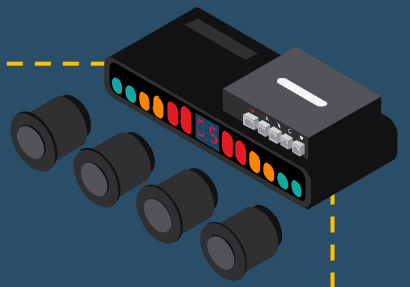
Automotive Predictor

Zalan Kaman	3119759
Michał Krzyżański	3172260
Jakub Małachowski	3163969
Roberto Jose Monaco	3174180
Romeo Felice Heukamp	3188553
Luis Manuel Pericchi Marrero	3164306

Table of Contents

Introduction



Overview

We are using a dataset which contains all car listings of the United States posted on Craigslist, which has the world's largest collection of used vehicles. It includes relevant information about the cars for sale such as: price, model, condition, odometer, etc... The dataset has 426 880 data points with 26 columns.

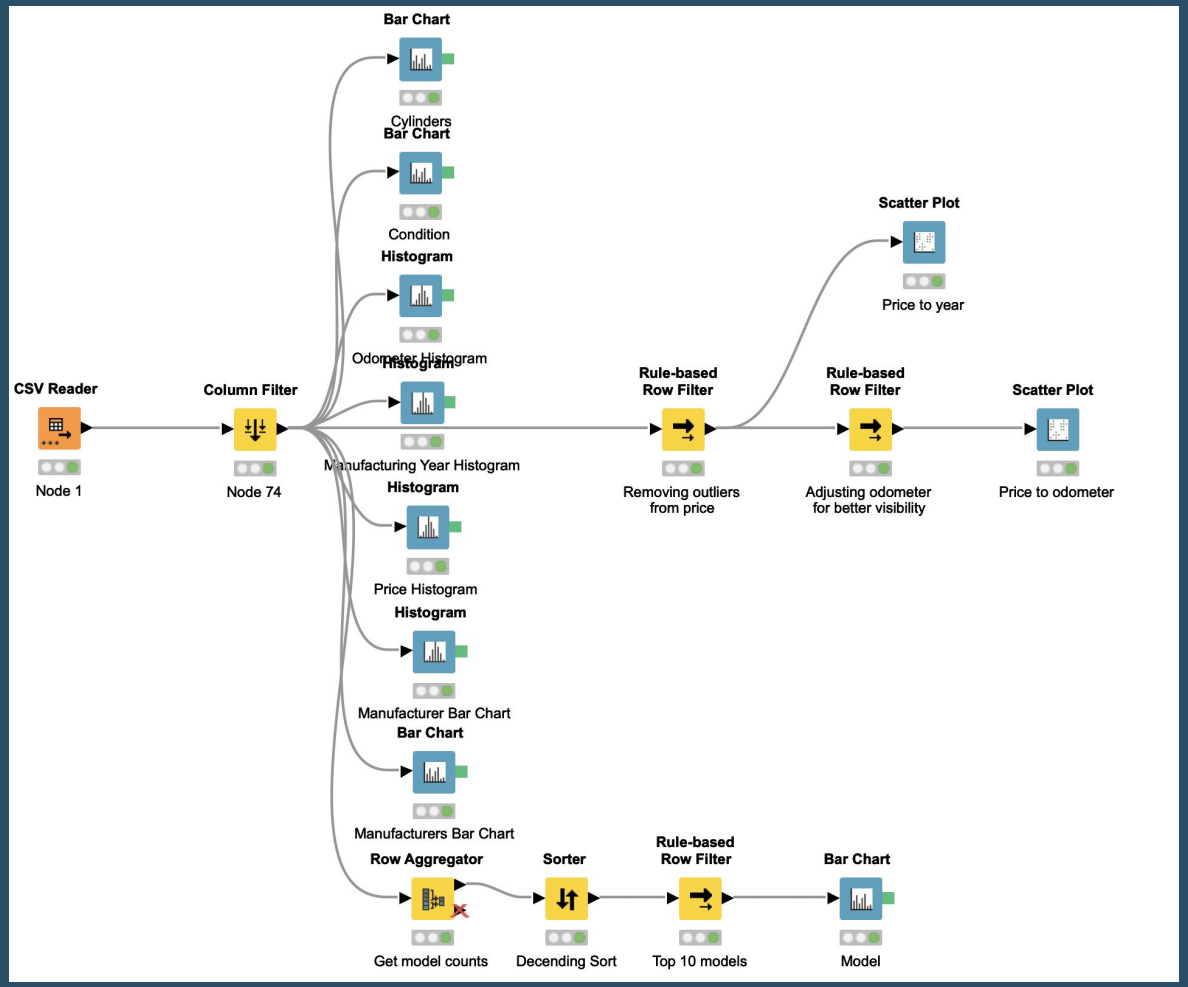
We want to use this data to help a dealership make purchasing and pricing decisions. The goal is for the dealership staff to use our model before buying a car they will resell and before pricing their existing fleet. As such, our target variable will be price. This can also help identify regions where cars are underpriced.

We first performed data description and data preparation in order to be able to use wisely the following tools to predict the optimal price for different cars: Linear Regression, Lazy Model Comparison with Random Forest, and a Neural Network.

