##  location.state = NV
##  vehicle.make = Kia
##  ->  class 5.2  [0.400]
##
## Rule 520: (6/5, lift 121.9)
##  location.state = CA
##  vehicle.make = Lexus
##  vehicle.type = suv
##  ->  class 5.2  [0.250]
##
## Rule 521: (3/2, lift 260.1)
##  location.state = NJ
##  vehicle.make = Toyota
##  vehicle.type = suv
##  ->  class 5.28571428571429  [0.400]
##
## Rule 522: (1, lift 200.1)
##  fuelType = DIESEL
```

```
##  location.state = GA
##  vehicle.make = Porsche
##  ->  class 5.33333333333333  [0.667]
##
## Rule 523: (1, lift 866.9)
##  location.state = WI
##  vehicle.make = Honda
##  vehicle.type = suv
##  ->  class 5.375  [0.667]
##
## Rule 524: (1, lift 153.0)
##  location.state = UT
##  vehicle.make = Acura
##  ->  class 5.5  [0.667]
##
## Rule 525: (2/1, lift 487.6)
##  fuelType = DIESEL
##  location.state = PA
##  vehicle.make = Volkswagen
##  ->  class 5.6  [0.500]
##
## Rule 526: (2/1, lift 130.0)
##  location.state = CO
##  vehicle.make = Nissan
##  vehicle.type = car
##  ->  class 5.66666666666667  [0.500]
##
## Rule 527: (3/2, lift 104.0)
##  location.state = NJ
##  vehicle.make = Dodge
##  ->  class 5.66666666666667  [0.400]
##
## Rule 528: (1, lift 433.4)
##  location.state = MO
##  vehicle.make = Mazda
##  ->  class 5.75  [0.667]
##
## Rule 529: (1, lift 2600.7)
##  location.state = GA
##  vehicle.make = Volkswagen
##  vehicle.type = suv
##  ->  class 5.77777777777778  [0.667]
##
## Rule 530: (2/1, lift 650.2)
##  location.state = OR
##  vehicle.make = MINI
##  ->  class 5.8  [0.500]
##
## Rule 531: (2/1, lift 975.2)
##  location.state = MN
##  vehicle.make = Audi
##  ->  class 5.88888888888889  [0.500]
##
## Rule 532: (1, lift 44.8)
```

```
##  fuelType = HYBRID
##  location.state = IN
##  vehicle.make = Toyota
##  ->  class 6  [0.667]
##
## Rule 533: (1, lift 44.8)
##  fuelType = ELECTRIC
##  location.state = RI
##  ->  class 6  [0.667]
##
## Rule 534: (1, lift 44.8)
##  location.state = MO
##  vehicle.make = BMW
##  ->  class 6  [0.667]
##
## Rule 535: (1, lift 44.8)
##  location.state = KY
##  vehicle.make = Honda
##  ->  class 6  [0.667]
##
## Rule 536: (1, lift 44.8)
##  location.state = IA
##  vehicle.make = Toyota
##  ->  class 6  [0.667]
##
## Rule 537: (1, lift 44.8)
##  fuelType = DIESEL
##  location.state = WA
##  vehicle.make = Volkswagen
##  ->  class 6  [0.667]
##
## Rule 538: (3/2, lift 26.9)
##  location.state = FL
##  vehicle.make = Alfa Romeo
##  ->  class 6  [0.400]
##
## Rule 539: (2/1, lift 975.2)
##  location.state = MN
##  vehicle.make = BMW
##  vehicle.type = suv
##  ->  class 6.14285714285714  [0.500]
##
## Rule 540: (1, lift 2600.7)
##  location.state = NV
##  vehicle.make = Nissan
##  ->  class 6.18181818181818  [0.667]
##
## Rule 541: (1, lift 866.9)
##  location.state = SC
##  vehicle.make = Jeep
##  ->  class 6.2  [0.667]
##
## Rule 542: (1, lift 866.9)
##  location.state = FL
```

```
##  vehicle.make = smart
##  ->  class 6.2  [0.667]
##
## Rule 543: (2/1, lift 650.2)
##  location.state = NE
##  vehicle.make = Jeep
##  ->  class 6.2  [0.500]
##
## Rule 544: (2/1, lift 243.8)
##  fuelType = HYBRID
##  location.state = NJ
##  vehicle.make = BMW
##  ->  class 6.25  [0.500]
##
## Rule 545: (3/2, lift 195.1)
##  location.state = NV
##  vehicle.make = Subaru
##  ->  class 6.25  [0.400]
##
## Rule 546: (1, lift 2600.7)
##  fuelType = HYBRID
##  location.state = CA
##  vehicle.make = Porsche
##  ->  class 6.28571428571429  [0.667]
##
## Rule 547: (1, lift 2600.7)
##  location.state = NM
##  vehicle.make = Chevrolet
##  ->  class 6.3  [0.667]
##
## Rule 548: (1, lift 2600.7)
##  location.state = OK
##  vehicle.make = MINI
##  ->  class 6.44444444444444  [0.667]
##
## Rule 549: (1, lift 162.5)
##  location.state = FL
##  vehicle.make = Jaguar
##  vehicle.type = suv
##  ->  class 6.5  [0.667]
##
## Rule 550: (2/1, lift 121.9)
##  location.state = WA
##  vehicle.make = Chevrolet
##  vehicle.type = car
##  ->  class 6.5  [0.500]
##
## Rule 551: (1, lift 371.5)
##  fuelType = ELECTRIC
##  location.state = MN
##  vehicle.make = Chevrolet
##  ->  class 6.66666666666667  [0.667]
##
## Rule 552: (2/1, lift 278.6)
```

```
##  location.state = KS
##  vehicle.make = Chevrolet
##  ->  class 6.66666666666667  [0.500]
##
## Rule 553: (1, lift 2600.7)
##  location.state = NC
##  vehicle.make = Scion
##  ->  class 6.7  [0.667]
##
## Rule 554: (1, lift 2600.7)
##  vehicle.make = Saab
##  ->  class 6.77777777777778  [0.667]
##
## Rule 555: (2/1, lift 1950.5)
##  location.state = AZ
##  vehicle.make = Dodge
##  vehicle.type = car
##  ->  class 6.90909090909091  [0.500]
##
## Rule 556: (1, lift 43.3)
##  fuelType = HYBRID
##  location.state = NC
##  vehicle.make = Toyota
##  ->  class 7  [0.667]
##
## Rule 557: (1, lift 43.3)
##  location.state = FL
##  vehicle.make = Nissan
##  vehicle.type = minivan
##  ->  class 7  [0.667]
##
## Rule 558: (1, lift 43.3)
##  fuelType = HYBRID
##  location.state = HI
##  vehicle.make = Ford
##  ->  class 7  [0.667]
##
## Rule 559: (1, lift 43.3)
##  fuelType = HYBRID
##  location.state = CO
##  vehicle.make = Chevrolet
##  ->  class 7  [0.667]
##
## Rule 560: (1, lift 43.3)
##  location.state = CA
##  vehicle.make = Scion
##  ->  class 7  [0.667]
##
## Rule 561: (1, lift 43.3)
##  location.state = WA
##  vehicle.make = Acura
##  ->  class 7  [0.667]
##
## Rule 562: (1, lift 43.3)
```

```
##  location.state = MA
##  vehicle.make = Jaguar
##  ->  class 7  [0.667]
##
## Rule 563: (2/1, lift 32.5)
##  fuelType = HYBRID
##  location.state = HI
##  vehicle.make = BMW
##  ->  class 7  [0.500]
##
## Rule 564: (2/1, lift 32.5)
##  location.state = WI
##  vehicle.make = Mazda
##  ->  class 7  [0.500]
##
## Rule 565: (7/5, lift 21.7)
##  location.state = NV
##  vehicle.make = Jeep
##  ->  class 7  [0.333]
##
## Rule 566: (4/3, lift 21.7)
##  location.state = KY
##  vehicle.make = Tesla
##  ->  class 7  [0.333]
##
## Rule 567: (11/9, lift 15.0)
##  location.state = NJ
##  vehicle.make = Chevrolet
##  ->  class 7  [0.231]
##
## Rule 568: (11/9, lift 15.0)
##  location.state = UT
##  vehicle.make = Jeep
##  ->  class 7  [0.231]
##
## Rule 569: (1, lift 2600.7)
##  location.state = KS
##  vehicle.make = Mercedes-Benz
##  ->  class 7.1  [0.667]
##
## Rule 570: (1, lift 2600.7)
##  location.state = TN
##  vehicle.make = Pontiac
##  ->  class 7.23076923076923  [0.667]
##
## Rule 571: (2/1, lift 325.1)
##  location.state = OK
##  vehicle.make = Ford
##  vehicle.type = suv
##  ->  class 7.25  [0.500]
##
## Rule 572: (4/3, lift 216.7)
##  location.state = NV
##  vehicle.make = Ford
```

