

# STAC67 Final Project Report

Nilson Gao, Vedat Goktepe, Rebecca Han, Ben Wang

2024-12-03

[TO DO: make this in title page as required by rubric]

## Research Context

The objective of this model is to investigate the intrinsic factors that affect Price of Cars in Serbia based off detailed car listings provided by an online marketplace where [TO DO - yap about car price significance, what question we want to answer, how this can be beneficial knowledge for consumers/dealerships/car companies/manufacturers/etc.]

## Exploratory Data Analysis

```
# read data file, published in 2024 on https://www.kaggle.com/datasets/mmakarovlab-serbia-car-sales-price
car_price_data <- read.csv("serbia_car_sales_price_2024.csv")
```

Before we begin investigating, we notice that there are some issues with the data. Some rows are missing values under certain variables (i.e. #2, #233, #1705, etc.), and some variables are hard to work with. Knowing a car's **year** might be less informative than knowing its age, so we made a new column containing values for 2024 – Year called **age**. A car's **horsepower** is significant, but it's hard to use that data when it's given as two values in the format *HP (kW)*, so we keep only the HP metric. Additionally, some variable names are hard to work with because of length or how it might interfere with R code, such as **car\_mileage**, **km**, so we made those easier to process as well. As for the missing values, when we analyze the significance of a variable, we'll make sure to exclude rows where values for that variable are empty.

```
clean_data <- car_price_data
clean_data[clean_data == ""] <- NA
clean_data <- na.omit(clean_data)

clean_data$age <- 2024 - clean_data$year
clean_data$horsepower<-gsub(pattern = "^(\\d+) HP.*", replacement = "\\\\1", clean_data$horsepower)
clean_data$horsepower<-as.numeric(clean_data$horsepower)
```

```
# making the variable names easier to process
names(clean_data) <- gsub(pattern = "\\\\.*", replacement = "", names(clean_data))
```

```
#will keep this here
summary(clean_data)
```

```
##      views        favorite       post_info        price
##  Min.   :  0.0   Min.   : 0.000   Length:7076   Min.   : 100
##  1st Qu.: 59.0   1st Qu.: 0.000   Class :character 1st Qu.: 1750
##  Median :109.0   Median : 1.000   Mode   :character Median : 3500
##  Mean   :301.7   Mean   : 2.572                    Mean   : 5030
```

```

## 3rd Qu.: 239.0 3rd Qu.: 3.000 3rd Qu.: 6100
## Max. :27770.0 Max. :151.000 Max. :71500
## car_name year A_C emission_class
## Length:7076 Min. :1970 Length:7076 Length:7076
## Class :character 1st Qu.:2003 Class :character Class :character
## Mode :character Median :2007 Mode :character Mode :character
## Mean :2007
## 3rd Qu.:2010
## Max. :2024
## seats_amount horsepower color car_mileage
## Min. :2.000 Min. : 1.0 Length:7076 Min. :1.000e+00
## 1st Qu.:5.000 1st Qu.: 86.0 Class :character 1st Qu.:1.780e+05
## Median :5.000 Median :109.0 Mode :character Median :2.200e+05
## Mean :4.949 Mean :116.1 Mean :2.118e+06
## 3rd Qu.:5.000 3rd Qu.:140.0 3rd Qu.:2.700e+05
## Max. :9.000 Max. :900.0 Max. :4.295e+09
## engine_capacity type_of_drive doors fuel
## Min. : 100 Length:7076 Length:7076 Length:7076
## 1st Qu.: 1400 Class :character Class :character Class :character
## Median : 1700 Mode :character Mode :character Mode :character
## Mean : 1727
## 3rd Qu.: 1995
## Max. :10000
## car_type gearbox age
## Length:7076 Length:7076 Min. : 0.0
## Class :character Class :character 1st Qu.:14.0
## Mode :character Mode :character Median :17.0
## Mean :17.5
## 3rd Qu.:21.0
## Max. :54.0

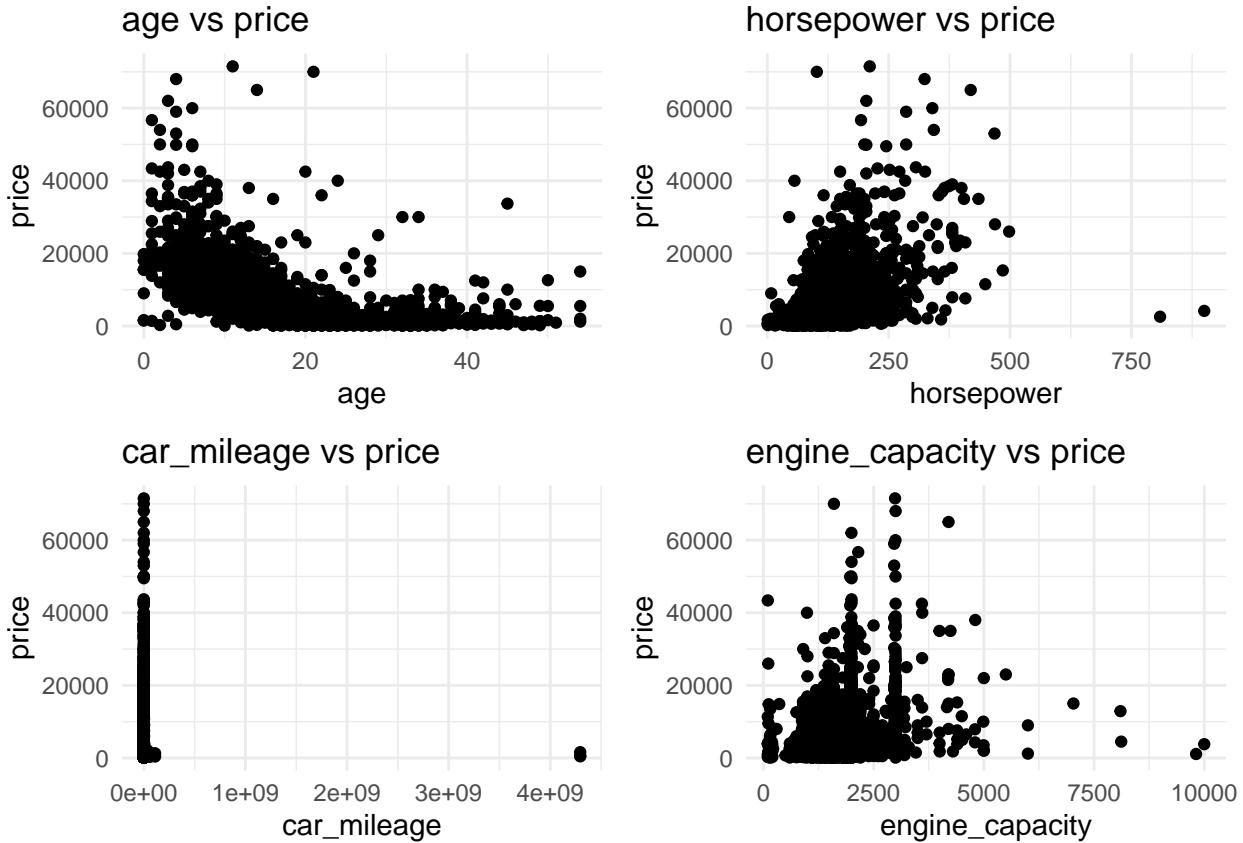
```

Now, we want to check on which variables are good predictors. For the continuous variables, we first plot scatter graphs for each variable against car price:

```

p1 <- ggplot(clean_data, aes(x = age, y = price)) + geom_point() + theme_minimal() + ggtitle("age vs pr")
p2 <- ggplot(clean_data, aes(x = horsepower, y = price)) + geom_point() + theme_minimal() + ggtitle("ho")
p3 <- ggplot(clean_data, aes(x = car_mileage, y = price)) + geom_point() + theme_minimal() + ggtitle("ca")
p4 <- ggplot(clean_data, aes(x = engine_capacity, y = price)) + geom_point() + theme_minimal() + ggtitle("en")
grid.arrange(p1,p2,p3,p4,ncol=2)

```



We notice that there are some influential points (and possibly leverage points) on the car\_mileage predictor. Let's get rid of those using hat values and semistudentized residuals to detect which ones are too high:

THIS MAY NEED TRIMMING BUT IDK - Nilson

```
outlier_clean <- function(data, model) {
  #removal of points in x (leverage points)
  leverage <- hatvalues(model)
  threshold <- 2 * length(coef(model)) / nrow(data)
  leverage_points <- which(leverage > threshold | leverage > 0.5)
  clean <- data[-leverage_points,]

  n <- nrow(data)
  threshold <- qt(1 - (0.05/(2*n)), n-2-1)
  semistudentized_residuals <- rstandard(model)
  outlier_points <- which(abs(semistudentized_residuals) > threshold)
  clean <- clean[-outlier_points,]

  cooks_d <- cooks.distance(model)
  cooks_threshold <- 4/n #might need to change
  influential_points <- which(cooks_d > cooks_threshold)
  clean <- clean[-influential_points,]
  return(clean)
}

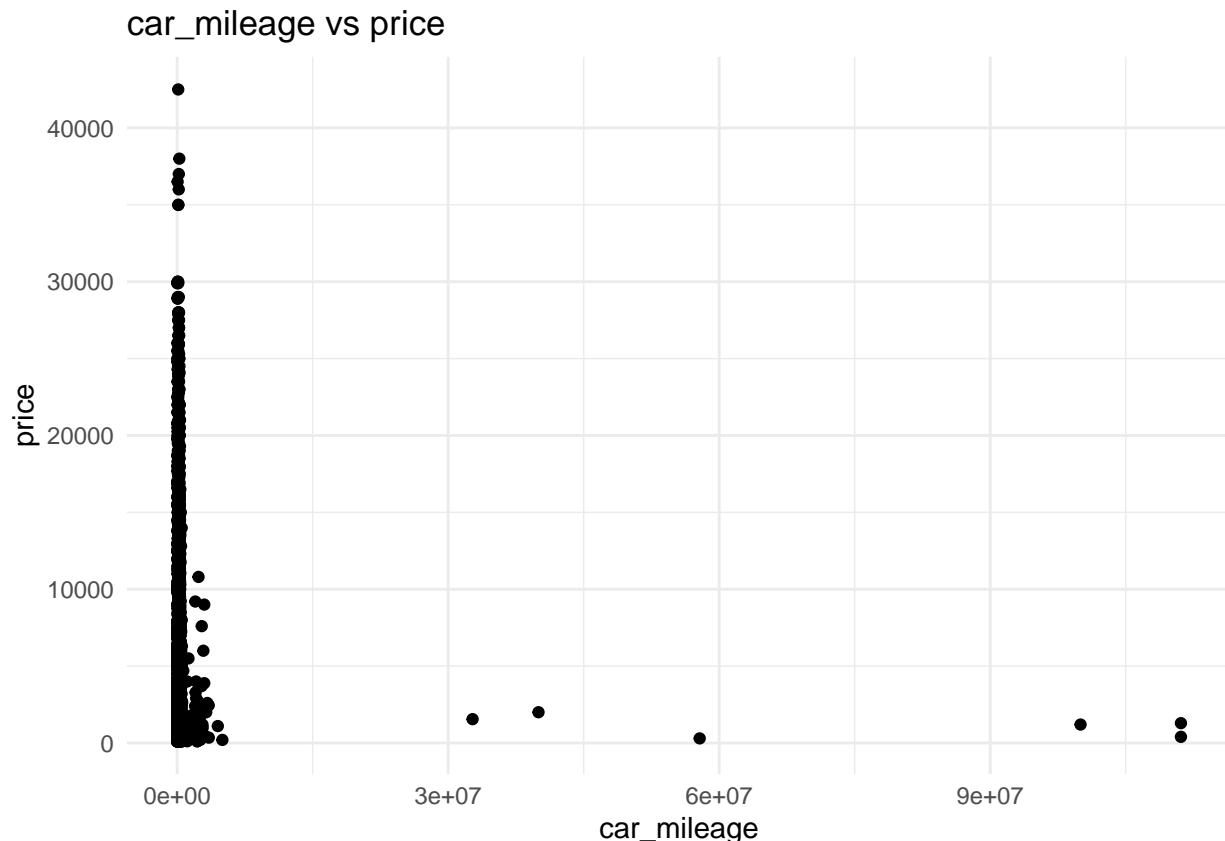
cm_model <- lm(price ~ car_mileage, data = clean_data)
clean_cm <- outlier_clean(clean_data, cm_model)
```

```
hp_model <- lm(price ~ horsepower, data = clean_data)
clean_hp <- outlier_clean(clean_data, hp_model)
```

```
age_model <- lm(price ~ age, data = clean_data)
clean_age <- outlier_clean(clean_data, age_model)
```

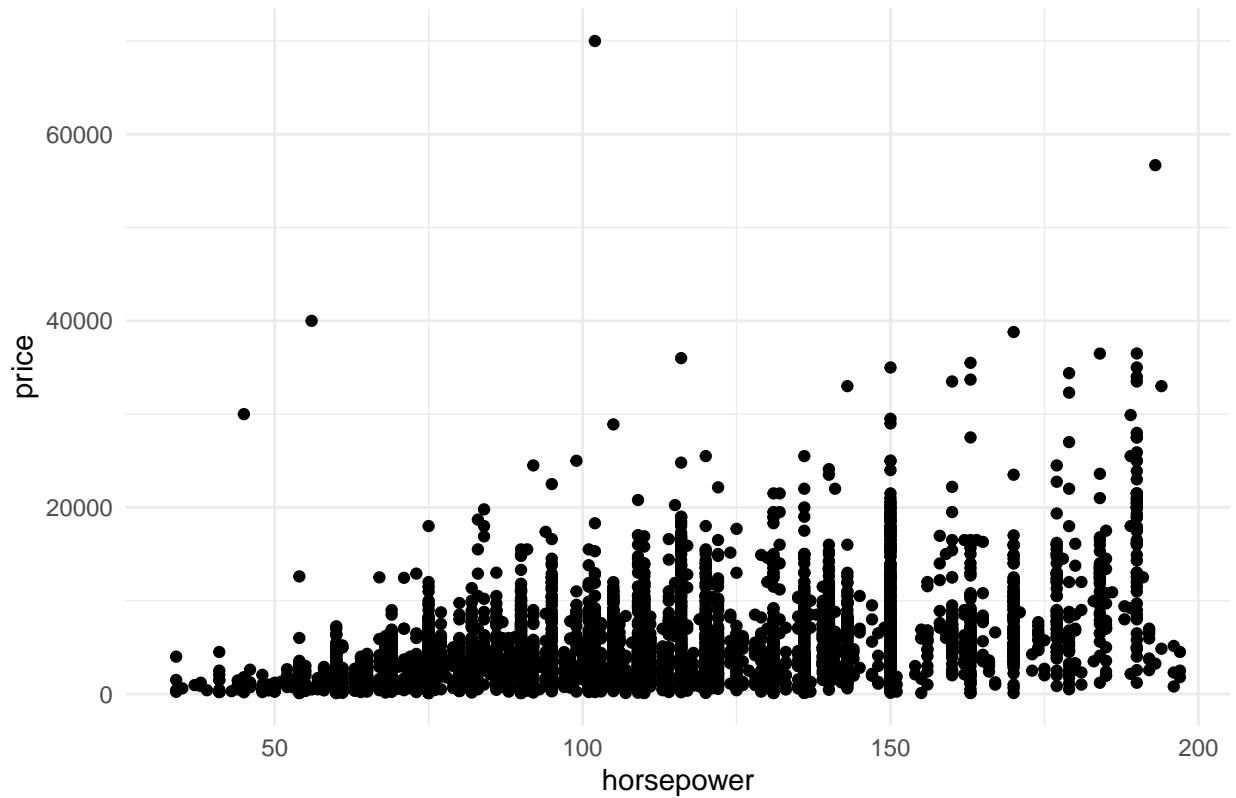
```
ec_model <- lm(price ~ engine_capacity, data = clean_data)
clean_ec <- outlier_clean(clean_data, ec_model)
```

```
ggplot(clean_cm, aes(x = car_mileage, y = price)) + geom_point() + theme_minimal() + ggtitle("car_mileage vs price")
```



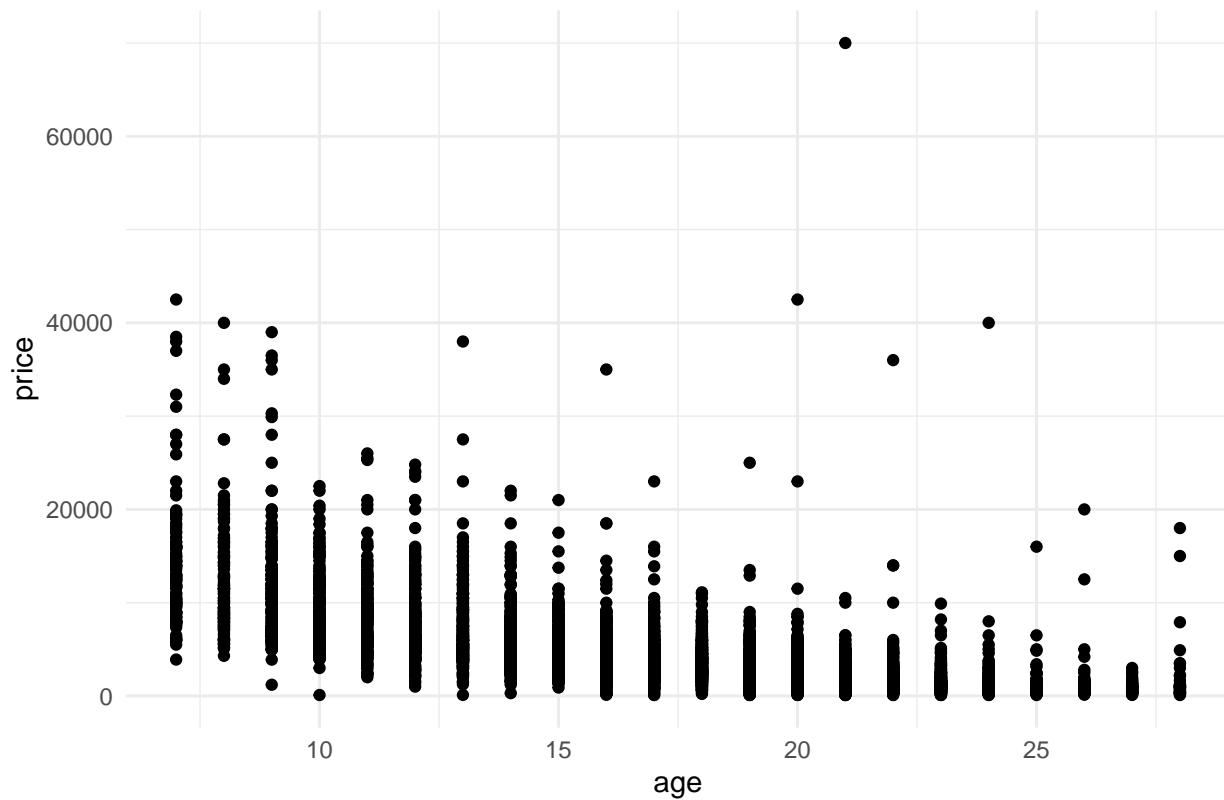
```
ggplot(clean_hp, aes(x = horsepower, y = price)) + geom_point() + theme_minimal() + ggtitle("horsepower vs price")
```