Data Description

Univariate Analysis

Data Description Workflow



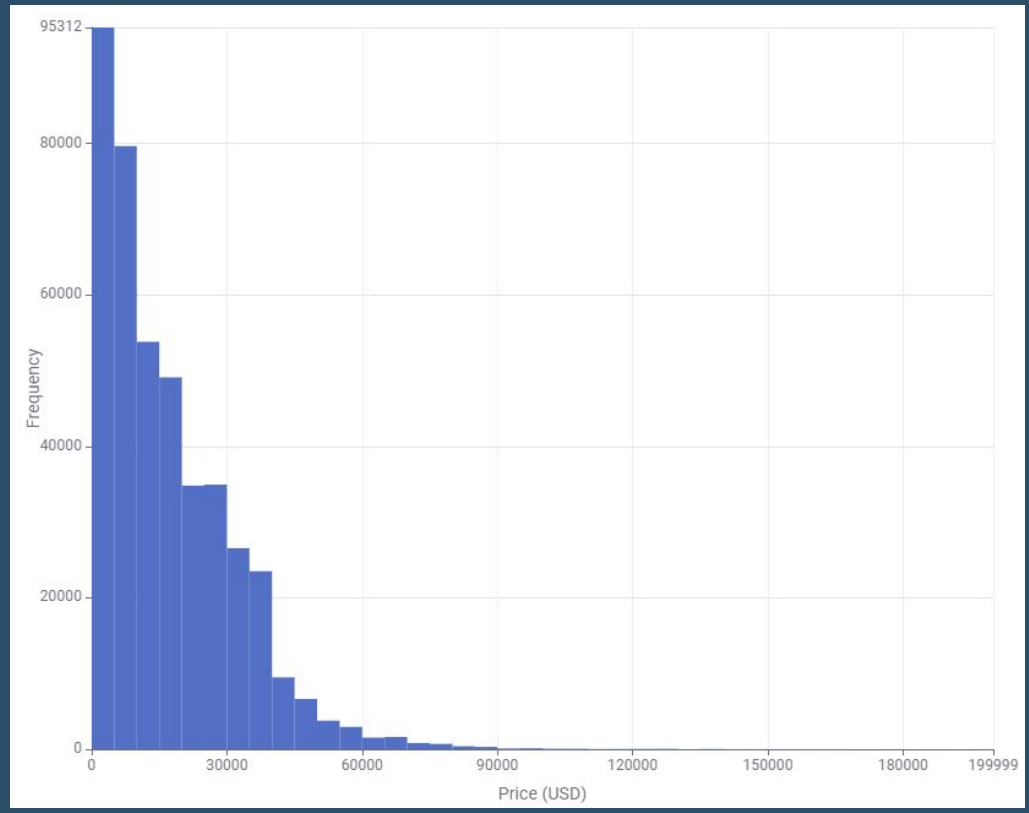
Price - Target Variable

Nature: Numerical, discrete

Description: price (in USD) as posted on the listing

Range: 0 - 3,736,928,711

Insight: data is right-skewed, some cars have prices set to 0, most cars with prices above $200K (124 of them) are likely to be wrong inputs.



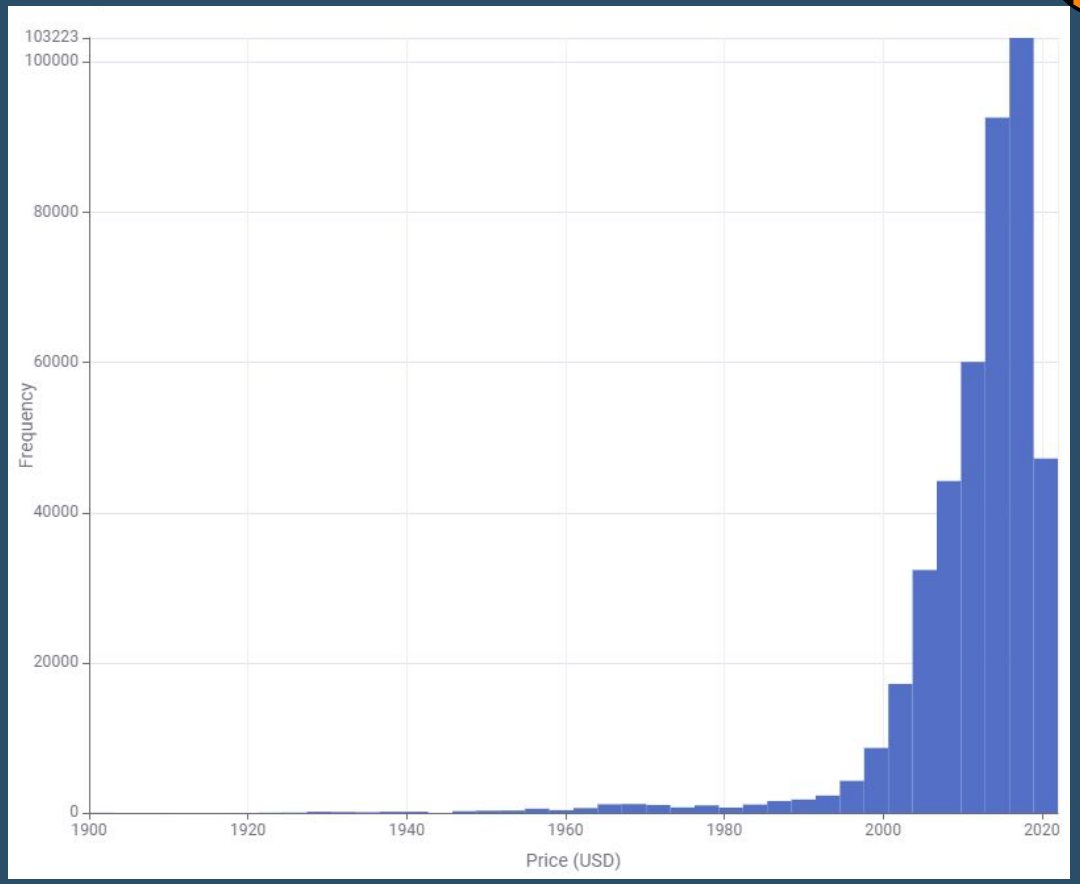
Manufacturing Year

Nature: Numerical, discrete

Description: year in which the car listed was manufactured

Range: [1900 - 2022]

Insight: We observe that most cars were manufactured in 2018 and 2019. Funnily we can observe the effects of the Great recession as in 2008 and 2009 new vehicle sales fell nearly 40 percent in the US.