```
##  vehicle.type = car
##  ->  class 7.25  [0.333]
##
## Rule 573: (2/1, lift 1950.5)
##  location.state = VA
##  vehicle.make = Audi
##  ->  class 7.42857142857143  [0.500]
##
## Rule 574: (1, lift 236.4)
##  location.state = AZ
##  vehicle.make = Polaris
##  ->  class 7.5  [0.667]
##
## Rule 575: (1, lift 236.4)
##  fuelType = DIESEL
##  vehicle.make = GMC
##  ->  class 7.5  [0.667]
##
## Rule 576: (5/4, lift 101.3)
##  location.state = MA
##  vehicle.make = Tesla
##  vehicle.type = car
##  ->  class 7.5  [0.286]
##
## Rule 577: (7/6, lift 78.8)
##  location.state = GA
##  vehicle.make = Nissan
##  ->  class 7.5  [0.222]
##
## Rule 578: (1, lift 289.0)
##  location.state = AZ
##  vehicle.make = Volkswagen
##  ->  class 7.66666666666667  [0.667]
##
## Rule 579: (4/3, lift 144.5)
##  location.state = TN
##  vehicle.make = Chevrolet
##  vehicle.type = car
##  ->  class 7.66666666666667  [0.333]
##
## Rule 580: (32/30, lift 38.2)
##  location.state = CA
##  vehicle.make = Porsche
##  vehicle.type = car
##  ->  class 7.66666666666667  [0.088]
##
## Rule 581: (1, lift 1300.3)
##  fuelType = HYBRID
##  location.state = AZ
##  vehicle.type = suv
##  ->  class 7.71428571428571  [0.667]
##
## Rule 582: (1, lift 433.4)
##  location.state = LA
```

```
##  vehicle.make = Maserati
##  ->  class 7.75  [0.667]
##
## Rule 583: (1, lift 56.5)
##  location.state = OR
##  vehicle.make = Alfa Romeo
##  ->  class 8  [0.667]
##
## Rule 584: (1, lift 56.5)
##  location.state = AZ
##  vehicle.make = Dodge
##  vehicle.type = truck
##  ->  class 8  [0.667]
##
## Rule 585: (3/2, lift 33.9)
##  location.state = KS
##  vehicle.make = Tesla
##  ->  class 8  [0.400]
##
## Rule 586: (3/2, lift 33.9)
##  location.state = MI
##  vehicle.make = Toyota
##  ->  class 8  [0.400]
##
## Rule 587: (17/15, lift 13.4)
##  location.state = CO
##  vehicle.make = Tesla
##  ->  class 8  [0.158]
##
## Rule 588: (42/39, lift 7.7)
##  location.state = FL
##  vehicle.make = BMW
##  ->  class 8  [0.091]
##
## Rule 589: (1, lift 2600.7)
##  location.state = AZ
##  vehicle.make = Buick
##  ->  class 8.1  [0.667]
##
## Rule 590: (3/2, lift 1560.4)
##  location.state = MD
##  vehicle.make = Toyota
##  vehicle.type = car
##  ->  class 8.30769230769231  [0.400]
##
## Rule 591: (1, lift 520.1)
##  location.state = VA
##  vehicle.make = Acura
##  ->  class 8.33333333333333  [0.667]
##
## Rule 592: (1, lift 520.1)
##  location.state = LA
##  vehicle.make = Kia
##  ->  class 8.33333333333333  [0.667]
```

```
##
## Rule 593: (1, lift 153.0)
##  location.state = GA
##  vehicle.make = Nissan
##  vehicle.type = minivan
##  ->  class 8.5  [0.667]
##
## Rule 594: (1, lift 153.0)
##  location.state = OR
##  vehicle.make = Ram
##  ->  class 8.5  [0.667]
##
## Rule 595: (1, lift 153.0)
##  location.state = CO
##  vehicle.make = Subaru
##  vehicle.type = car
##  ->  class 8.5  [0.667]
##
## Rule 596: (1, lift 153.0)
##  location.state = GA
##  vehicle.make = Mitsubishi
##  ->  class 8.5  [0.667]
##
## Rule 597: (1, lift 153.0)
##  location.state = MN
##  vehicle.make = Volkswagen
##  ->  class 8.5  [0.667]
##
## Rule 598: (2/1, lift 114.7)
##  location.state = SC
##  vehicle.make = FIAT
##  ->  class 8.5  [0.500]
##
## Rule 599: (3/2, lift 91.8)
##  location.state = MA
##  vehicle.make = Toyota
##  ->  class 8.5  [0.400]
##
## Rule 600: (39/37, lift 16.8)
##  location.state = FL
##  vehicle.make = Ford
##  vehicle.type = car
##  ->  class 8.5  [0.073]
##
## Rule 601: (1, lift 1300.3)
##  location.state = MD
##  vehicle.make = Nissan
##  ->  class 8.57142857142857  [0.667]
##
## Rule 602: (10/8, lift 121.9)
##  location.state = DC
##  ->  class 8.75  [0.250]
##
## Rule 603: (1, lift 63.4)
```

```
##  fuelType = ELECTRIC
##  location.state = CA
##  vehicle.make = Nissan
##  ->  class 9  [0.667]
##
## Rule 604: (1, lift 63.4)
##  location.state = MO
##  vehicle.make = Cadillac
##  ->  class 9  [0.667]
##
## Rule 605: (1, lift 63.4)
##  location.state = GA
##  vehicle.make = Dodge
##  vehicle.type = suv
##  ->  class 9  [0.667]
##
## Rule 606: (2/1, lift 47.6)
##  location.state = HI
##  vehicle.make = Jeep
##  vehicle.type = truck
##  ->  class 9  [0.500]
##
## Rule 607: (20/17, lift 17.3)
##  fuelType = GASOLINE
##  location.state = CA
##  vehicle.make = Toyota
##  vehicle.type = car
##  ->  class 9  [0.182]
##
## Rule 608: (153/147, lift 4.3)
##  location.state = GA
##  ->  class 9  [0.045]
##
## Rule 609: (2/1, lift 325.1)
##  location.state = OR
##  vehicle.make = Volkswagen
##  vehicle.type = car
##  ->  class 9.2  [0.500]
##
## Rule 610: (1, lift 433.4)
##  location.state = AZ
##  vehicle.make = MINI
##  ->  class 9.33333333333333  [0.667]
##
## Rule 611: (1, lift 289.0)
##  location.state = MD
##  vehicle.make = Dodge
##  ->  class 9.5  [0.667]
##
## Rule 612: (1, lift 520.1)
##  location.state = LA
##  vehicle.make = Cadillac
##  ->  class 9.66666666666667  [0.667]
##
```

```
## Rule 613: (2/1, lift 390.1)
##  location.state = NE
##  vehicle.make = Ford
##  ->  class 9.66666666666667  [0.500]
##
## Rule 614: (11/9, lift 225.1)
##  location.state = GA
##  vehicle.make = Mercedes-Benz
##  vehicle.type = car
##  ->  class 9.75  [0.231]
##
## Rule 615: (1, lift 76.5)
##  location.state = KS
##  vehicle.make = Toyota
##  ->  class 10  [0.667]
##
## Rule 616: (1, lift 76.5)
##  fuelType = DIESEL
##  location.state = SC
##  vehicle.make = Mercedes-Benz
##  ->  class 10  [0.667]
##
## Rule 617: (1, lift 76.5)
##  location.state = NE
##  vehicle.make = Hyundai
##  ->  class 10  [0.667]
##
## Rule 618: (1, lift 76.5)
##  location.state = ID
##  vehicle.make = Tesla
##  ->  class 10  [0.667]
##
## Rule 619: (1, lift 76.5)
##  location.state = KY
##  vehicle.make = Lexus
##  ->  class 10  [0.667]
##
## Rule 620: (2/1, lift 57.4)
##  location.state = AZ
##  vehicle.make = Nissan
##  vehicle.type = suv
##  ->  class 10  [0.500]
##
## Rule 621: (2/1, lift 57.4)
##  location.state = CO
##  vehicle.make = Lexus
##  ->  class 10  [0.500]
##
## Rule 622: (4/3, lift 38.2)
##  location.state = CA
##  vehicle.make = Ford
##  vehicle.type = suv
##  ->  class 10  [0.333]
##
```