horsepower vs price



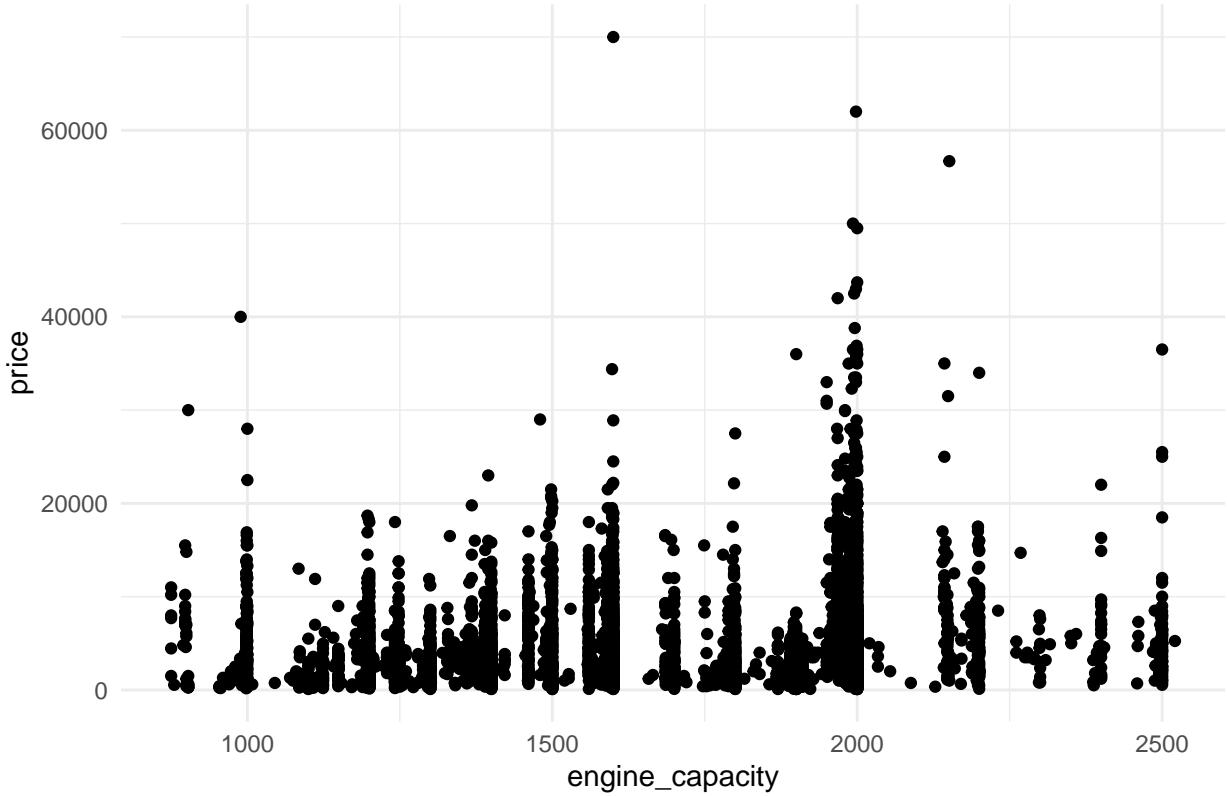
```
ggplot(clean_age, aes(x = age, y = price)) + geom_point() + theme_minimal() + ggtitle("age vs price")
```

age vs price



```
ggplot(clean_ec, aes(x = engine_capacity, y = price)) + geom_point() + theme_minimal() + ggttitle("engine
```

engine\_capacity vs price



```
#cleaning for car mileage
model <- lm(price ~ car_mileage, data = clean_data)

# leverage points (outliers in x) removal
leverage <- hatvalues(model)
threshold <- 2 * length(coef(model)) / nrow(clean_data)
leverage_points <- which(leverage > threshold | leverage > 0.5)
clean_cm_data <- clean_data[-leverage_points, ] # clean_cm_data filters out for car_mileage ONLY. This is important for OLS

clean_cm_data <- subset(clean_cm_data, car_mileage <= 750000) # trimming x-axis data, will cause bias. This is important for OLS

# outliers in y removal
n <- nrow(clean_data)
print(n)

## [1] 7076

threshold <- qt(1 - (0.05/(2*n)), n-2-1) # considering p'=2 (intercept and car_mileage), could be wrong
print(threshold)

## [1] 4.495051

studentized_residuals <- rstudent(model)
outlier_indices <- which(abs(studentized_residuals) > threshold)
clean_cm_data <- clean_cm_data[-outlier_indices, ]

#cleaning for horsepower
model <- lm(price ~ horsepower, data = clean_data)
```

```

# leverage points (outliers in x) removal
leverage <- hatvalues(model)
threshold <- 2 * length(coef(model)) / nrow(clean_data)
leverage_points <- which(leverage > threshold | leverage > 0.5)
clean_hp_data <- clean_data[-leverage_points, ]

#clean_hp_data <- subset(clean_data, price <= 60000) # trimming y-axis data, will cause bias.

# outliers in y removal
n <- nrow(clean_data)
print(n)

## [1] 7076

threshold <- qt(1 - (0.05/(2*n)), n-2-1) # considering p'=2 (intercept and horsepower), could be wrong
print(threshold)

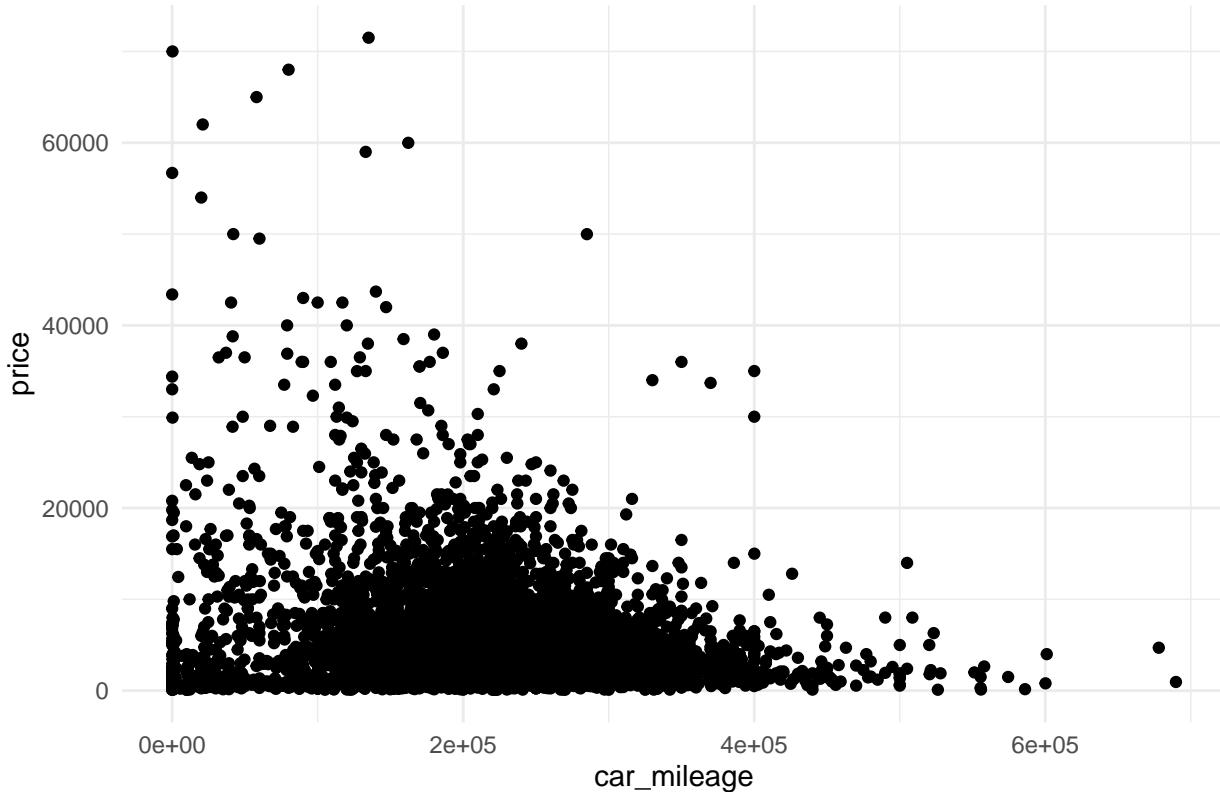
## [1] 4.495051

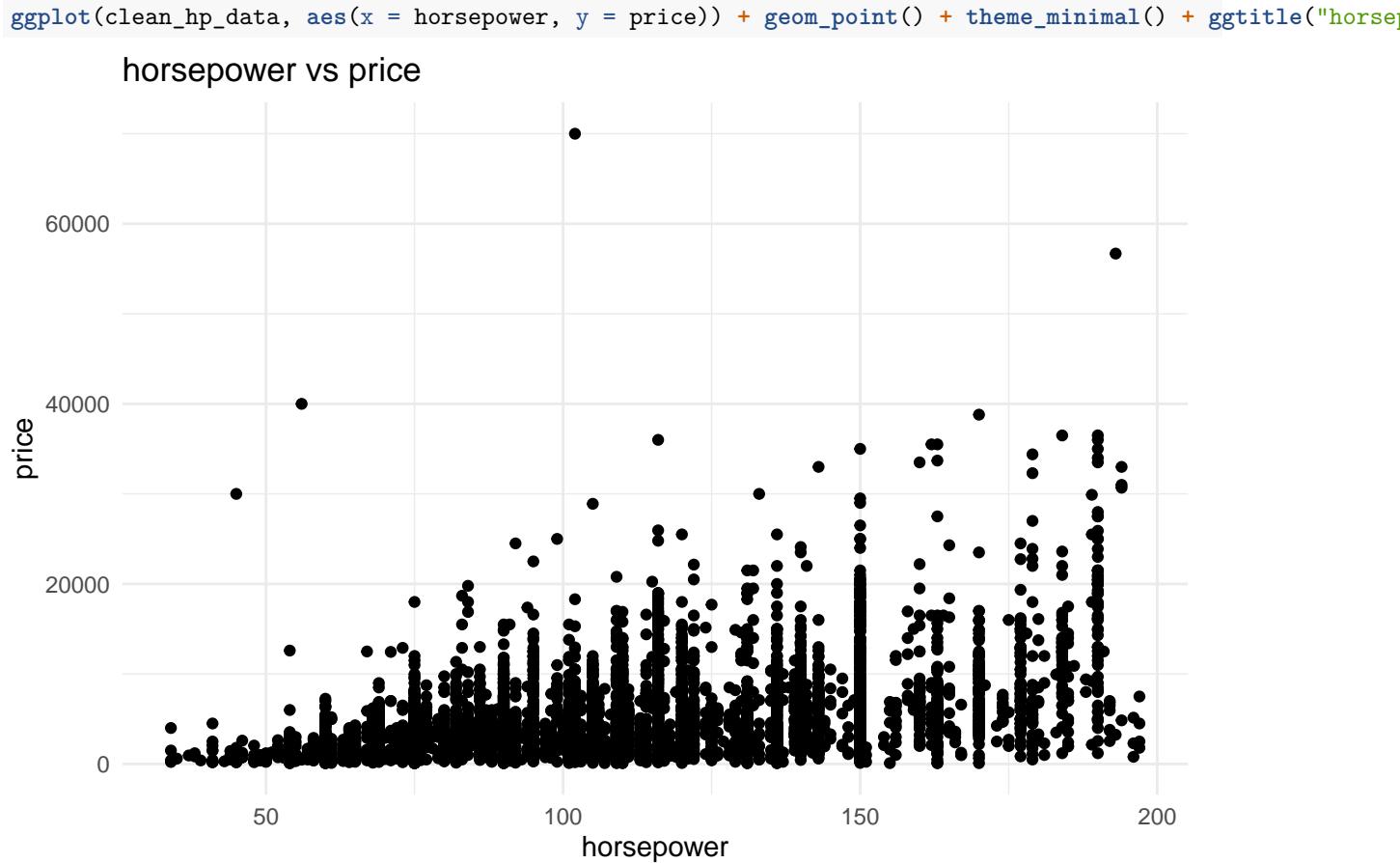
studentized_residuals <- rstudent(model)
outlier_indices <- which(abs(studentized_residuals) > threshold)
clean_hp_data <- clean_hp_data[-outlier_indices, ]

# TO DO: (?) should we repeat this process for the other continuous variables?

ggplot(clean_cm_data, aes(x = car_mileage, y = price)) + geom_point() + theme_minimal() + ggtitle("car_mileage vs price")