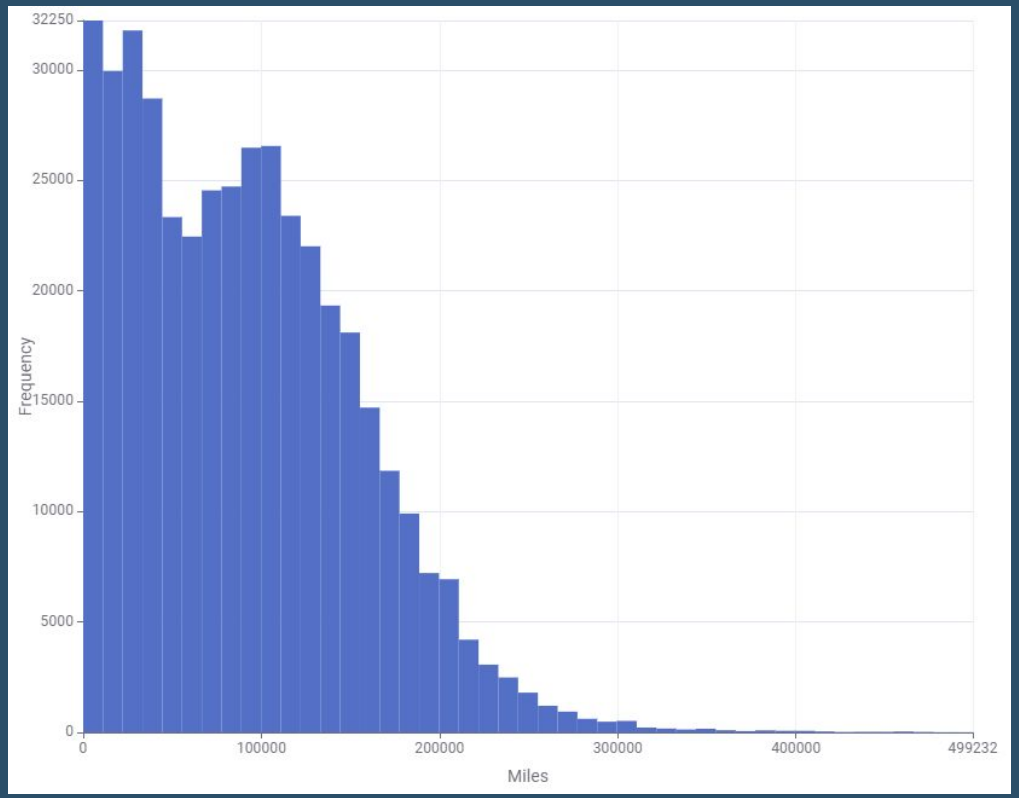
Odometer

Nature: Numerical, continuous

Description: mileage of each car listed

Range: 0 - 10,000,000

Insight: data is right-skewed, many cars sit closest 0 before a dip down, then back up. We believe this may be cause by the fact that many cars are sold brand new and many after initial leases expire. We think that there is a honeymoon period where the car is not brand new but not quite used either.

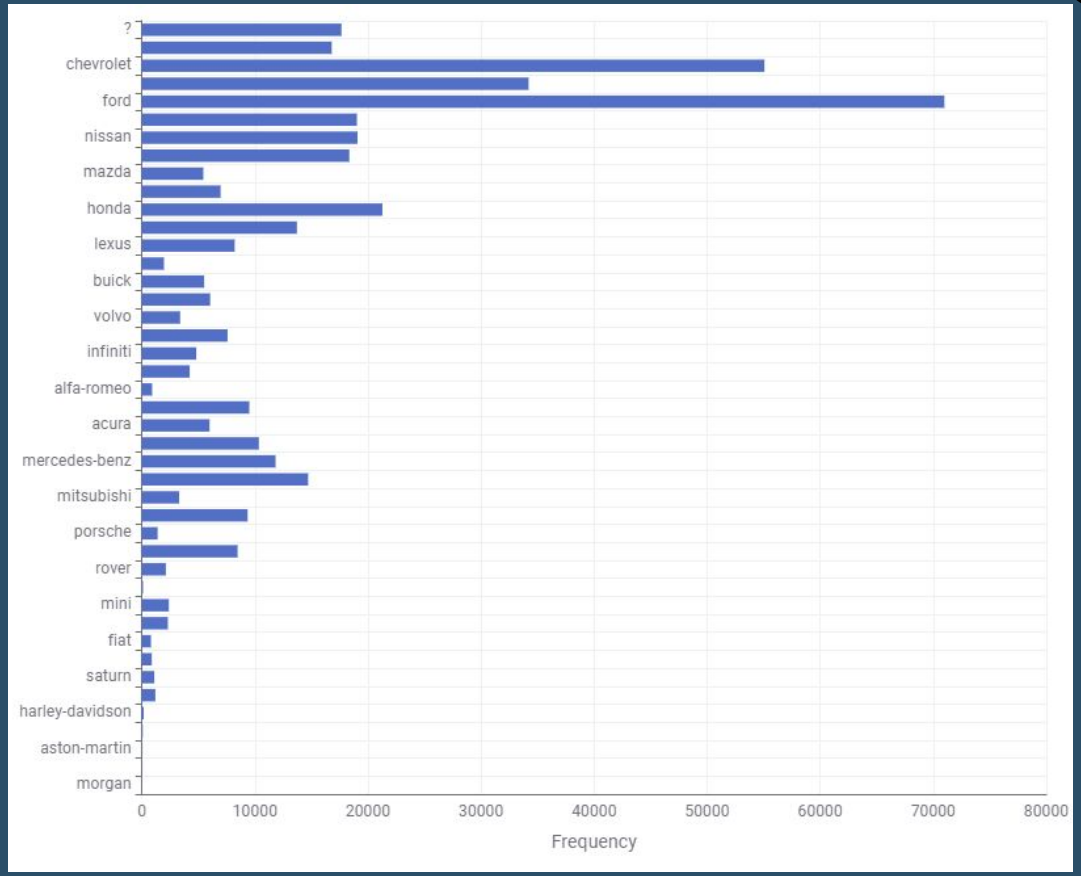


Manufacturer

Nature: categorical data

Description: There are 42 manufacturers in our dataset

Insight: Ford and Chevrolet, two iconic american brands, are the biggest manufacturers in the postings, having combined more postings than the next 5 manufacturers summed together. We also see that the Japanese manufacturers: Toyota, Nissan, and Honda are all in the top 5 most listed cars