```
## Rule 623: (5/4, lift 32.8)
##  location.state = CA
##  vehicle.make = Chrysler
##  ->  class 10  [0.286]
##
## Rule 624: (1, lift 1300.3)
##  location.state = UT
##  vehicle.make = Chevrolet
##  vehicle.type = car
##  ->  class 10.1666666666667  [0.667]
##
## Rule 625: (1, lift 325.1)
##  location.state = MN
##  vehicle.make = smart
##  ->  class 10.3333333333333  [0.667]
##
## Rule 626: (5/4, lift 139.3)
##  location.state = AZ
##  vehicle.make = Kia
##  ->  class 10.3333333333333  [0.286]
##
## Rule 627: (1, lift 200.1)
##  location.state = IL
##  vehicle.make = Toyota
##  vehicle.type = suv
##  ->  class 10.5  [0.667]
##
## Rule 628: (1, lift 200.1)
##  location.state = ME
##  vehicle.make = Tesla
##  ->  class 10.5  [0.667]
##
## Rule 629: (2/1, lift 150.0)
##  location.state = AZ
##  vehicle.make = Kia
##  vehicle.type = suv
##  ->  class 10.5  [0.500]
##
## Rule 630: (3/2, lift 120.0)
##  location.state = MA
##  vehicle.make = Audi
##  ->  class 10.5  [0.400]
##
## Rule 631: (1, lift 650.2)
##  fuelType = HYBRID
##  location.state = GA
##  vehicle.make = Lexus
##  ->  class 10.6  [0.667]
##
## Rule 632: (1, lift 650.2)
##  location.state = FL
##  vehicle.make = Acura
##  ->  class 10.6  [0.667]
##
```

```
## Rule 633: (3/2, lift 1560.4)
##  location.state = NC
##  vehicle.make = Volkswagen
##  ->  class 10.7142857142857  [0.400]
##
## Rule 634: (1, lift 520.1)
##  fuelType = HYBRID
##  location.state = OR
##  vehicle.make = Lexus
##  ->  class 10.8  [0.667]
##
## Rule 635: (1, lift 2600.7)
##  location.state = MN
##  vehicle.make = Pontiac
##  ->  class 10.8181818181818  [0.667]
##
## Rule 636: (1, lift 81.3)
##  location.state = MN
##  vehicle.make = Honda
##  vehicle.type = minivan
##  ->  class 11  [0.667]
##
## Rule 637: (1, lift 81.3)
##  location.state = MA
##  vehicle.make = Acura
##  ->  class 11  [0.667]
##
## Rule 638: (3/2, lift 48.8)
##  location.state = PA
##  vehicle.make = Toyota
##  vehicle.type = car
##  ->  class 11  [0.400]
##
## Rule 639: (7/5, lift 40.6)
##  location.state = HI
##  vehicle.make = Tesla
##  ->  class 11  [0.333]
##
## Rule 640: (1, lift 2600.7)
##  location.state = HI
##  vehicle.make = Scion
##  ->  class 11.1428571428571  [0.667]
##
## Rule 641: (1, lift 371.5)
##  location.state = TN
##  vehicle.make = Subaru
##  ->  class 11.3333333333333  [0.667]
##
## Rule 642: (1, lift 2600.7)
##  location.state = OR
##  vehicle.make = FIAT
##  ->  class 11.4285714285714  [0.667]
##
## Rule 643: (1, lift 371.5)
```

```
##  location.state = KY
##  vehicle.make = Mitsubishi
##  ->  class 11.5  [0.667]
##
## Rule 644: (5/4, lift 159.2)
##  location.state = PA
##  vehicle.make = Honda
##  vehicle.type = car
##  ->  class 11.5  [0.286]
##
## Rule 645: (1, lift 1300.3)
##  location.state = AZ
##  vehicle.make = Chrysler
##  ->  class 11.7142857142857  [0.667]
##
## Rule 646: (2/1, lift 975.2)
##  location.state = OR
##  vehicle.make = Lexus
##  vehicle.type = suv
##  ->  class 11.7777777777778  [0.500]
##
## Rule 647: (1, lift 650.2)
##  location.state = IA
##  vehicle.make = Hyundai
##  ->  class 11.8  [0.667]
##
## Rule 648: (1, lift 650.2)
##  location.state = MA
##  vehicle.make = Porsche
##  ->  class 11.8  [0.667]
##
## Rule 649: (1, lift 86.7)
##  location.state = KY
##  vehicle.make = Nissan
##  ->  class 12  [0.667]
##
## Rule 650: (1, lift 86.7)
##  location.state = FL
##  vehicle.make = Jeep
##  vehicle.type = truck
##  ->  class 12  [0.667]
##
## Rule 651: (1, lift 86.7)
##  location.state = UT
##  vehicle.make = Honda
##  vehicle.type = car
##  ->  class 12  [0.667]
##
## Rule 652: (1, lift 86.7)
##  fuelType = HYBRID
##  location.state = WA
##  vehicle.make = Hyundai
##  ->  class 12  [0.667]
##
```

```
## Rule 653: (3/2, lift 52.0)
##  location.state = UT
##  vehicle.make = Buick
##  ->  class 12  [0.400]
##
## Rule 654: (2/1, lift 1950.5)
##  location.state = CA
##  vehicle.make = Toyota
##  vehicle.type = minivan
##  ->  class 12.1428571428571  [0.500]
##
## Rule 655: (2/1, lift 975.2)
##  location.state = AZ
##  vehicle.make = Kia
##  vehicle.type = car
##  ->  class 12.25  [0.500]
##
## Rule 656: (5/4, lift 139.3)
##  location.state = AZ
##  vehicle.make = Hyundai
##  ->  class 12.5  [0.286]
##
## Rule 657: (1, lift 520.1)
##  location.state = ID
##  vehicle.make = Jeep
##  ->  class 12.6  [0.667]
##
## Rule 658: (2/1, lift 390.1)
##  location.state = OR
##  vehicle.make = Toyota
##  vehicle.type = minivan
##  ->  class 12.6  [0.500]
##
## Rule 659: (1, lift 2600.7)
##  fuelType = DIESEL
##  location.state = CO
##  vehicle.make = BMW
##  ->  class 12.8  [0.667]
##
## Rule 660: (1, lift 89.7)
##  location.state = GA
##  vehicle.make = Nissan
##  vehicle.type = suv
##  ->  class 13  [0.667]
##
## Rule 661: (1, lift 89.7)
##  location.state = WA
##  vehicle.make = Chrysler
##  ->  class 13  [0.667]
##
## Rule 662: (1, lift 89.7)
##  location.state = MA
##  vehicle.make = Alfa Romeo
##  ->  class 13  [0.667]
```

```
##
## Rule 663: (3/2, lift 53.8)
##  location.state = MI
##  vehicle.make = Tesla
##  vehicle.type = car
##  ->  class 13  [0.400]
##
## Rule 664: (3/2, lift 390.1)
##  location.state = AZ
##  vehicle.make = Hyundai
##  vehicle.type = car
##  ->  class 13.25  [0.400]
##
## Rule 665: (1, lift 1300.3)
##  location.state = VA
##  vehicle.make = Honda
##  vehicle.type = suv
##  ->  class 13.375  [0.667]
##
## Rule 666: (1, lift 289.0)
##  location.state = IN
##  vehicle.make = Chevrolet
##  vehicle.type = car
##  ->  class 13.5  [0.667]
##
## Rule 667: (1, lift 74.3)
##  location.state = SC
##  vehicle.make = Ford
##  ->  class 14  [0.667]
##
## Rule 668: (1, lift 74.3)
##  location.state = OK
##  vehicle.make = Chevrolet
##  vehicle.type = car
##  ->  class 14  [0.667]
##
## Rule 669: (4/2, lift 55.7)
##  location.state = MI
##  vehicle.make in {BMW, Jeep}
##  ->  class 14  [0.500]
##
## Rule 670: (2/1, lift 55.7)
##  location.state = HI
##  vehicle.make = Honda
##  vehicle.type = suv
##  ->  class 14  [0.500]
##
## Rule 671: (2/1, lift 55.7)
##  location.state = NJ
##  vehicle.make = Jaguar
##  vehicle.type = car
##  ->  class 14  [0.500]
##
## Rule 672: (3/2, lift 44.6)
```

```
##  location.state = NV
##  vehicle.make = Honda
##  vehicle.type = car
##  ->  class 14  [0.400]
##
## Rule 673: (3/2, lift 44.6)
##  location.state = AZ
##  vehicle.make = Ford
##  vehicle.type = suv
##  ->  class 14  [0.400]
##
## Rule 674: (4/3, lift 37.2)
##  location.state = MA
##  vehicle.make = Jeep
##  ->  class 14  [0.333]
##
## Rule 675: (1, lift 1300.3)
##  location.state = NV
##  vehicle.make = Chrysler
##  ->  class 14.2  [0.667]
##
## Rule 676: (1, lift 371.5)
##  location.state = MA
##  vehicle.make = BMW
##  ->  class 14.5  [0.667]
##
## Rule 677: (1, lift 2600.7)
##  location.state = AZ
##  vehicle.make = Mazda
##  ->  class 14.8571428571429  [0.667]
##
## Rule 678: (1, lift 153.0)
##  location.state = MO
##  vehicle.make = Tesla
##  ->  class 15  [0.667]
##
## Rule 679: (13/11, lift 390.1)
##  location.state = HI
##  vehicle.make = Ford
##  vehicle.type = car
##  ->  class 15.25  [0.200]
##
## Rule 680: (1, lift 866.9)
##  location.state = MD
##  vehicle.make = Lexus
##  ->  class 15.6666666666667  [0.667]
##
## Rule 681: (4/3, lift 433.4)
##  location.state = NJ
##  vehicle.make = Nissan
##  ->  class 15.8  [0.333]
##
## Rule 682: (1, lift 130.0)
##  location.state = NJ
```