```

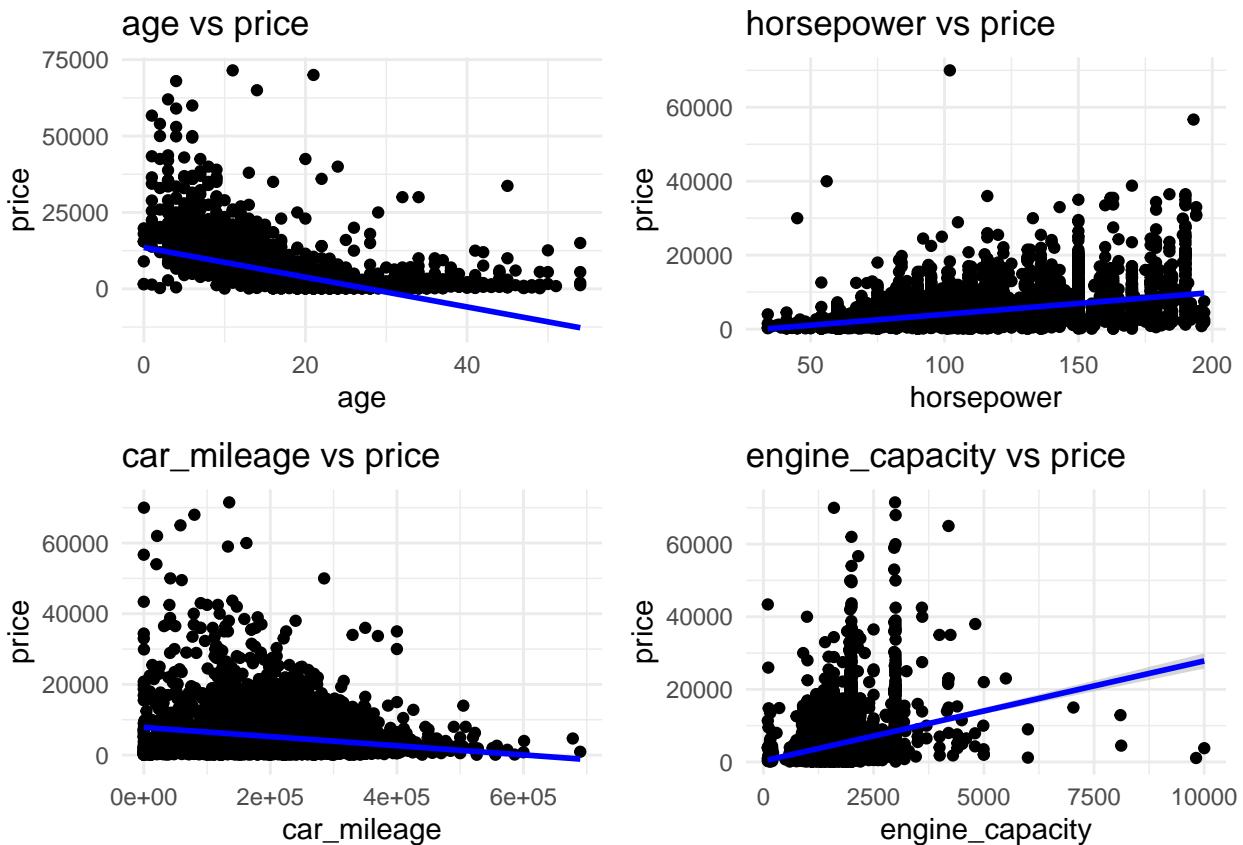




After outlier cleaning, let's check on the scatter plots again to see if these continuous variables have a significant influence on the car price:

```
p1 <- ggplot(clean_data, aes(x = age, y = price)) + geom_point() + geom_smooth(method = "lm", se = TRUE)
p2 <- ggplot(clean_hp_data, aes(x = horsepower, y = price)) + geom_point() + geom_smooth(method = "lm", se = TRUE)
p3 <- ggplot(clean_cm_data, aes(x = car_mileage, y = price)) + geom_point() + geom_smooth(method = "lm", se = TRUE)
p4 <- ggplot(clean_data, aes(x = engine_capacity, y = price)) + geom_point() + geom_smooth(method = "lm", se = TRUE)
grid.arrange(p1,p2,p3,p4,ncol=2)

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



[note: minor concern... if you look at the p-value and t-value for car\_mileage, you'll notice that we fail to reject H\_0: beta\_1=0 for it...? does this mean we should exclude it...?]

```
model <- lm(price ~ age, data = clean_data)
summary(model)

## 
## Call:
## lm(formula = price ~ age, data = clean_data)
## 
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -12321    -2260   -1167     664   66673 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 13544.388   156.228   86.70 <2e-16 ***
## age         -486.547    8.352  -58.26 <2e-16 ***
## ---        
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 4644 on 7074 degrees of freedom
## Multiple R-squared:  0.3242, Adjusted R-squared:  0.3241 
## F-statistic: 3394 on 1 and 7074 DF,  p-value: < 2.2e-16

model <- lm(price ~ horsepower, data = clean_data)
summary(model)
```

```

## 
## Call:
## lm(formula = price ~ horsepower, data = clean_data)
## 
## Residuals:
##    Min     1Q Median     3Q    Max 
## -48735 -2565   -786   1260  65834 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) -2060.783   152.426 -13.52   <2e-16 ***
## horsepower    61.051     1.215   50.25   <2e-16 ***
## --- 
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 
## 
## Residual standard error: 4849 on 7074 degrees of freedom
## Multiple R-squared:  0.2631, Adjusted R-squared:  0.263 
## F-statistic:  2525 on 1 and 7074 DF,  p-value: < 2.2e-16 

model <- lm(price ~ car_mileage, data = clean_data)
summary(model)

## 
## Call:
## lm(formula = price ~ car_mileage, data = clean_data)
## 
## Residuals:
##    Min     1Q Median     3Q    Max 
## -4932 -3282 -1532   1068  66468 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 5.032e+03  6.716e+01  74.927   <2e-16 *** 
## car_mileage -1.014e-06  7.592e-07 -1.336    0.182  
## --- 
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 
## 
## Residual standard error: 5648 on 7074 degrees of freedom
## Multiple R-squared:  0.0002523, Adjusted R-squared:  0.000111 
## F-statistic: 1.785 on 1 and 7074 DF,  p-value: 0.1815 

model <- lm(price ~ engine_capacity, data = clean_data)
summary(model)

## 
## Call:
## lm(formula = price ~ engine_capacity, data = clean_data)
## 
## Residuals:
##    Min     1Q Median     3Q    Max 
## -26213 -3036 -1428   1230  65319 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 272.7652   232.8749   1.171    0.242 

```

```

## engine_capacity    2.7550      0.1295  21.278   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5476 on 7074 degrees of freedom
## Multiple R-squared:  0.06015,    Adjusted R-squared:  0.06002
## F-statistic: 452.8 on 1 and 7074 DF,  p-value: < 2.2e-16

```

Now moving on to the categorical variables.

[TO DO: There's some weird charts that the exemplar from last yr uses. If anyone can figure out how to make and interpret those that'd be great.]

We need to make sure that there's no correlation between the different categorical variables.

```

# Select numeric predictors only
numeric_data <- clean_data[sapply(clean_data, is.numeric)]

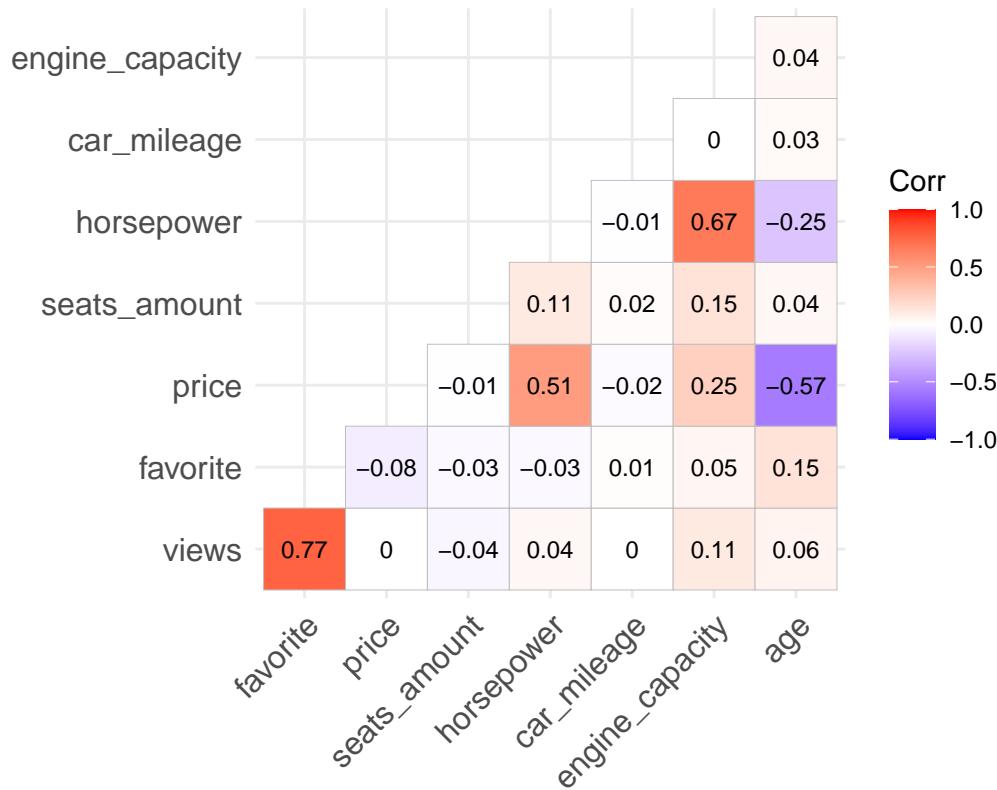
# Remove specific variables like 'Year'
numeric_data <- numeric_data[, !names(numeric_data) %in% c("year")]