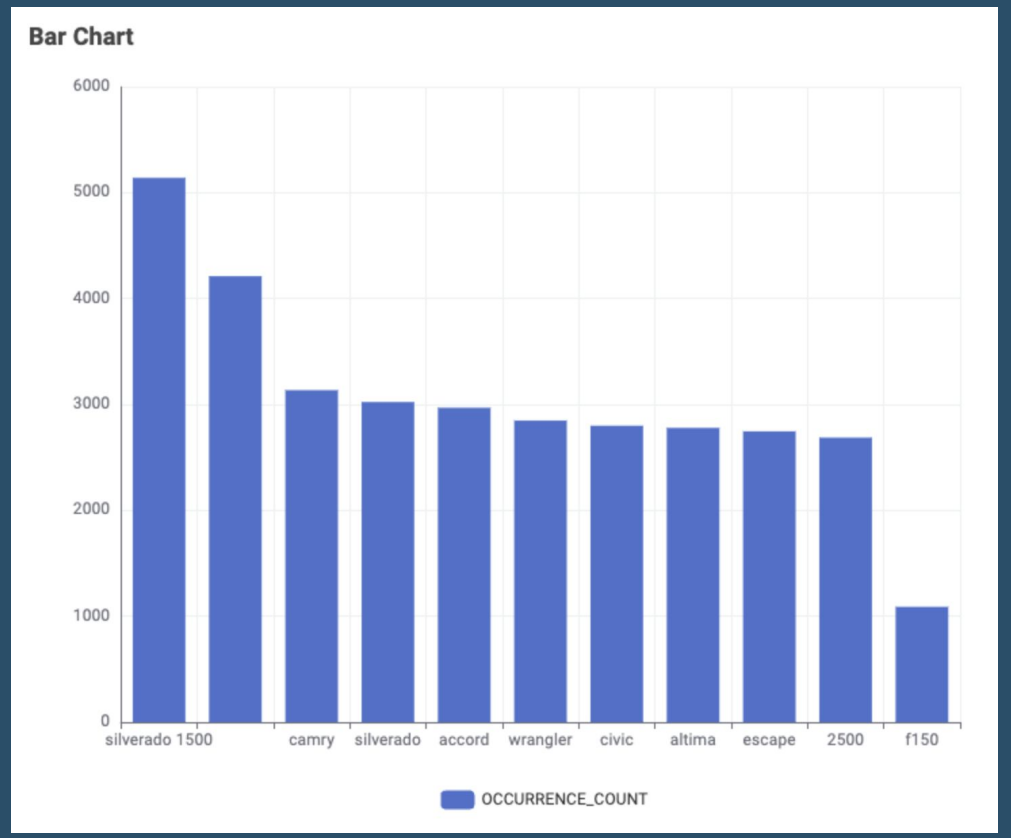


Models

Nature: Categorical

Description: Name of the 29667 models of cars in our dataset

Insight: We observe again a domination of ford and Chevrolet models. We note that the Silverado 1500 is represented 3 times written differently. Aggregated it becomes the top model. We also observe the amount of Japanese cars liken the Camry, Civic and Altima are all on the top list as well. However, there are 29667 unique models, which will need to be addressed in the data preparation.

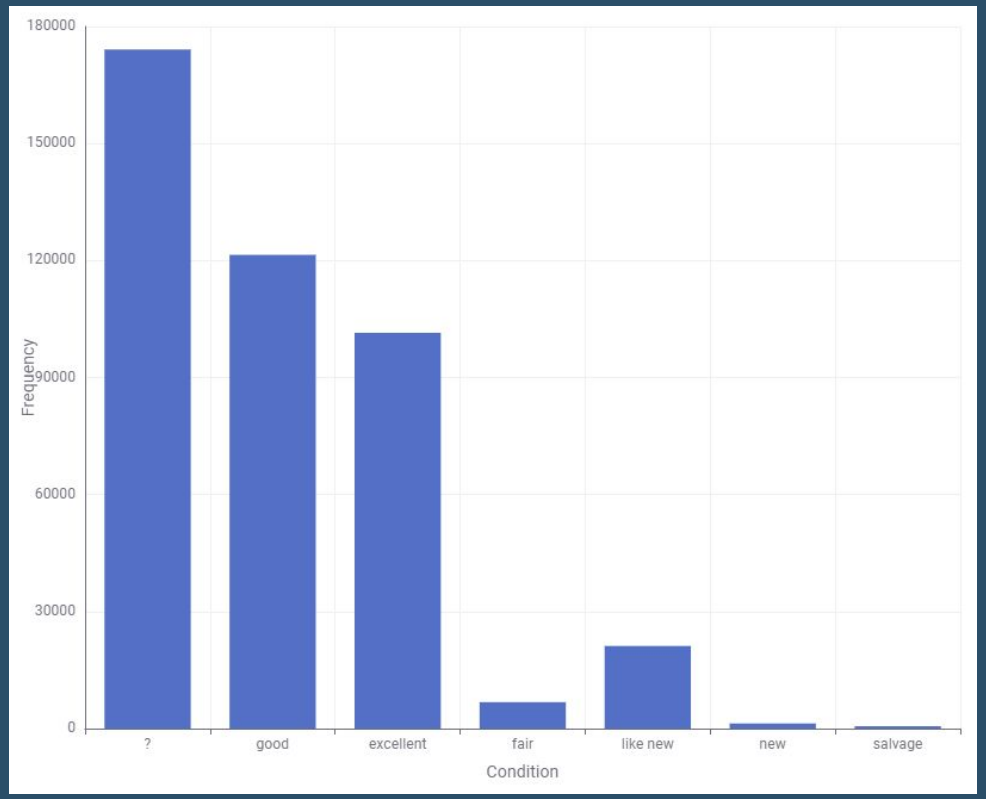


Condition

Nature: categorical

Description: count of cars according to 6 categories: salvage, fair, good, excellent, like new, new.

It is important to note that these categories can be subjective, specially when listing your own car, where owners can reevaluate the condition of their car or even be untruthful.