```
##  vehicle.make = Kia
##  ->  class 16  [0.667]
##
## Rule 683: (1, lift 130.0)
##  location.state = TN
##  vehicle.make = Tesla
##  vehicle.type = suv
##  ->  class 16  [0.667]
##
## Rule 684: (1, lift 130.0)
##  location.state = VA
##  vehicle.make = Kia
##  vehicle.type = suv
##  ->  class 16  [0.667]
##
## Rule 685: (2/1, lift 97.5)
##  location.state = PA
##  vehicle.make = Dodge
##  vehicle.type = truck
##  ->  class 16  [0.500]
##
## Rule 686: (10/8, lift 48.8)
##  location.state = UT
##  vehicle.make = Kia
##  ->  class 16  [0.250]
##
## Rule 687: (12/10, lift 41.8)
##  location.state = FL
##  vehicle.make = Maserati
##  ->  class 16  [0.214]
##
## Rule 688: (1, lift 236.4)
##  location.state = PA
##  vehicle.make = Lexus
##  ->  class 16.5  [0.667]
##
## Rule 689: (1, lift 2600.7)
##  fuelType = GASOLINE
##  location.state = DE
##  vehicle.type = car
##  ->  class 16.6  [0.667]
##
## Rule 690: (1, lift 104.0)
##  location.state = MO
##  vehicle.make = Hyundai
##  ->  class 17  [0.667]
##
## Rule 691: (1, lift 104.0)
##  location.state = MD
##  vehicle.make = Volvo
##  ->  class 17  [0.667]
##
## Rule 692: (1, lift 104.0)
##  location.state = KY
```

```
##  vehicle.make = Tesla
##  vehicle.type = suv
##  ->  class 17  [0.667]
##
## Rule 693: (1, lift 104.0)
##  location.state = VA
##  vehicle.make = Tesla
##  vehicle.type = suv
##  ->  class 17  [0.667]
##
## Rule 694: (3/2, lift 62.4)
##  location.state = GA
##  vehicle.make = Dodge
##  vehicle.type = car
##  ->  class 17  [0.400]
##
## Rule 695: (1, lift 650.2)
##  fuelType = HYBRID
##  location.state = DE
##  ->  class 17.3333333333333  [0.667]
##
## Rule 696: (1, lift 371.5)
##  location.state = NM
##  vehicle.make = Nissan
##  ->  class 17.5  [0.667]
##
## Rule 697: (1, lift 650.2)
##  location.state = GA
##  vehicle.make = Mazda
##  ->  class 17.75  [0.667]
##
## Rule 698: (2/1, lift 97.5)
##  location.state = ID
##  vehicle.make = Kia
##  ->  class 18  [0.500]
##
## Rule 699: (2/1, lift 84.8)
##  location.state = MD
##  vehicle.make = Jeep
##  ->  class 19  [0.500]
##
## Rule 700: (7/6, lift 37.7)
##  location.state = GA
##  vehicle.make = Volkswagen
##  ->  class 19  [0.222]
##
## Rule 701: (1, lift 1300.3)
##  location.state = NV
##  vehicle.make = Mercedes-Benz
##  vehicle.type = minivan
##  ->  class 19.25  [0.667]
##
## Rule 702: (2/1, lift 1950.5)
##  fuelType = HYBRID
```

```
##  location.state = VA
##  vehicle.make = Chevrolet
##  ->  class 19.2857142857143  [0.500]
##
## Rule 703: (1, lift 216.7)
##  fuelType = DIESEL
##  location.state = NJ
##  vehicle.make = Land Rover
##  ->  class 19.5  [0.667]
##
## Rule 704: (1, lift 650.2)
##  location.state = GA
##  vehicle.make = Ford
##  vehicle.type = suv
##  ->  class 19.6666666666667  [0.667]
##
## Rule 705: (1, lift 89.7)
##  location.state = CA
##  vehicle.make = Kia
##  vehicle.type = minivan
##  ->  class 20  [0.667]
##
## Rule 706: (1, lift 89.7)
##  location.state = WA
##  vehicle.make = Alfa Romeo
##  ->  class 20  [0.667]
##
## Rule 707: (2/1, lift 67.3)
##  location.state = NJ
##  vehicle.make = Ford
##  vehicle.type = truck
##  ->  class 20  [0.500]
##
## Rule 708: (2/1, lift 278.6)
##  location.state = AZ
##  vehicle.make = Dodge
##  vehicle.type = suv
##  ->  class 20.5  [0.500]
##
## Rule 709: (1, lift 1300.3)
##  location.state = WI
##  vehicle.make = GMC
##  ->  class 20.6  [0.667]
##
## Rule 710: (3/2, lift 67.8)
##  location.state = CA
##  vehicle.make = Lamborghini
##  ->  class 21  [0.400]
##
## Rule 711: (1, lift 866.9)
##  fuelType = GASOLINE
##  location.state = GA
##  vehicle.make = Porsche
##  vehicle.type = suv
```

```
##  ->  class 21.3333333333333  [0.667]
##
## Rule 712: (1, lift 866.9)
##  location.state = LA
##  vehicle.make = Mitsubishi
##  ->  class 21.5  [0.667]
##
## Rule 713: (1, lift 866.9)
##  location.state = ME
##  vehicle.make = Nissan
##  ->  class 21.8  [0.667]
##
## Rule 714: (2, lift 154.0)
##  location.state = WI
##  vehicle.make in {Chevrolet, Jeep}
##  ->  class 22  [0.750]
##
## Rule 715: (1, lift 136.9)
##  location.state = NC
##  vehicle.make = Tesla
##  vehicle.type = suv
##  ->  class 22  [0.667]
##
## Rule 716: (1, lift 520.1)
##  location.state = NJ
##  vehicle.make = Volvo
##  ->  class 22.5  [0.667]
##
## Rule 717: (1, lift 2600.7)
##  location.state = HI
##  vehicle.make = Kia
##  ->  class 22.8333333333333  [0.667]
##
## Rule 718: (1, lift 371.5)
##  location.state = FL
##  vehicle.make = Dodge
##  vehicle.type = truck
##  ->  class 23  [0.667]
##
## Rule 719: (1, lift 185.8)
##  fuelType = HYBRID
##  location.state = CA
##  vehicle.make = Lexus
##  vehicle.type = suv
##  ->  class 24  [0.667]
##
## Rule 720: (2/1, lift 325.1)
##  location.state = HI
##  vehicle.make = Tesla
##  vehicle.type = suv
##  ->  class 24.5  [0.500]
##
## Rule 721: (2/1, lift 325.1)
##  location.state = MA
```

```
##  vehicle.make = Nissan
##  ->  class 24.5  [0.500]
##
## Rule 722: (1, lift 2600.7)
##  vehicle.make = Suzuki
##  ->  class 24.8571428571429  [0.667]
##
## Rule 723: (1, lift 2600.7)
##  fuelType = HYBRID
##  location.state = AZ
##  vehicle.make = Honda
##  ->  class 25.375  [0.667]
##
## Rule 724: (1, lift 325.1)
##  location.state = WA
##  vehicle.make = Jeep
##  vehicle.type = truck
##  ->  class 26  [0.667]
##
## Rule 725: (1, lift 520.1)
##  fuelType = ELECTRIC
##  location.state = VA
##  vehicle.make = Chevrolet
##  ->  class 26.6666666666667  [0.667]
##
## Rule 726: (1, lift 2600.7)
##  location.state = MI
##  vehicle.make = Chrysler
##  ->  class 26.75  [0.667]
##
## Rule 727: (1, lift 260.1)
##  fuelType = GASOLINE
##  location.state = FL
##  vehicle.make = Tesla
##  ->  class 27  [0.667]
##
## Rule 728: (1, lift 2600.7)
##  location.state = NJ
##  vehicle.make = Ford
##  vehicle.type = car
##  ->  class 28.2  [0.667]
##
## Rule 729: (1, lift 2600.7)
##  location.state = OR
##  vehicle.make = Scion
##  ->  class 28.2142857142857  [0.667]
##
## Rule 730: (1, lift 173.4)
##  location.state = VA
##  vehicle.make = Hyundai
##  vehicle.type = suv
##  ->  class 29  [0.667]
##
## Rule 731: (2/1, lift 130.0)
```