# Compute correlation matrix
cor_matrix <- cor(numeric_data, use = "complete.obs")

# Plot correlation matrix with ggcorrplot
ggcorrplot(
  cor_matrix,
  method = "square",           # Style of the plot
  type = "lower",              # Show lower triangular correlation matrix
  lab = TRUE,                  # Display correlation values inside the cells
  lab_size = 3,                # Adjust size of labels
  title = "Correlation Heatmap", # Add a title
  colors = c("blue", "white", "red") # Define color gradient (negative, neutral, positive correlations)
)

```

### Correlation Heatmap



[Note: based off of this correlation map: - Remove one of views/favourites since this has strong correlation (but I think we're getting rid of both anyways) - Ensure no multicollinearity between horsepower and engine\_capacity (VIF?) - Check to see if we can remove seats\_amount and car\_mileage from the model since they don't give much information on price it looks like - this is just a little suspicious since usually more mileage should mean cheaper car? just make sure to do lots of investigation before removing it.]

Before we delve further into data analysis, we notice that there's some information in our dataset that is unlikely to be relevant, such as how many views or favourites the car posting gets, or the date of which it was posted. However, we need to run a t-test to make sure that those variables indeed do not have any influence on the final price of the car.

[TO DO: - get rid of certain variables like Views etc., justify using math/stats (can't just say "pretty sure it won't affect anything") - pretty sure it's just a basic  $\beta_i = 0$  t-test? correct me if I'm wrong]

### Categorical Variable Analysis

```
# List of all categorical variables
categorical_vars <- c("post_info", "A_C", "emission_class", "seats_amount",
                      "color", "type_of_drive", "doors", "fuel",
                      "car_type", "gearbox")

# Create individual plots for each categorical variable
for (var in categorical_vars) {

  # Filter non-empty data for the current variable
  non_empty_data <- clean_data[!is.na(clean_data[[var]]) & !is.na(clean_data$price), ]

  # Check if the variable is a factor; if not, convert it
}
```

```

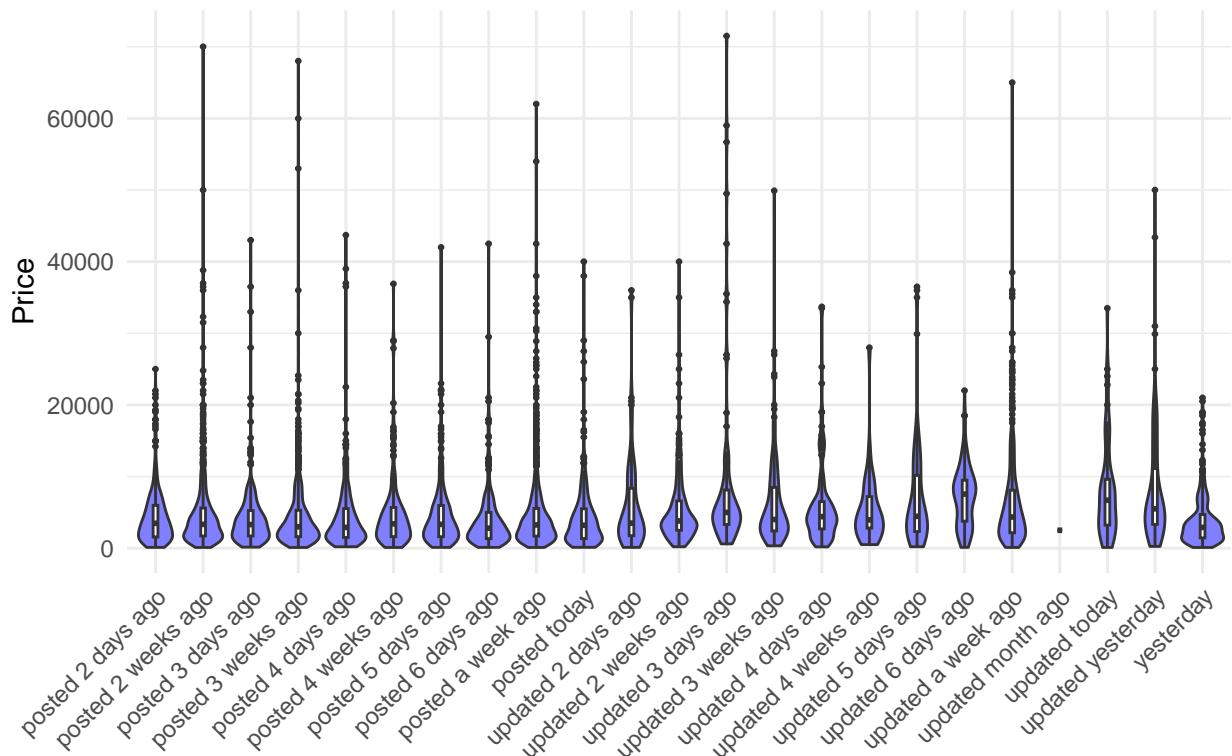
if (!is.factor(non_empty_data[[var]])) {
  non_empty_data[[var]] <- as.factor(non_empty_data[[var]])
}

# Create the violin + box plot
plot <- ggplot(non_empty_data, aes(x = .data[[var]], y = price)) +
  geom_violin(fill = "blue", alpha = 0.5) +
  geom_boxplot(width = 0.1, fill = "white", outlier.size = 0.5) +
  labs(title = paste("Price vs", var), y = "Price", x = "") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
print(plot)
}