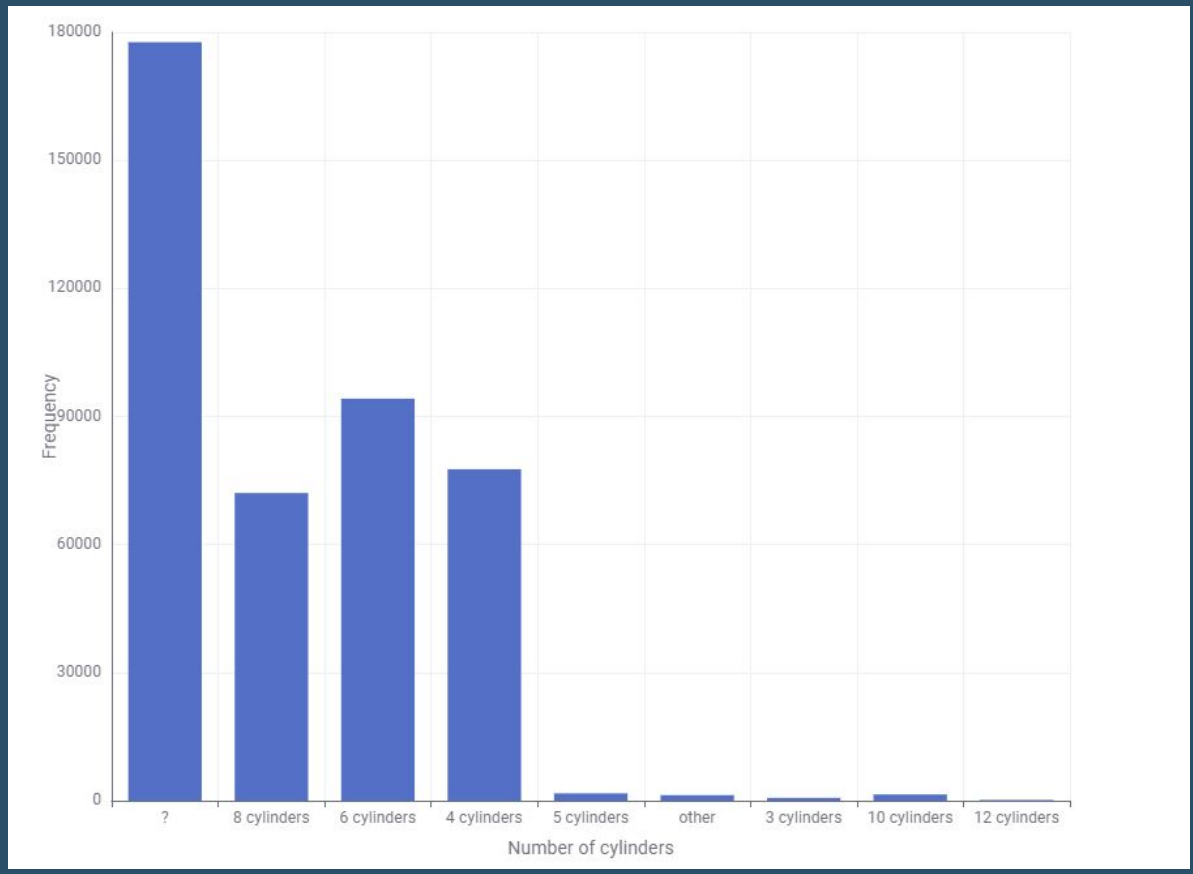


Cylinders

Nature: Categorical

Description: The amount of cylinders in each car (8 possibilities)

Insight: The amount of cylinders is representative of engine size, and in most cases is also a good indicator of power, which is important for estimating price. As we see the majority of cars have either 4, 6, and 8 cylinders which is the most common in contemporary cars. Older cars may be the one with lower cylinders, and race cars the one with high number of cylinders.

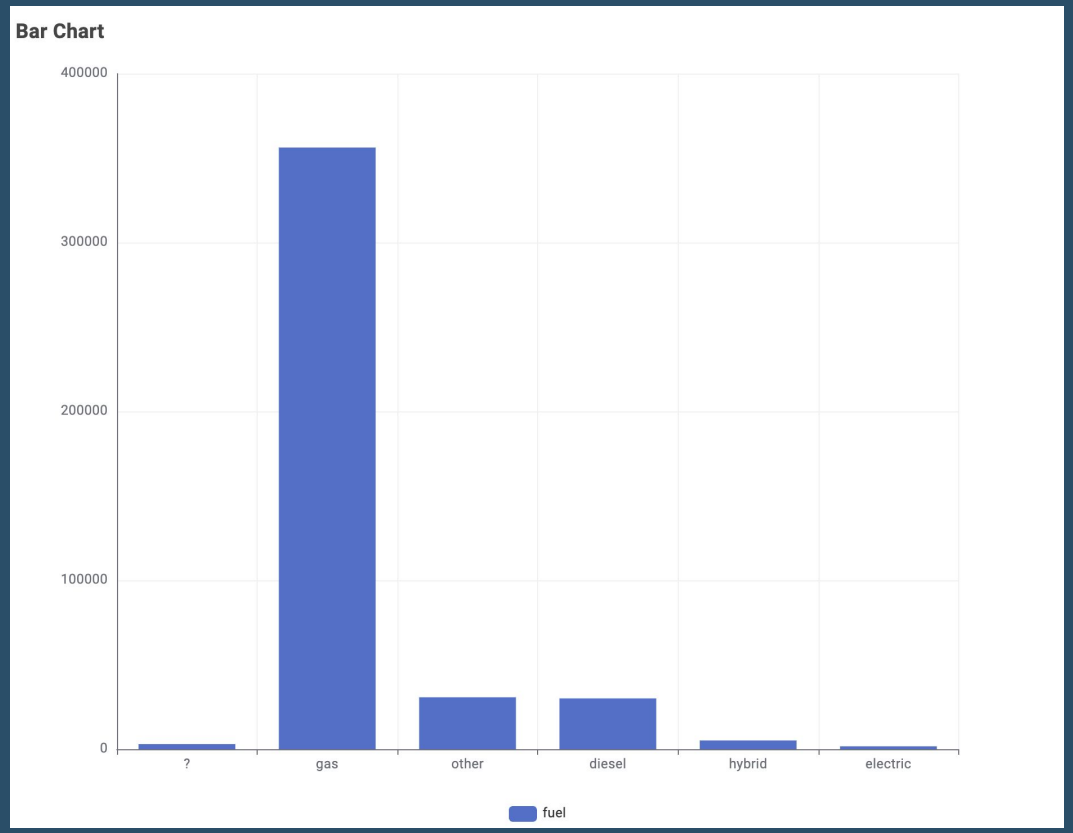


Fuel

Nature: Categorical

Description: The type of fuel needed to run the car (5 fuels)

Insight: Majority of cars are run in gas since it has always been the most conventional fuel. Few cars are electric specially since the production of electric cars had an increase in last years and before that almost all cars used fossil fuels.