```
##  location.state = CO
##  vehicle.make = Hyundai
##  ->  class 29  [0.500]
##
## Rule 732: (1, lift 216.7)
##  location.state = IN
##  vehicle.make = Hyundai
##  ->  class 30  [0.667]
##
## Rule 733: (2/1, lift 975.2)
##  location.state = CA
##  vehicle.make = Dodge
##  vehicle.type = suv
##  ->  class 30.3333333333333  [0.500]
##
## Rule 734: (1, lift 2600.7)
##  location.state = NJ
##  vehicle.make = Honda
##  vehicle.type = minivan
##  ->  class 30.7142857142857  [0.667]
##
## Rule 735: (1, lift 371.5)
##  location.state = IL
##  vehicle.make = Volvo
##  ->  class 31  [0.667]
##
## Rule 736: (1, lift 371.5)
##  location.state = WA
##  vehicle.make = Subaru
##  vehicle.type = car
##  ->  class 31  [0.667]
##
## Rule 737: (66/63, lift 16.4)
##  fuelType = GASOLINE
##  location.state = CA
##  vehicle.make = BMW
##  ->  class 32  [0.059]
##
## Rule 738: (78/75, lift 13.9)
##  location.state = CA
##  vehicle.make = BMW
##  ->  class 32  [0.050]
##
## Rule 739: (1, lift 2600.7)
##  location.state = WA
##  vehicle.make = Honda
##  vehicle.type = suv
##  ->  class 32.6  [0.667]
##
## Rule 740: (2/1, lift 1950.5)
##  location.state = NJ
##  vehicle.make = Nissan
##  vehicle.type = suv
##  ->  class 32.6666666666667  [0.500]
```

```
##
## Rule 741: (1, lift 236.4)
##  location.state = NC
##  vehicle.make = Jeep
##  vehicle.type = truck
##  ->  class 33  [0.667]
##
## Rule 742: (1, lift 236.4)
##  location.state = MA
##  vehicle.make = GMC
##  ->  class 33  [0.667]
##
## Rule 743: (1, lift 236.4)
##  location.state = IL
##  vehicle.make = Acura
##  ->  class 33  [0.667]
##
## Rule 744: (1, lift 216.7)
##  location.state = NE
##  vehicle.make = Tesla
##  ->  class 34  [0.667]
##
## Rule 745: (14/12, lift 61.0)
##  location.state = CA
##  vehicle.make = Dodge
##  ->  class 34  [0.188]
##
## Rule 746: (2/1, lift 650.2)
##  location.state = NJ
##  vehicle.make = Alfa Romeo
##  ->  class 35.6666666666667  [0.500]
##
## Rule 747: (1, lift 185.8)
##  location.state = UT
##  vehicle.make = BMW
##  ->  class 38  [0.667]
##
## Rule 748: (1, lift 185.8)
##  location.state = MN
##  vehicle.make = Hyundai
##  vehicle.type = suv
##  ->  class 38  [0.667]
##
## Rule 749: (1, lift 185.8)
##  location.state = NJ
##  vehicle.make = Audi
##  ->  class 38  [0.667]
##
## Rule 750: (1, lift 1300.3)
##  location.state = NV
##  vehicle.make = Toyota
##  vehicle.type = suv
##  ->  class 38.5  [0.667]
##
```

```
## Rule 751: (1, lift 2600.7)
##  location.state = WA
##  vehicle.make = Lexus
##  ->  class 39.3333333333333  [0.667]
##
## Rule 752: (1, lift 200.1)
##  location.state = VA
##  vehicle.make = Mazda
##  vehicle.type = suv
##  ->  class 40  [0.667]
##
## Rule 753: (2/1, lift 150.0)
##  location.state = NJ
##  vehicle.make = BMW
##  vehicle.type = suv
##  ->  class 40  [0.500]
##
## Rule 754: (16/14, lift 81.3)
##  fuelType = ELECTRIC
##  location.state = CO
##  ->  class 41  [0.167]
##
## Rule 755: (1, lift 236.4)
##  location.state = ME
##  vehicle.make = Chevrolet
##  ->  class 43  [0.667]
##
## Rule 756: (2/1, lift 487.6)
##  location.state = LA
##  vehicle.make = Audi
##  ->  class 43.5  [0.500]
##
## Rule 757: (1, lift 2600.7)
##  location.state = GA
##  vehicle.make = Honda
##  ->  class 45.6666666666667  [0.667]
##
## Rule 758: (2/1, lift 325.1)
##  location.state = KS
##  vehicle.make = Hyundai
##  ->  class 49  [0.500]
##
## Rule 759: (1, lift 433.4)
##  fuelType = HYBRID
##  location.state = CA
##  vehicle.make = Hyundai
##  ->  class 52  [0.667]
##
## Rule 760: (1, lift 2600.7)
##  location.state = NV
##  vehicle.make = Mercedes-Benz
##  vehicle.type = suv
##  ->  class 54.5  [0.667]
##
```

```
## Rule 761: (1, lift 866.9)
##  location.state = NM
##  vehicle.make = Volkswagen
##  ->  class 61  [0.667]
##
## Rule 762: (1, lift 1300.3)
##  location.state = IL
##  vehicle.make = Alfa Romeo
##  ->  class 63.5  [0.667]
##
## Rule 763: (1, lift 650.2)
##  location.state = MI
##  vehicle.make = Kia
##  ->  class 73  [0.667]
##
## Rule 764: (1, lift 1300.3)
##  location.state = MI
##  vehicle.make = Dodge
##  vehicle.type = truck
##  ->  class 75  [0.667]
##
## Rule 765: (1, lift 1300.3)
##  location.state = CA
##  vehicle.make = Mitsubishi
##  ->  class 75  [0.667]
##
## Rule 766: (1, lift 2600.7)
##  fuelType = HYBRID
##  location.state = PA
##  ->  class 109  [0.667]
##
## Default class: 0
##
##
## Evaluation on training data (3901 cases):
##
##          Rules
##    ----------------
##      No      Errors
##
##     766 2823(72.4%)   <<
##
##
##    Class                Cases   False   False
##                                  Pos     Neg
##    -----                -----   -----   -----
##    0                      279    1227      50
##    0.0181818181818182       1       0       1
##    0.0384615384615385       1       0       1
##    0.0769230769230769       1       0       0
##    0.0833333333333333       7       6       3
##    0.0909090909090909       3       7       0
##    0.1                      2       0       1
##    0.111111111111111        6       4       2
```

```
##    0.125                  9      14       4
##    0.138461538461538      1       0       1
##    0.142857142857143     14      13       7
##    0.166666666666667     26      30      13
##    0.181818181818182      3       4       0
##    0.2                   23      20      16
##    0.222222222222222      2       6       0
##    0.230769230769231      1       0       0
##    0.25                  32      84      19
##    0.272727272727273      3       0       3
##    0.285714285714286      7       3       3
##    0.3                    3       4       0
##    0.3125                 1       0       0
##    0.333333333333333     42      39      21
##    0.363636363636364      2       0       2
##    0.375                  4       2       2
##    0.384615384615385      1       1       0
##    0.4                   15      13       7
##    0.416666666666667      3       3       2
##    0.428571428571429      8       9       3
##    0.444444444444444      5      17       2
##    0.461538461538462      1       0       1
##    0.5                   55      15      39
##    0.545454545454545      1       0       1
##    0.555555555555556      3       0       3
##    0.571428571428571     10      27       4
##    0.583333333333333      1       0       0
##    0.6                   21     107      11
##    0.625                  3       0       1
##    0.636363636363636      2       0       2
##    0.666666666666667     28      12      20
##    0.692307692307692      1       0       1
##    0.7                    3       0       2
##    0.714285714285714      9       9       3
##    0.727272727272727      2       0       2
##    0.75                  21      15      11
##    0.777777777777778      3       1       1
##    0.8                   16      24       7
##    0.818181818181818      3       5       0
##    0.833333333333333     16      24       7
##    0.857142857142857      4       0       4
##    0.866666666666667      1       0       0
##    0.875                  3       3       2
##    0.888888888888889      2       0       2
##    0.9                    2       0       1
##    0.915254237288136      1       0       1
##    0.916666666666667      2       1       1
##    0.928571428571429      1       2       0
##    1                    128      74      87
##    1.09090909090909       1       0       1
##    1.1                    3       0       3
##    1.11111111111111       2       1       1
##    1.125                  3       0       2
##    1.14285714285714       5       1       4
```