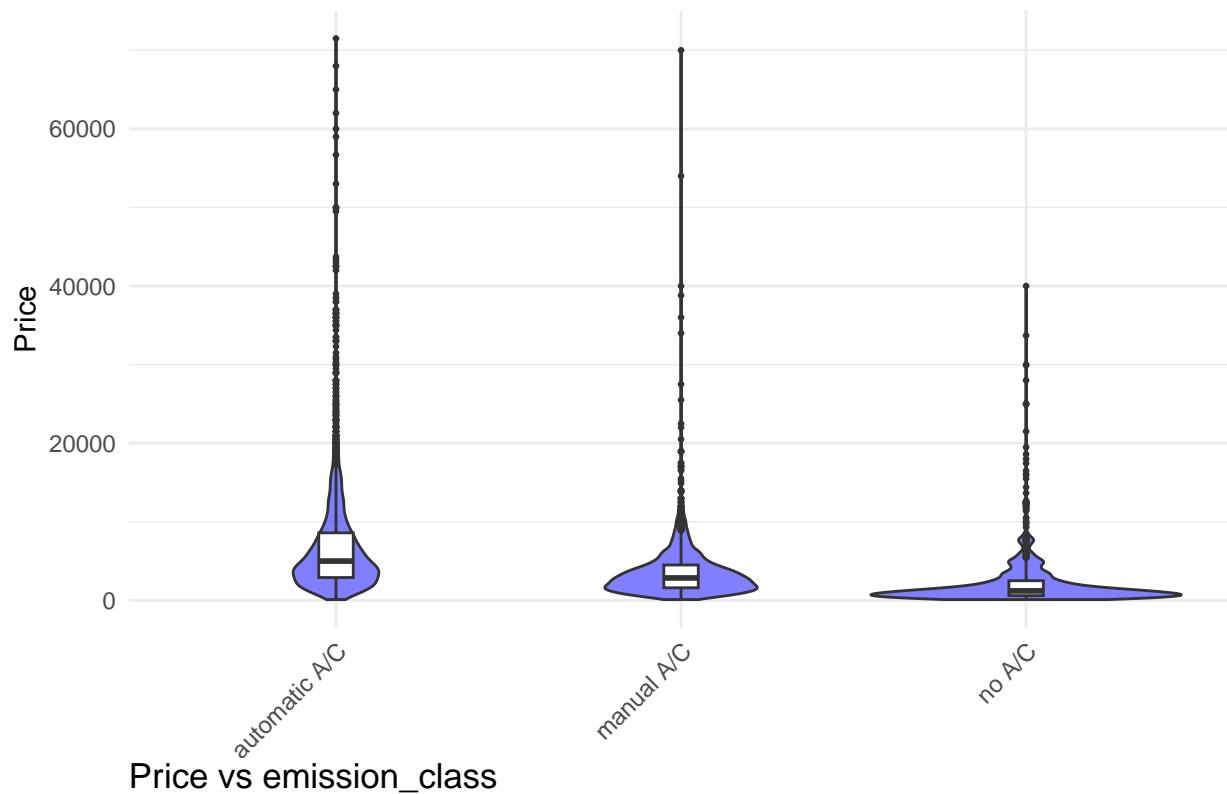
```

## Warning: Groups with fewer than two datapoints have been dropped.  
## i Set `drop = FALSE` to consider such groups for position adjustment purposes.

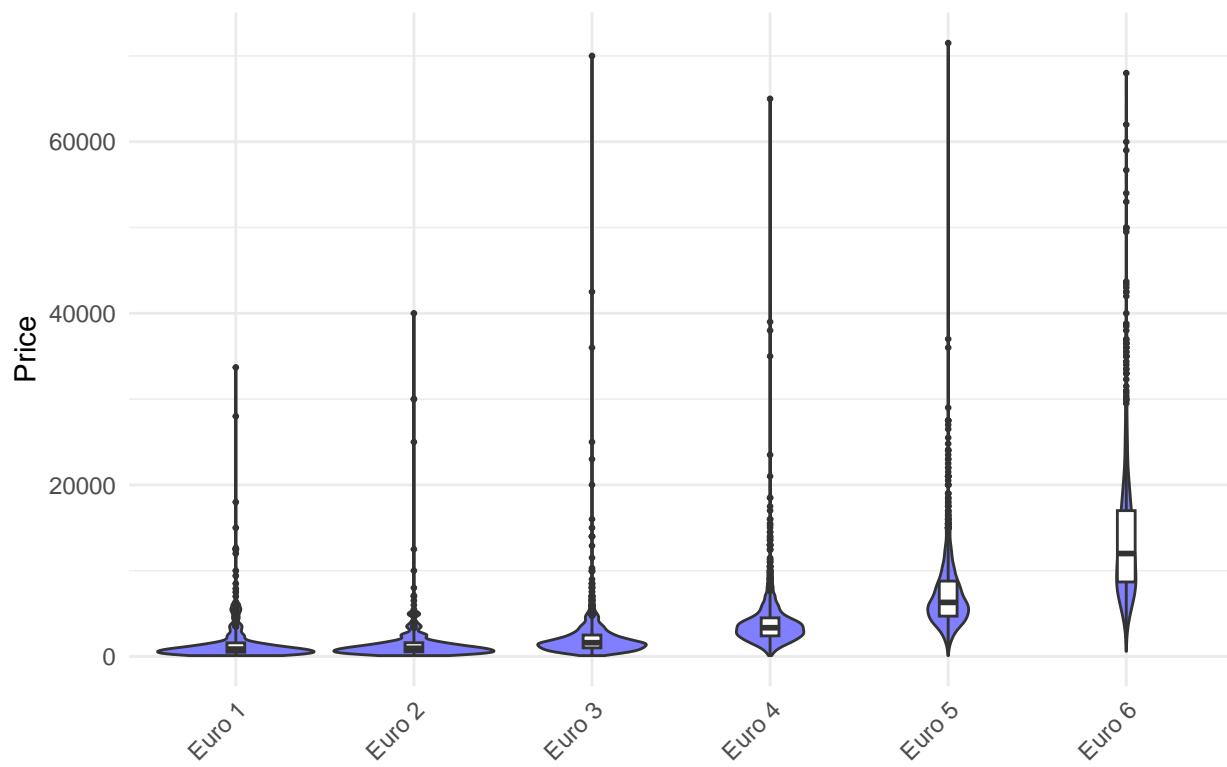
Price vs post\_info



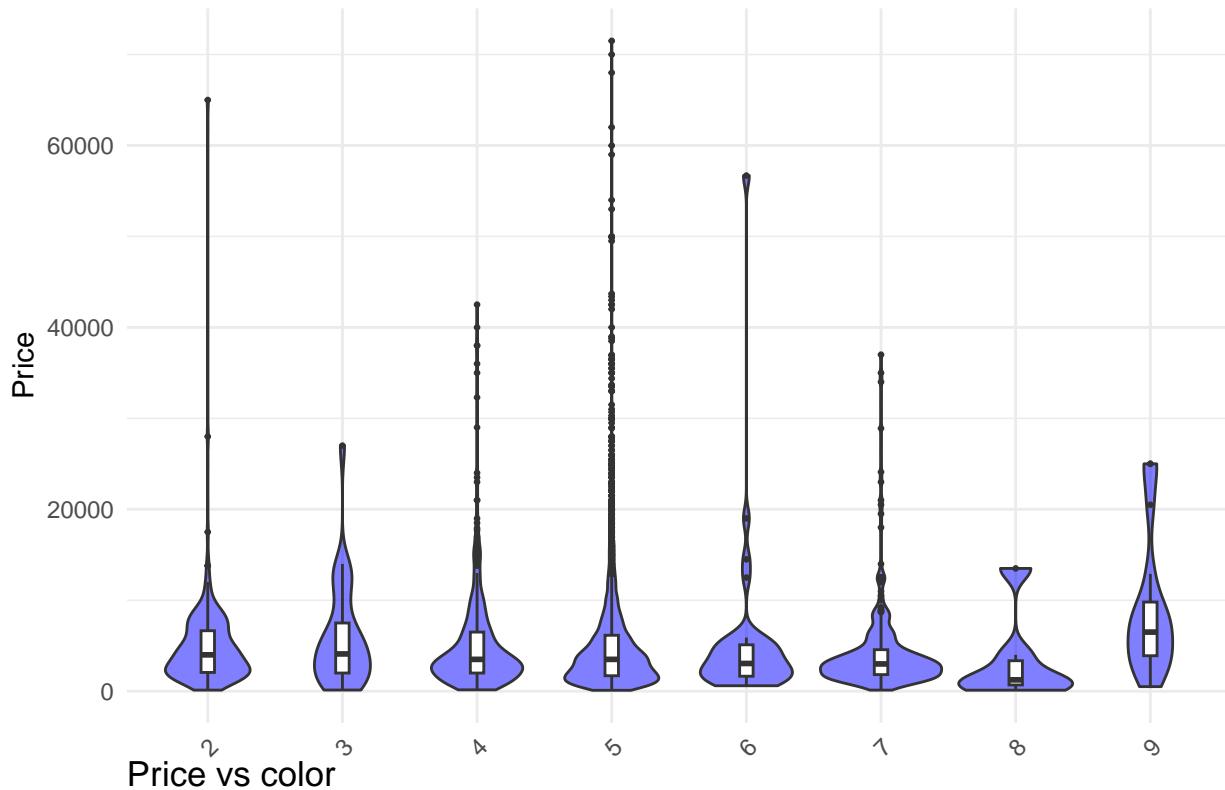
Price vs A\_C



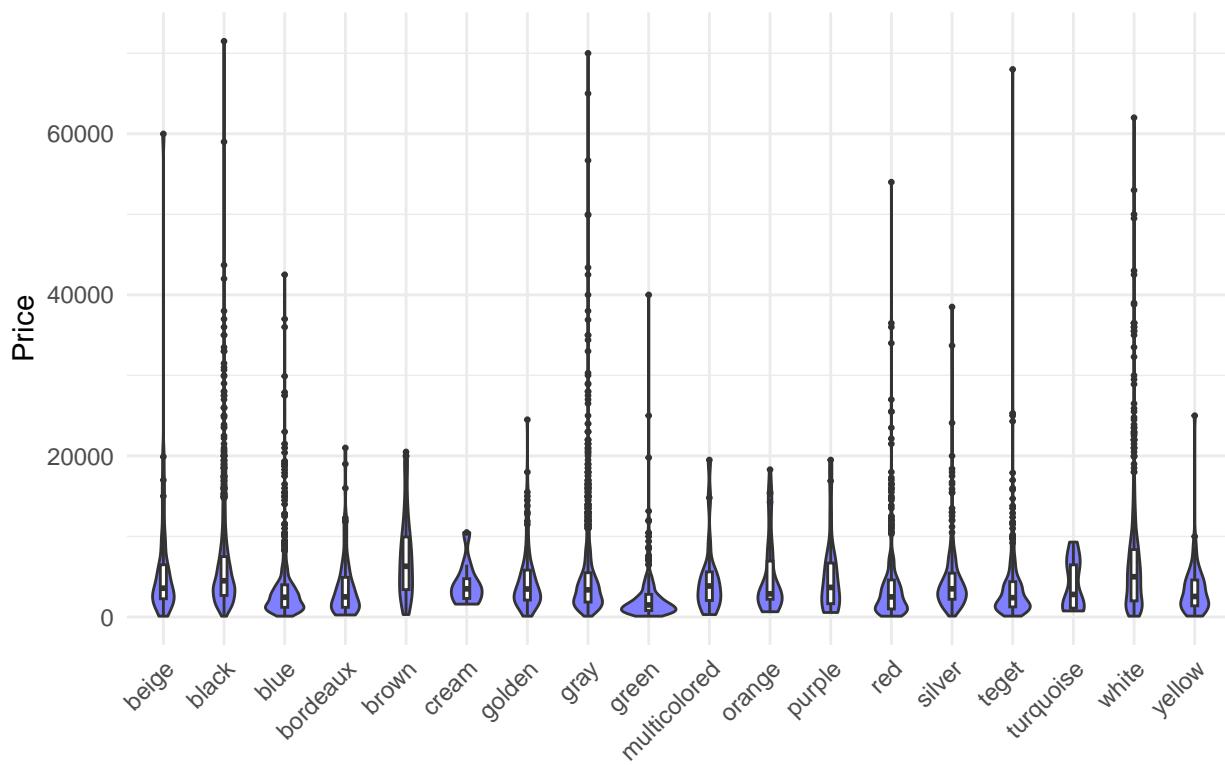
Price vs emission\_class



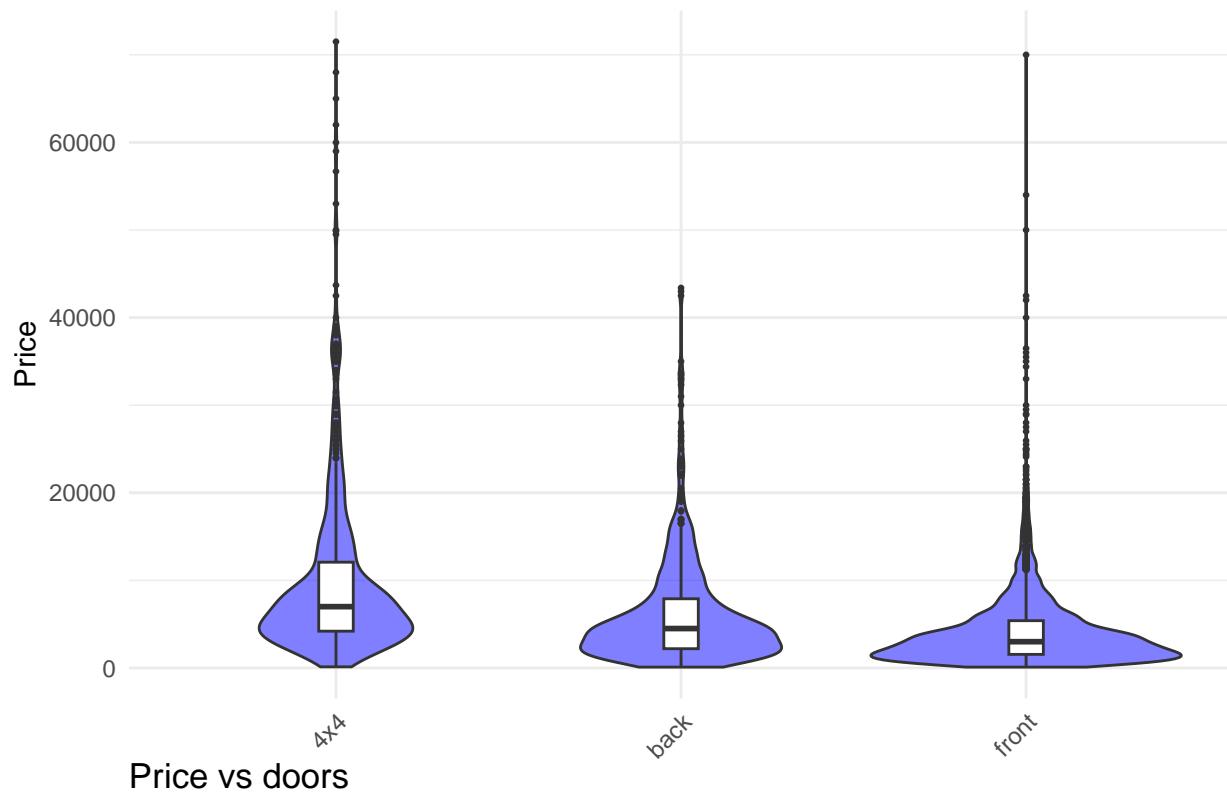
Price vs seats\_amount



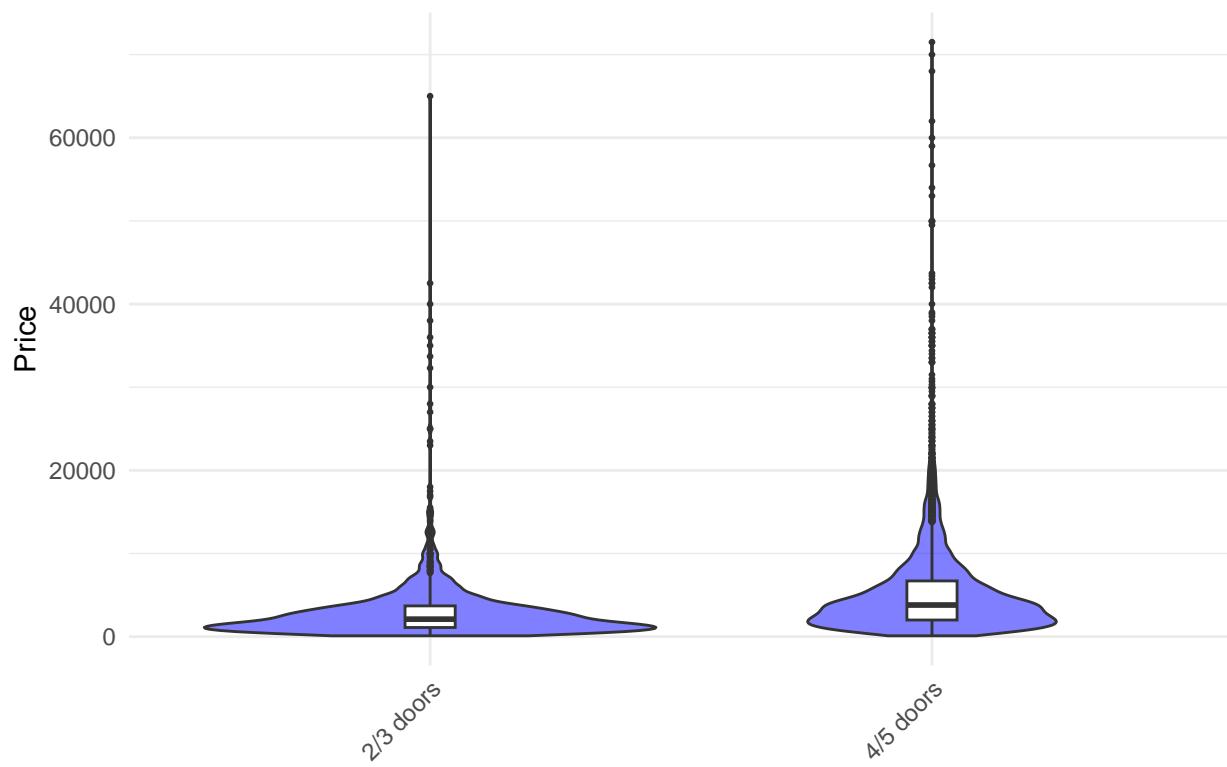
Price vs color



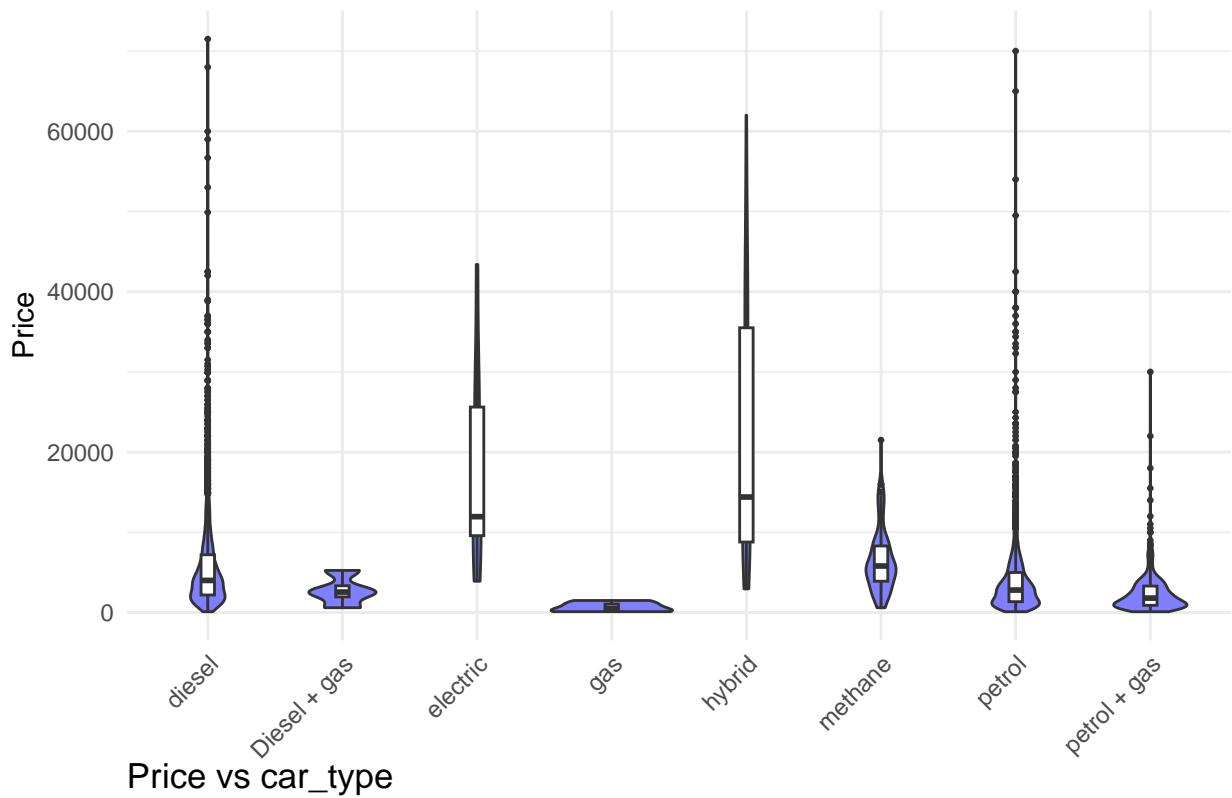
Price vs type\_of\_drive



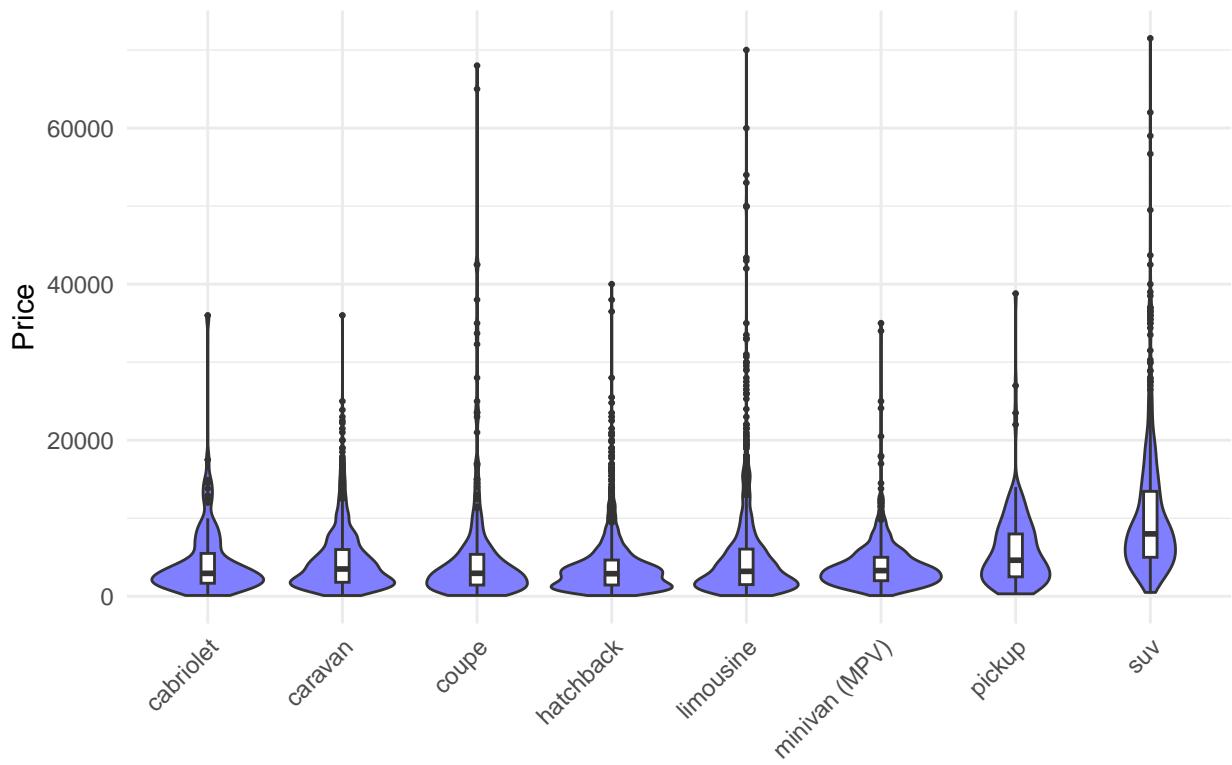
Price vs doors



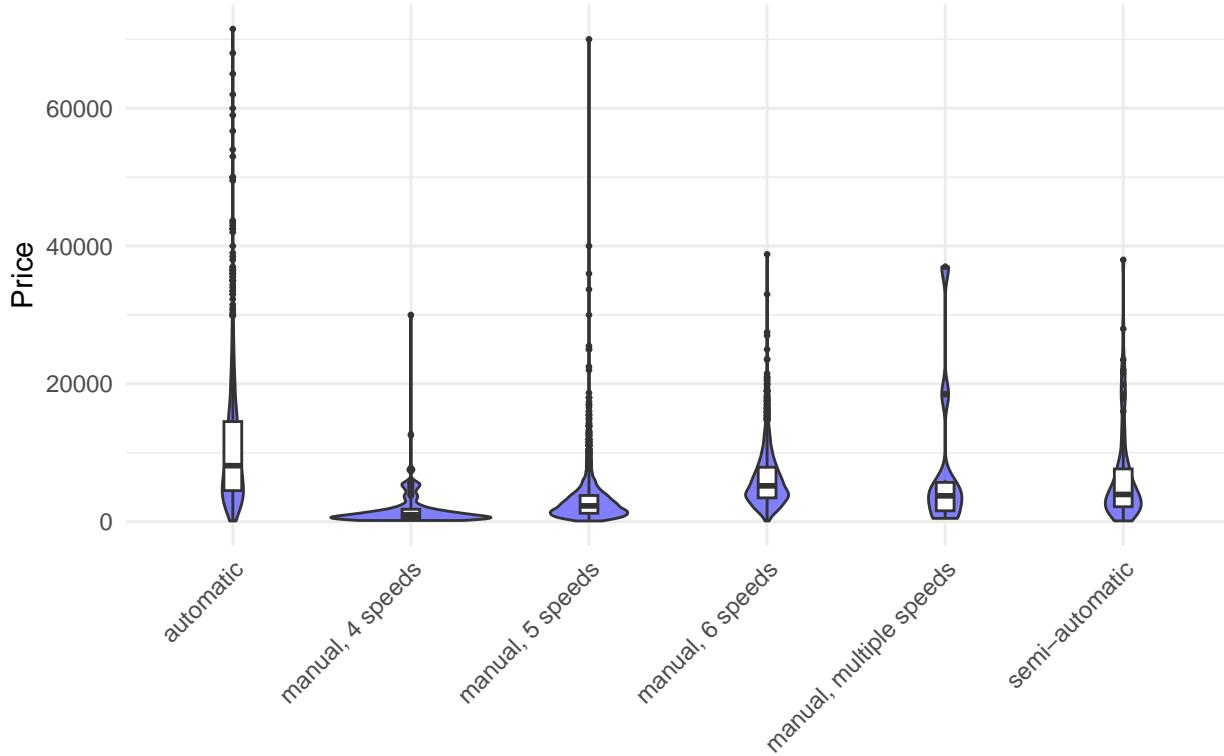
Price vs fuel



Price vs car\_type



## Price vs gearbox



[from here on is a load of garbage :( if you think you can help fix it, you're more than welcome. Though, it might be easier to just use this as code reference]

-Rebecca]

## Outlier Detection

[TO DO, pretty sure outliers are causing some of these other tests to look weird or become incomprehensible?]

Removing Leverage Points (outliers along X-axis):

Removing outliers along Y-axis:

Removing Influential Points:

## Data Analysis

[TO DO, need to analyse the variables by themselves - The violin plots are NOT what we want to see. Either they'll fix themselves after we remove outliers or we have to do something else]

Now, we want to take a look at the distributions of our data, to see if there are any peculiarities that we should be aware of. For continuous data: **age**, **horsepower**, **car mileage**, and **engine capacity**, we examine their violin plots:

```
non_empty_age <- clean_data[!is.na(clean_data$age) & !is.na(clean_data$age), ]  
  
age_graph <- ggplot(non_empty_age, aes(x = "", y = age)) +  
  geom_violin(fill = "purple", alpha = 0.5, trim=TRUE) +  
  geom_boxplot(width = 0.1, alpha = 0.8) +  
  labs(title = "Age Distribution", y = "Age", x = "")
```

```

non_empty_hp <- data.frame(horsepower = as.integer(clean_data$horsepower[clean_data$horsepower != ""]))

horsepower_graph <- ggplot(non_empty_hp, aes(x = "", y = horsepower)) +
  geom_violin(fill = "blue", alpha = 0.5) +
  geom_boxplot(width = 0.1, fill = "white", outlier.size = 0.5) +
  labs(title = "Horsepower Distribution", y = "Horsepower", x = "")

non_empty_cm <- data.frame(car_mileage = as.integer(clean_data$car_mileage[clean_data$car_mileage != ""]))

## Warning in data.frame(car_mileage =
## as.integer(clean_data$car_mileage[clean_data$car_mileage != : NAs introduced by
## coercion to integer range

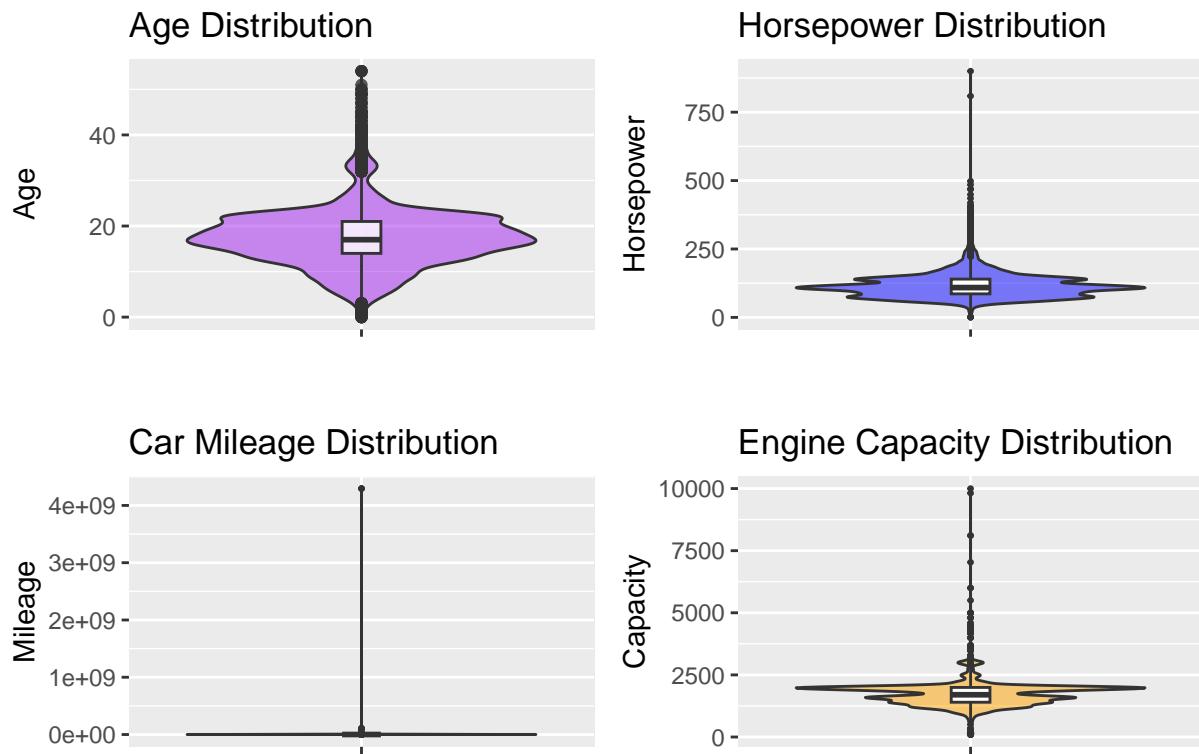
# Plot for "car_mileage"
car_mileage_graph <- ggplot(clean_data, aes(x = "", y = car_mileage)) +
  geom_violin(fill = "green", alpha = 0.5) +
  geom_boxplot(width = 0.1, fill = "white", outlier.size = 0.5) +
  labs(title = "Car Mileage Distribution", y = "Mileage", x = "")

non_empty_ec <- data.frame(engine_capacity = as.integer(clean_data$engine_capacity[clean_data$engine_capac

# Plot for "engine_capacity"
engine_capacity_graph <- ggplot(clean_data, aes(x = "", y = engine_capacity)) +
  geom_violin(fill = "orange", alpha = 0.5) +
  geom_boxplot(width = 0.1, fill = "white", outlier.size = 0.5) +
  labs(title = "Engine Capacity Distribution", y = "Capacity", x = "")

all_plots <- age_graph + horsepower_graph + car_mileage_graph + engine_capacity_graph + plot_layout(ncol = 2)
print(all_plots)

```



## Correlation Analysis

[Note from Rebecca (last person to work on this): this is a little broken right now.

- need to modify the correlation chart to get rid of some useless variables like views
- heatmap is not working I think bc there's a bunch of outliers in dataset. Going to clean out outliers first, then I'll get back to heatmap ]

```
numeric_data <- clean_data[sapply(clean_data, is.numeric)]
cor_matrix <- cor(numeric_data, use = "complete.obs")
```

```
ggcorrplot(
  cor_matrix,
  method = "square",
  type = "lower",
  lab = TRUE,
  title = "Correlation Heatmap",
  colors = c("blue", "white", "red"))
)
```

Correlation Heatmap



```
clean_data$log_horsepower <- log10(as.integer(clean_data$horsepower))
clean_data$log_car_mileage <- log10(as.integer(clean_data$car_mileage))
```

## Warning: NAs introduced by coercion to integer range

```
ggplot(clean_data, aes(x = log_horsepower, y = log_car_mileage)) +
  geom_bin2d(binwidth = c(10, 500)) + # Adjust bin width for better visualization
  scale_fill_gradient(low = "blue", high = "yellow") + # Density color scale
  labs(title = "Density of Horsepower vs. Car Mileage",
       x = "Horsepower", y = "Car Mileage", fill = "Count") +
```

```
theme_minimal()
```

```
## Warning: Removed 3 rows containing non-finite outside the scale range
## (`stat_bin2d()`).
```

Density of Horsepower vs. Car Mileage