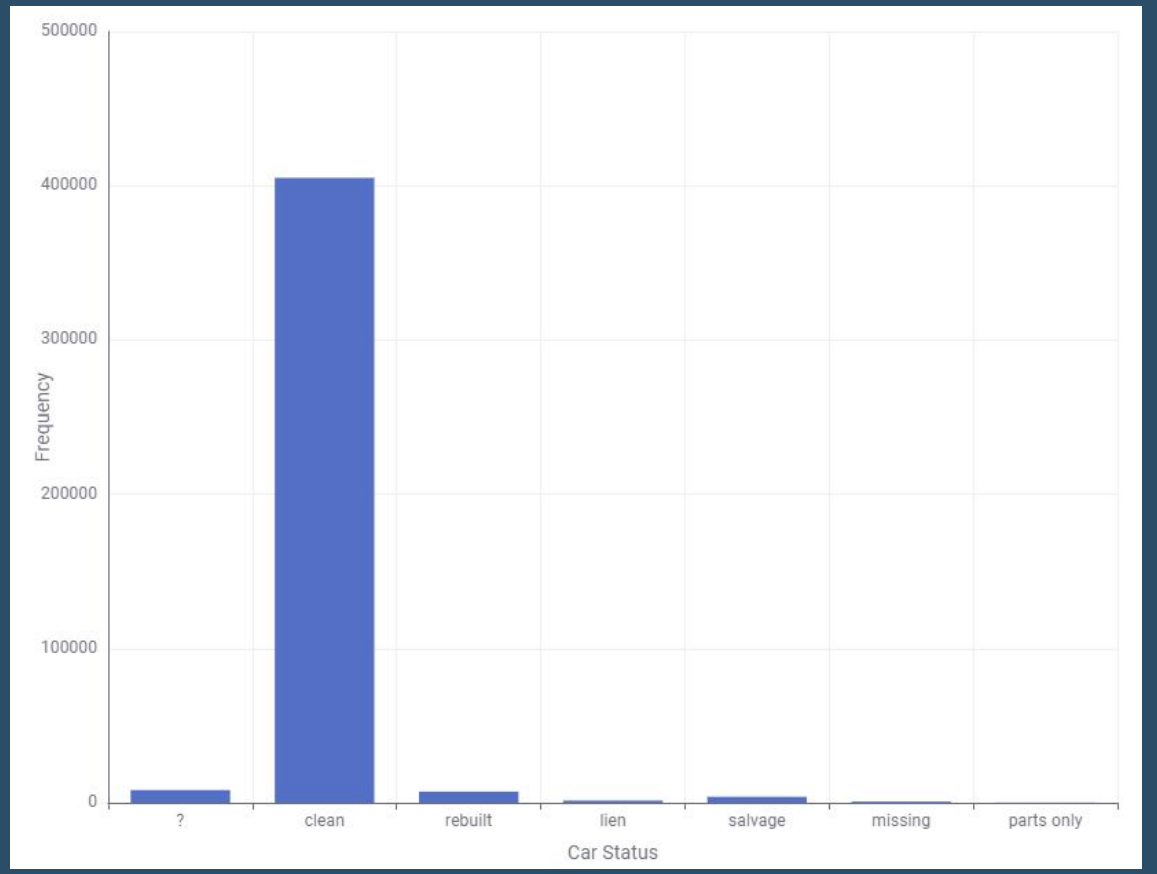


Title Status

Nature: Categorical

Description: There are certain legal statuses that a car will always fall under. Typical undamaged cars will have clean titles, while cars that have undergone bad damaged will have salvage titles. A rebuilt title is granted when formly salvage cars pass roadworthiness testing after being heavily redone.

Insight: Nearly all data points are clean vehicles, which are of course the most desirable titles for customers.