```
##    1.16666666666667         4       6       3
##    1.18181818181818         2       0       2
##    1.2                      8       1       6
##    1.22222222222222         7       1       6
##    1.23076923076923         1       0       0
##    1.25                    20       7      14
##    1.27272727272727         1       0       0
##    1.28571428571429         7      18       3
##    1.3                      1       3       0
##    1.33333333333333        29      14      14
##    1.375                    3       3       1
##    1.4                     20       1      13
##    1.41666666666667         4       5       1
##    1.42857142857143         5       6       2
##    1.44444444444444         3       1       2
##    1.45454545454545         2       1       0
##    1.5                     50      48      33
##    1.54545454545455         2       0       2
##    1.55555555555556         5       0       5
##    1.57142857142857         5       5       3
##    1.58333333333333         1       0       1
##    1.6                      9       2       6
##    1.61538461538462         1       0       1
##    1.625                    5      14       2
##    1.63636363636364         3       0       3
##    1.66666666666667        25       5      20
##    1.7                      1       0       1
##    1.71428571428571         7       4       4
##    1.72727272727273         1       0       1
##    1.75                    12       8       7
##    1.76923076923077         2       2       0
##    1.77777777777778         5       0       5
##    1.8                     18       5      14
##    1.83333333333333         5       2       4
##    1.84615384615385         1       0       1
##    1.85714285714286         1       0       1
##    1.875                    5       0       4
##    1.88888888888889         1       0       1
##    1.9                      2       0       2
##    1.90909090909091         3       3       2
##    1.92307692307692         1       1       0
##    2                       99      44      66
##    2.1                      3       4       2
##    2.11111111111111         2       0       2
##    2.125                    4       4       1
##    2.14285714285714         3       2       1
##    2.16666666666667         7       2       5
##    2.18181818181818         1       0       1
##    2.2                     10       5       7
##    2.21428571428571         1       0       0
##    2.22222222222222         5       2       3
##    2.25                    14       6      10
##    2.28571428571429         5       1       2
##    2.3                      1       1       0
```

```
##    2.30769230769231        2       1       0
##    2.33333333333333       24      16      13
##    2.375                   4       0       3
##    2.4                     9       4       6
##    2.42857142857143        3       0       2
##    2.45454545454545        1       0       1
##    2.5                    32      18      22
##    2.53846153846154        2       1       1
##    2.54545454545455        1       0       1
##    2.55555555555556        1       2       0
##    2.57142857142857        4       3       2
##    2.6                     7       3       5
##    2.625                   2       0       2
##    2.66666666666667       19       8      13
##    2.69230769230769        1       0       0
##    2.7                     1       0       0
##    2.71428571428571        3       2       1
##    2.75                   14       2      11
##    2.76923076923077        1       0       1
##    2.8                     6       0       4
##    2.81818181818182        2       0       2
##    2.83333333333333        5       0       4
##    2.85714285714286        1       0       1
##    2.88888888888889        6       3       3
##    2.9                     2       0       2
##    2.90909090909091        1       0       1
##    2.9375                  1       0       1
##    3                      84      75      56
##    3.09090909090909        1       0       1
##    3.1                     1       0       1
##    3.11111111111111        1       0       1
##    3.14285714285714        3       0       1
##    3.15384615384615        3       0       2
##    3.16666666666667        2       0       2
##    3.18181818181818        1       0       1
##    3.2                     2       0       1
##    3.22222222222222        1       0       1
##    3.23076923076923        1       0       1
##    3.25                    8       3       6
##    3.28571428571429        3       0       3
##    3.3                     1       0       1
##    3.30769230769231        1       0       1
##    3.33333333333333       15       8      11
##    3.36363636363636        2       0       0
##    3.375                   1       0       1
##    3.4                    10       5       8
##    3.41666666666667        2       0       2
##    3.42857142857143        3       0       2
##    3.44444444444444        2       0       2
##    3.45454545454545        2       0       2
##    3.5                    25       9      20
##    3.53333333333333        1       0       0
##    3.55555555555556        1       0       1
##    3.57142857142857        6      25       3
```

```
##    3.58333333333333         1       0       1
##    3.6                      8       0       7
##    3.61538461538462         1       0       1
##    3.625                    1       0       1
##    3.63636363636364         2       0       1
##    3.66666666666667        14      11      10
##    3.69230769230769         1       0       1
##    3.71428571428571         3       0       2
##    3.75                     7       2       5
##    3.77777777777778         3       0       3
##    3.8                      9       7       7
##    3.83333333333333         5       0       5
##    3.85714285714286         2       0       1
##    3.875                    2       0       2
##    3.90909090909091         3       0       2
##    3.91666666666667         1       0       1
##    3.92307692307692         1       2       0
##    3.92857142857143         1       0       1
##    4                       67      15      57
##    4.07142857142857         1       0       1
##    4.08333333333333         1       0       0
##    4.125                    3       0       3
##    4.14285714285714         3       3       1
##    4.16666666666667         4       0       3
##    4.18181818181818         1       0       1
##    4.2                      5       3       3
##    4.22222222222222         1       1       0
##    4.23076923076923         1       5       0
##    4.25                    10       0       7
##    4.27272727272727         2       0       1
##    4.28571428571429         7       1       6
##    4.3                      1       0       1
##    4.33333333333333        12       2      10
##    4.36363636363636         1       0       1
##    4.4                      3       0       3
##    4.44444444444444         1       0       1
##    4.5                     23       6      16
##    4.53846153846154         2       0       2
##    4.55555555555556         1       0       1
##    4.57142857142857         3       0       3
##    4.58333333333333         1       0       1
##    4.6                      6      33       3
##    4.61538461538461         1       0       0
##    4.625                    4       0       4
##    4.66666666666667        13       0      12
##    4.7                      4       2       3
##    4.71428571428571         2       1       1
##    4.72727272727273         1       0       1
##    4.75                    11       1      10
##    4.8                      4       2       2
##    4.83333333333333         2       0       2
##    4.85714285714286         1       0       1
##    4.88888888888889         2       0       2
##    4.9                      2       0       2
```

```
##    4.90909090909091        1       0       1
##    4.91666666666667        1       0       1
##    4.92307692307692        1       0       1
##    5                      68      37      50
##    5.08333333333333        1       0       1
##    5.1                     2       0       2
##    5.11111111111111        1       0       1
##    5.125                   2       0       2
##    5.13333333333333        1       0       1
##    5.16666666666667        3       0       3
##    5.2                     8       6       5
##    5.22222222222222        2       0       2
##    5.25                    5       0       5
##    5.28571428571429        6       2       5
##    5.30769230769231        1       0       1
##    5.33333333333333       13       0      12
##    5.35714285714286        1       0       1
##    5.36363636363636        1       0       1
##    5.375                   3       0       2
##    5.4                     3       0       3
##    5.42857142857143        4       0       4
##    5.5                    17       0      16
##    5.53846153846154        1       0       1
##    5.55555555555556        1       0       1
##    5.6                     4       1       3
##    5.625                   3       0       3
##    5.63636363636364        1       0       1
##    5.66666666666667       15       3      13
##    5.7                     1       0       1
##    5.71428571428571        3       0       3
##    5.75                    6       0       5
##    5.77777777777778        1       0       0
##    5.8                     3       1       2
##    5.83333333333333        3       0       3
##    5.85714285714286        1       0       1
##    5.875                   3       0       3
##    5.88888888888889        2       1       1
##    5.90909090909091        1       0       1
##    6                      58       2      51
##    6.1                     1       0       1
##    6.14285714285714        2       1       1
##    6.18181818181818        1       0       0
##    6.2                     3       1       0
##    6.25                    8       3       6
##    6.28571428571429        1       0       0
##    6.3                     1       0       0
##    6.33333333333333        6       0       6
##    6.375                   1       0       1
##    6.38461538461539        2       0       2
##    6.4                     2       0       2
##    6.42857142857143        2       0       2
##    6.44444444444444        1       0       0
##    6.5                    16       1      14
##    6.55555555555556        2       0       2
```

```
##    6.57142857142857        2       0       2
##    6.6                     2       0       2
##    6.625                   2       0       2
##    6.66666666666667        7       1       5
##    6.7                     1       0       0
##    6.71428571428571        2       0       2
##    6.75                    5       0       5
##    6.77777777777778        1       0       0
##    6.8                     6       0       6
##    6.8125                  1       0       1
##    6.83333333333333        2       0       2
##    6.85714285714286        2       0       2
##    6.875                   1       0       1
##    6.9                     2       0       2
##    6.90909090909091        1       1       0
##    6.91666666666667        1       0       1
##    7                      60      27      44
##    7.1                     1       0       0
##    7.125                   1       0       1
##    7.14285714285714        3       0       3
##    7.15384615384615        1       0       1
##    7.16666666666667        1       0       1
##    7.2                     4       0       4
##    7.22222222222222        1       0       1
##    7.23076923076923        1       0       0
##    7.25                    6       4       4
##    7.28571428571429        1       0       1
##    7.33333333333333        9       0       9
##    7.4                     1       0       1
##    7.42857142857143        1       1       0
##    7.44444444444444        2       0       2
##    7.5                    11       8       7
##    7.54545454545455        1       0       1
##    7.57142857142857        1       0       1
##    7.6                     1       0       1
##    7.625                   2       0       2
##    7.66666666666667        9      32       5
##    7.7                     2       0       2
##    7.71428571428571        2       0       1
##    7.75                    6       0       5
##    7.76923076923077        1       0       1
##    7.78571428571429        1       0       1
##    7.8                     7       0       7
##    7.83333333333333        4       0       4
##    7.84615384615385        1       0       1
##    7.85714285714286        1       0       1
##    7.875                   1       0       1
##    7.88888888888889        1       0       1
##    7.9                     1       0       1
##    8                      46      39      38
##    8.1                     1       0       0
##    8.125                   3       0       3
##    8.14285714285714        1       0       1
##    8.15384615384615        1       0       1
```

```
##    8.16666666666667         1       0       1
##    8.2                      4       0       4
##    8.22222222222222         1       0       1
##    8.25                     7       0       7
##    8.27272727272727         1       0       1
##    8.28571428571429         1       0       1
##    8.30769230769231         1       2       0
##    8.33333333333333         5       0       3
##    8.35714285714286         1       0       1
##    8.375                    1       0       1
##    8.4                      1       0       1
##    8.41666666666667         2       0       2
##    8.42857142857143         2       0       2
##    8.44444444444444         1       0       1
##    8.5                     17      37       8
##    8.55555555555556         2       0       2
##    8.57142857142857         2       0       1
##    8.61538461538461         1       0       1
##    8.66666666666667         1       0       1
##    8.69230769230769         1       0       1
##    8.71428571428571         2       0       2
##    8.75                     8       8       6
##    8.8                      2       0       2
##    8.85714285714286         1       0       1
##    8.9                      2       0       2
##    8.90909090909091         1       0       1
##    8.91666666666667         1       0       1
##    9                       41      62      30
##    9.06666666666667         1       0       1
##    9.125                    1       0       1
##    9.16666666666667         2       0       2
##    9.2                      6       1       5
##    9.25                     4       0       4
##    9.27272727272727         1       0       1
##    9.28571428571429         2       0       2
##    9.30769230769231         1       0       1
##    9.33333333333333         6       0       5
##    9.4                      1       0       1
##    9.42857142857143         1       0       1
##    9.5                      9       0       8
##    9.58333333333333         1       0       1
##    9.6                      2       0       2
##    9.66666666666667         5       1       3
##    9.7                      1       0       1
##    9.75                     4       9       2
##    9.8                      3       0       3
##    9.83333333333333         2       0       2
##    9.875                    1       0       1
##    10                      34       9      25
##    10.1                     1       0       1
##    10.1111111111111         2       0       2
##    10.1428571428571         1       0       1
##    10.1666666666667         2       0       1
##    10.2                     4       0       4
```

```
##    10.2222222222222         1       0       1
##    10.25                    4       0       4
##    10.3333333333333         8       0       6
##    10.4                     2       0       2
##    10.4444444444444         1       0       1
##    10.5                    13       3       9
##    10.5555555555556         1       0       1
##    10.5714285714286         2       0       2
##    10.6                     4       0       2
##    10.6666666666667         5       0       5
##    10.7                     1       0       1
##    10.7142857142857         1       2       0
##    10.75                    2       0       2
##    10.7692307692308         1       0       1
##    10.8                     5       0       4
##    10.8181818181818         1       0       0
##    10.8571428571429         2       0       2
##    10.8666666666667         1       0       1
##    10.8888888888889         2       0       2
##    11                      32       5      27
##    11.125                   1       0       1
##    11.1428571428571         1       0       0
##    11.2                     1       0       1
##    11.2222222222222         1       0       1
##    11.25                    6       0       6
##    11.2857142857143         1       0       1
##    11.3333333333333         7       0       6
##    11.4                     3       0       3
##    11.4285714285714         1       0       0
##    11.4666666666667         1       0       1
##    11.5                     7       3       5
##    11.6                     2       0       2
##    11.6666666666667         5       0       5
##    11.7142857142857         2       0       1
##    11.75                    8       0       8
##    11.7777777777778         2       1       1
##    11.8                     4       0       2
##    11.8333333333333         1       0       1
##    12                      30       2      25
##    12.1                     1       0       1
##    12.1111111111111         1       0       1
##    12.125                   1       0       1
##    12.1428571428571         1       1       0
##    12.1666666666667         3       0       3
##    12.2                     2       0       2
##    12.25                    2       1       1
##    12.2857142857143         1       0       1
##    12.3333333333333         2       0       2
##    12.3571428571429         1       0       1
##    12.4                     2       0       2
##    12.5                     8       1       7
##    12.6                     5       1       3
##    12.6666666666667         4       0       4
##    12.7142857142857         1       0       1
```

```
##    12.75                2       0       2
##    12.8                 1       0       0
##    13                  29       2      25
##    13.0833333333333     1       0       1
##    13.1666666666667     2       0       2
##    13.2                 2       0       2
##    13.2222222222222     1       0       1
##    13.25                4       2       3
##    13.2857142857143     1       0       1
##    13.3333333333333     4       0       4
##    13.375               2       0       1
##    13.5                 9       0       8
##    13.5555555555556     2       0       2
##    13.5714285714286     1       0       1
##    13.625               1       0       1
##    13.6666666666667     3       0       3
##    13.75                4       0       4
##    13.8181818181818     1       0       1
##    13.8888888888889     1       0       1
##    14                  35      11      26
##    14.125               1       0       1
##    14.2                 2       0       1
##    14.25                3       0       3
##    14.3333333333333     5       0       5
##    14.3571428571429     1       0       1
##    14.375               1       0       1
##    14.5                 7       0       6
##    14.5555555555556     1       0       1
##    14.6                 2       0       2
##    14.6666666666667     4       0       4
##    14.75                2       0       2
##    14.8                 2       0       2
##    14.8333333333333     1       0       1
##    14.8571428571429     1       0       0
##    14.9                 2       0       2
##    15                  17       0      16
##    15.1666666666667     3       0       3
##    15.2                 1       0       1
##    15.25                2      10       0
##    15.2857142857143     1       0       1
##    15.3333333333333     3       0       3
##    15.4                 2       0       2
##    15.5                 4       0       4
##    15.5454545454545     1       0       1
##    15.6666666666667     3       0       2
##    15.75                1       0       1
##    15.8                 3       1       2
##    15.8333333333333     1       0       1
##    16                  20      18      12
##    16.2                 1       0       1
##    16.3333333333333     3       0       3
##    16.3636363636364     1       0       1
##    16.5                11       0      10
##    16.5714285714286     1       0       1
```

```
##    16.6                  1       0       0
##    16.6666666666667      2       0       2
##    16.75                 3       0       3
##    16.8333333333333      1       0       1
##    17                   25       2      20
##    17.2                  2       0       2
##    17.25                 1       0       1
##    17.3333333333333      4       0       3
##    17.4                  1       0       1
##    17.4444444444444      1       0       1
##    17.5                  7       0       6
##    17.6666666666667      2       0       2
##    17.75                 4       0       3
##    17.8                  2       0       2
##    18                   20       1      19
##    18.2                  2       0       2
##    18.25                 1       0       1
##    18.3333333333333      3       0       3
##    18.375                1       0       1
##    18.4285714285714      1       0       1
##    18.5                  7       0       7
##    18.6                  1       0       1
##    18.6666666666667      6       0       6
##    19                   23       6      21
##    19.1111111111111      1       0       1
##    19.2                  2       0       2
##    19.25                 2       0       1
##    19.2857142857143      1       1       0
##    19.3333333333333      2       0       2
##    19.5                 12       0      11
##    19.6                  1       0       1
##    19.6666666666667      4       0       3
##    19.8                  1       0       1
##    20                   29       1      26
##    20.25                 1       0       1
##    20.3333333333333      1       0       1
##    20.5                  7       1       6
##    20.6                  2       0       1
##    20.7142857142857      1       0       1
##    20.75                 2       0       2
##    20.8333333333333      2       0       2
##    21                   23       2      22
##    21.1111111111111      1       0       1
##    21.3333333333333      3       0       2
##    21.4                  2       0       2
##    21.4285714285714      1       0       1
##    21.5                  3       0       2
##    21.6                  1       0       1
##    21.6666666666667      4       0       4
##    21.75                 2       0       2
##    21.8                  3       0       2
##    22                   19       0      16
##    22.2                  1       0       1
##    22.3333333333333      1       0       1
```