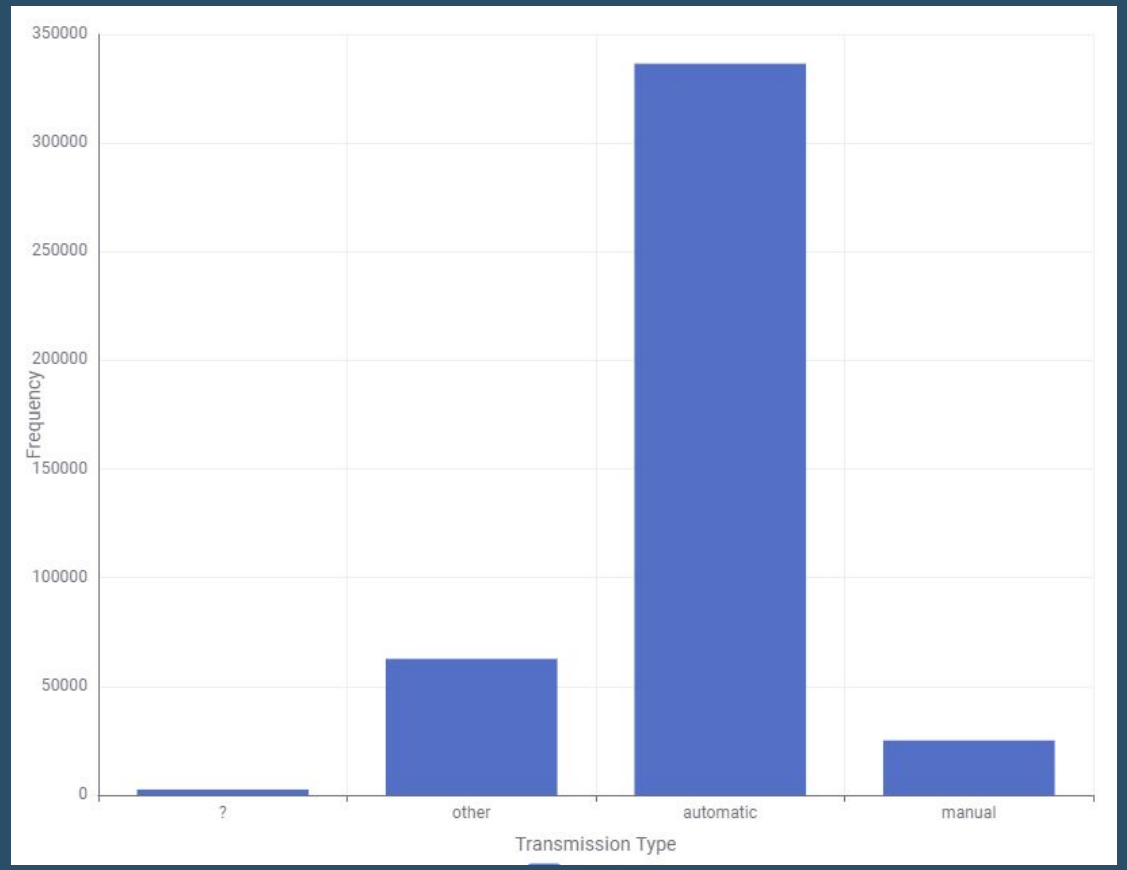


Transmission

Nature: Categorical

Description: Shows if the car transmission

Insight: Majority of cars are automatic and probably majority of buyers will only look for automatic cars.

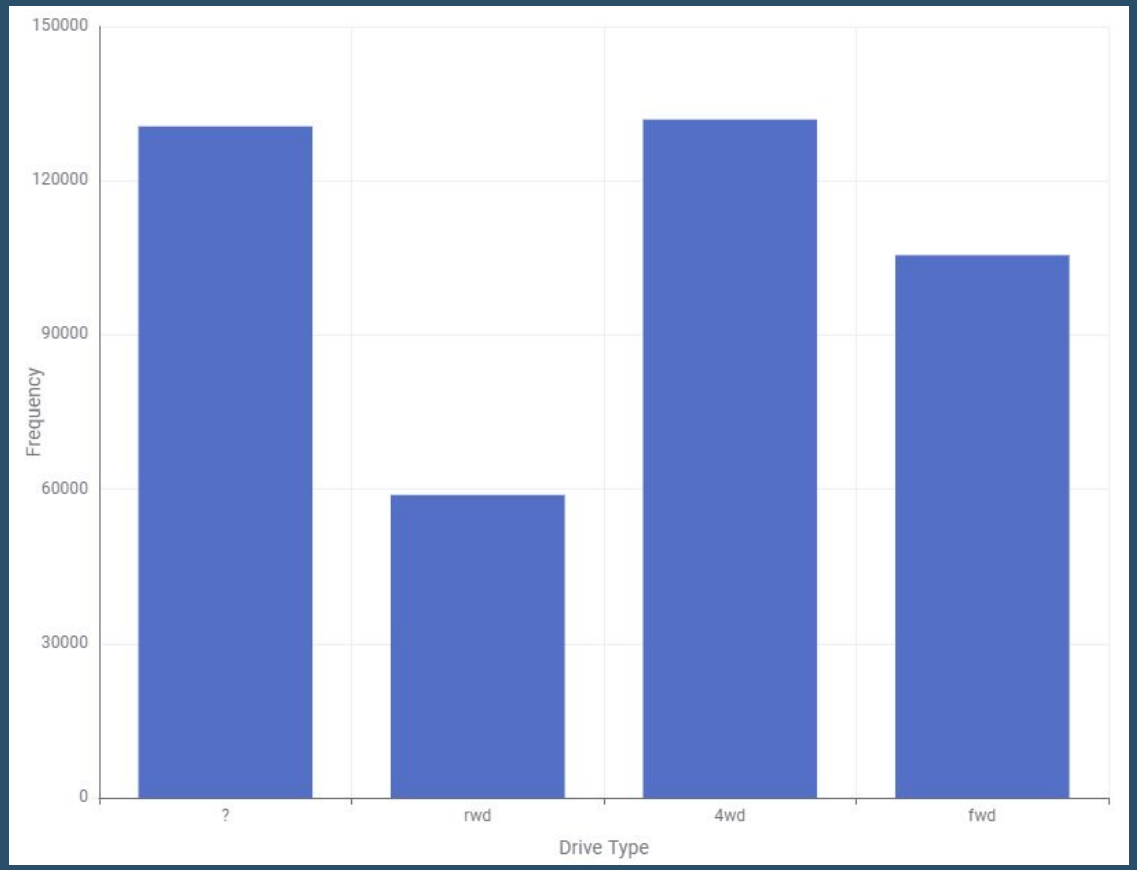


Drive

Nature: Categorical

Description: 4WD is the type of care which uses all four wheels to move. It is a popular choice for off-road vehicles. FWD is most common for small compact vehicles that tend to be lower in price. Sports cars often come in RWD.

Insight: Most of data falls into 4WD while least into RWD. This sounds right as RWD is a rarer more desirable trait.