```
##    22.5                  5       0       4
##    22.6                  1       0       1
##    22.75                 1       0       1
##    22.8333333333333      1       0       0
##    22.8571428571429      1       0       1
##    23                    7       0       6
##    23.2857142857143      1       0       1
##    23.5                  3       0       3
##    23.6666666666667      1       0       1
##    23.75                 1       0       1
##    23.8571428571429      1       0       1
##    24                   14       0      13
##    24.25                 1       0       1
##    24.3333333333333      1       0       1
##    24.5                  6       2       4
##    24.6666666666667      2       0       2
##    24.8571428571429      1       0       0
##    25                   13       0      13
##    25.2                  2       0       2
##    25.25                 1       0       1
##    25.3333333333333      3       0       3
##    25.375                1       0       0
##    25.5                  5       0       5
##    25.6                  1       0       1
##    25.8333333333333      1       0       1
##    26                    8       0       7
##    26.2                  1       0       1
##    26.3333333333333      2       0       2
##    26.4285714285714      1       0       1
##    26.5                  5       0       5
##    26.6                  1       0       1
##    26.6666666666667      5       0       4
##    26.75                 1       0       0
##    27                   10       0       9
##    27.25                 1       0       1
##    27.5                  1       0       1
##    27.6                  1       0       1
##    27.6666666666667      1       0       1
##    27.75                 1       0       1
##    28                    5       0       5
##    28.2                  1       0       0
##    28.2142857142857      1       0       0
##    28.5                  3       0       3
##    28.6666666666667      2       0       2
##    28.8                  1       0       1
##    29                   15       1      13
##    29.1428571428571      1       0       1
##    29.25                 1       0       1
##    29.3333333333333      1       0       1
##    29.5                  4       0       4
##    29.75                 2       0       2
##    30                   12       0      11
##    30.3333333333333      2       1       1
##    30.5                  6       0       6
```

```
##    30.6666666666667         5       0       5
##    30.7142857142857         1       0       0
##    30.75                    1       0       1
##    31                       7       0       5
##    31.1666666666667         1       0       1
##    31.25                    1       0       1
##    31.3333333333333         1       0       1
##    31.5                     1       0       1
##    31.6666666666667         2       0       2
##    32                      14      63      11
##    32.25                    1       0       1
##    32.4285714285714         1       0       1
##    32.6                     1       0       0
##    32.6666666666667         1       1       0
##    33                      11       0       8
##    33.5                     5       0       5
##    33.6                     1       0       1
##    33.6666666666667         1       0       1
##    34                      12       9       9
##    34.5                     2       0       2
##    35                      12       0      12
##    35.2                     1       0       1
##    35.5                     2       0       2
##    35.6666666666667         3       1       2
##    36                      14       0      14
##    36.3333333333333         1       0       1
##    36.75                    1       0       1
##    37                       6       0       6
##    37.125                   1       0       1
##    37.3333333333333         1       0       1
##    37.5                     4       0       4
##    37.6666666666667         1       0       1
##    37.75                    1       0       1
##    38                      14       0      11
##    38.25                    1       0       1
##    38.2857142857143         1       0       1
##    38.3333333333333         1       0       1
##    38.5                     2       0       1
##    38.6666666666667         1       0       1
##    39                       6       0       6
##    39.3333333333333         1       0       0
##    39.6                     1       0       1
##    39.7142857142857         1       0       1
##    40                      13       1      11
##    40.25                    1       0       1
##    40.3333333333333         1       0       1
##    40.5                     1       0       1
##    41                       8      14       6
##    41.1111111111111         1       0       1
##    41.3333333333333         1       0       1
##    41.5                     4       0       4
##    41.8333333333333         1       0       1
##    42                       9       0       9
##    42.25                    1       0       1
```

```
##    42.3333333333333         1       0       1
##    42.5                     1       0       1
##    43                      11       0      10
##    43.25                    2       0       2
##    43.3333333333333         1       0       1
##    43.4                     1       0       1
##    43.5                     4       1       3
##    43.6                     1       0       1
##    44                       3       0       3
##    44.5                     2       0       2
##    44.6666666666667         1       0       1
##    45                      12       0      12
##    45.5                     4       0       4
##    45.6666666666667         1       0       0
##    46                       5       0       5
##    46.5                     1       0       1
##    46.6666666666667         1       0       1
##    46.8                     1       0       1
##    47                       1       0       1
##    47.25                    1       0       1
##    47.3333333333333         1       0       1
##    47.5                     2       0       2
##    48                       4       0       4
##    48.5                     1       0       1
##    49                       6       1       5
##    49.5                     1       0       1
##    50                       4       0       4
##    50.5                     1       0       1
##    51                       2       0       2
##    51.5                     1       0       1
##    51.6666666666667         1       0       1
##    52                       6       0       5
##    53                       4       0       4
##    53.3333333333333         1       0       1
##    53.5                     1       0       1
##    53.75                    1       0       1
##    54                       4       0       4
##    54.2                     1       0       1
##    54.5                     1       0       0
##    55                       3       0       3
##    55.5                     1       0       1
##    56                       3       0       3
##    57                       3       0       3
##    58                       2       0       2
##    58.5                     1       0       1
##    59                       4       0       4
##    59.5                     1       0       1
##    59.6666666666667         1       0       1
##    60                       4       0       4
##    61                       3       0       2
##    62                       3       0       3
##    63                       2       0       2
##    63.5                     2       0       1
##    64                       3       0       3
```

```
##    64.5                    1       0       1
##    65.5                    1       0       1
##    66                      1       0       1
##    66.3333333333333        1       0       1
##    67                      4       0       4
##    68                      1       0       1
##    69                      1       0       1
##    69.3333333333333        1       0       1
##    69.5                    1       0       1
##    69.6666666666667        1       0       1
##    70                      2       0       2
##    71                      1       0       1
##    71.5                    1       0       1
##    72                      2       0       2
##    73                      4       0       3
##    73.5                    2       0       2
##    73.75                   1       0       1
##    74                      1       0       1
##    74.5                    1       0       1
##    74.6666666666667        1       0       1
##    75                      2       0       0
##    75.5                    1       0       1
##    76                      1       0       1
##    76.6666666666667        1       0       1
##    77                      1       0       1
##    78                      2       0       2
##    79.5                    1       0       1
##    81.3333333333333        1       0       1
##    82                      1       0       1
##    86.5                    1       0       1
##    87                      1       0       1
##    92                      1       0       1
##    96                      1       0       1
##    96.5                    1       0       1
##    97                      1       0       1
##    98.6666666666667        1       0       1
##    99                      2       0       2
##    104                     1       0       1
##    105                     1       0       1
##    106                     1       0       1
##    108                     2       0       2
##    109                     1       0       0
##    113                     2       0       2
##    115.5                   1       0       1
##    122                     1       0       1
##    123                     1       0       1
##    137                     1       0       1
##    140                     1       0       1
##    151                     1       0       1
##
##
##  Attribute usage:
##
##   71.60% vehicle.make
```

```
##   70.90% location.state
##   43.35% vehicle.type
##    7.64% fuelType
##
##
## Time: 0.6 secs
```

El model resulta en un error bastant gran: un 72,4 % de les mostres es considera mal catalogades i amb més de 700 regles, no és pràctic.

Com a següent pas decidim fer una discretització de la variable **frequency** per repetir el modelat a continuació. Discretitzem la variable en base als seus 4 quaartils.

```
summary(ds[,"frequency"])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00    1.40    5.00   10.92   13.71  151.00
```

```
ds["frequency_segments"] <- cut(ds$frequency, breaks = c(0,1.4,10.92,13.71,151), labels = c("low", "medi
```

Repetim el modelat

```
set.seed(1)
data_random <- ds[sample(nrow(ds)),]

# Agafem com a **y** la vairable **frequency_segments**, i les categoriques com a **X**.
set.seed(666)
y <- data_random["frequency_segments"]
X <- data_random[c("fuelType","location.state","vehicle.make","vehicle.type")]

# Triem un subconjunt per a l'entrenament del model i un subconjunt per a provar-lo
# El subconjunt d'entenament tindra 2/3 de les observacions: per tant 3901
# El subconjunt de prova tindra 1/3 de les observacion: per tant 1950
trainX <- X[1:3901,]
trainy <- y[1:3901,]
testX  <- X[3902:5851,]
testy  <- y[3902:5851,]

# Creem el model, en visibilitzem la qualitat i n'extraiem les regles

trainy = as.factor(trainy)
model <- C50::C5.0(trainX, trainy, rules=TRUE )
#summary(model)
```

Trobem que s'ha creat un model que simplement simplifica que la classe típica de **frequency_segment** és medium-low i deixa un 51,6 % d'error, que s'explica pels casos que no pertanyen a la classe medium-low.

Fi A REVISAR

Generar un model que pugui preveure income

Conclusions

Bibliografia