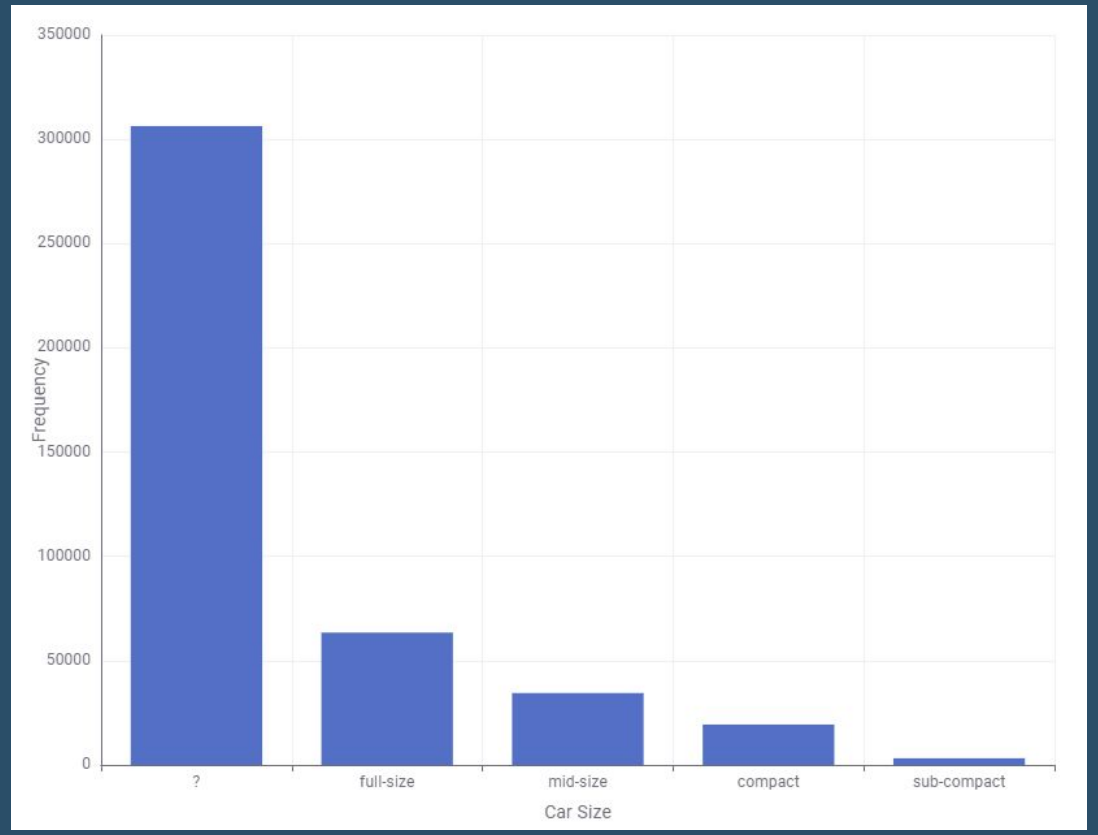


Size

Nature: Categorical

Description: Size of the car listed

Insight: The majority of cars are full size or mid-size, few are compact and sub-compacts are a big minority

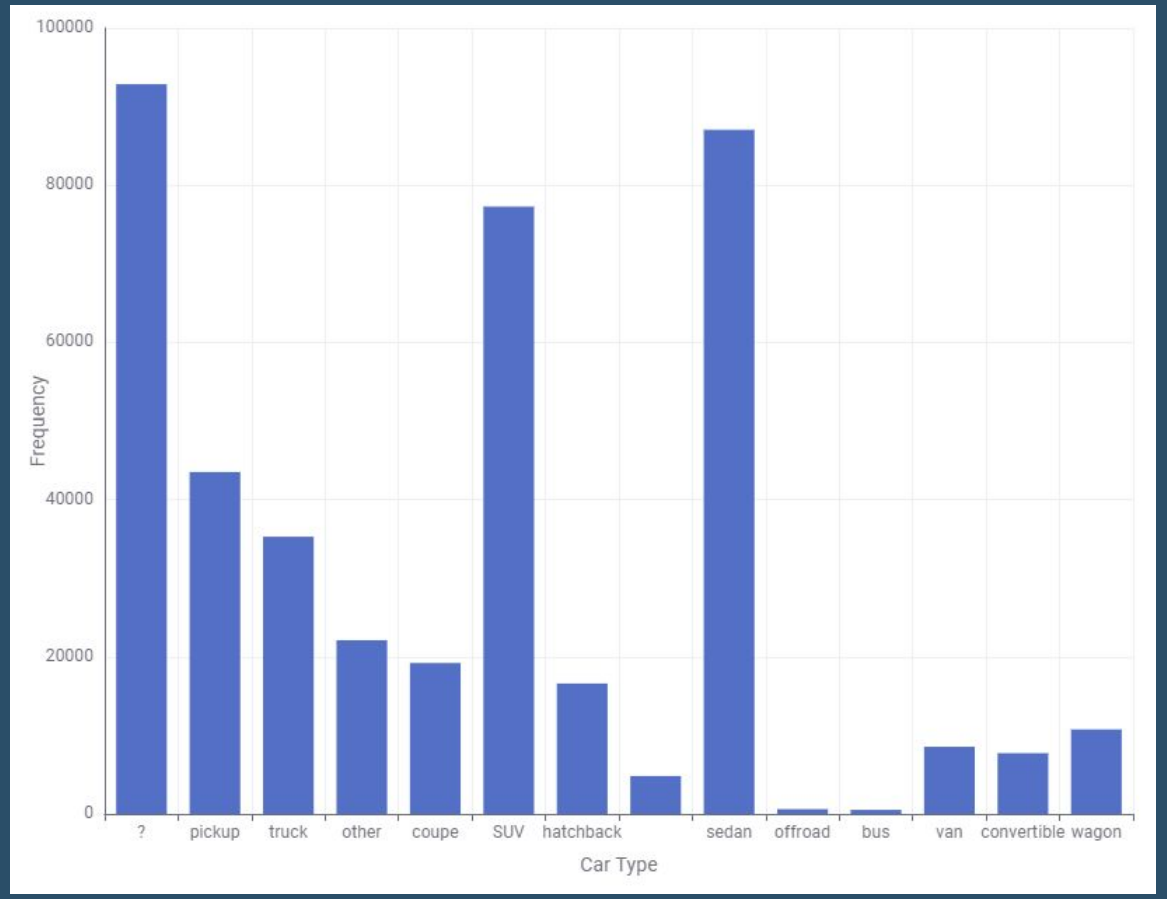


Type

Nature: Categorical

Description: Type of car listed

Insight: There are 13 types of car in our dataset with “Sedan” and “SUV” as the top types of car in the dataset.

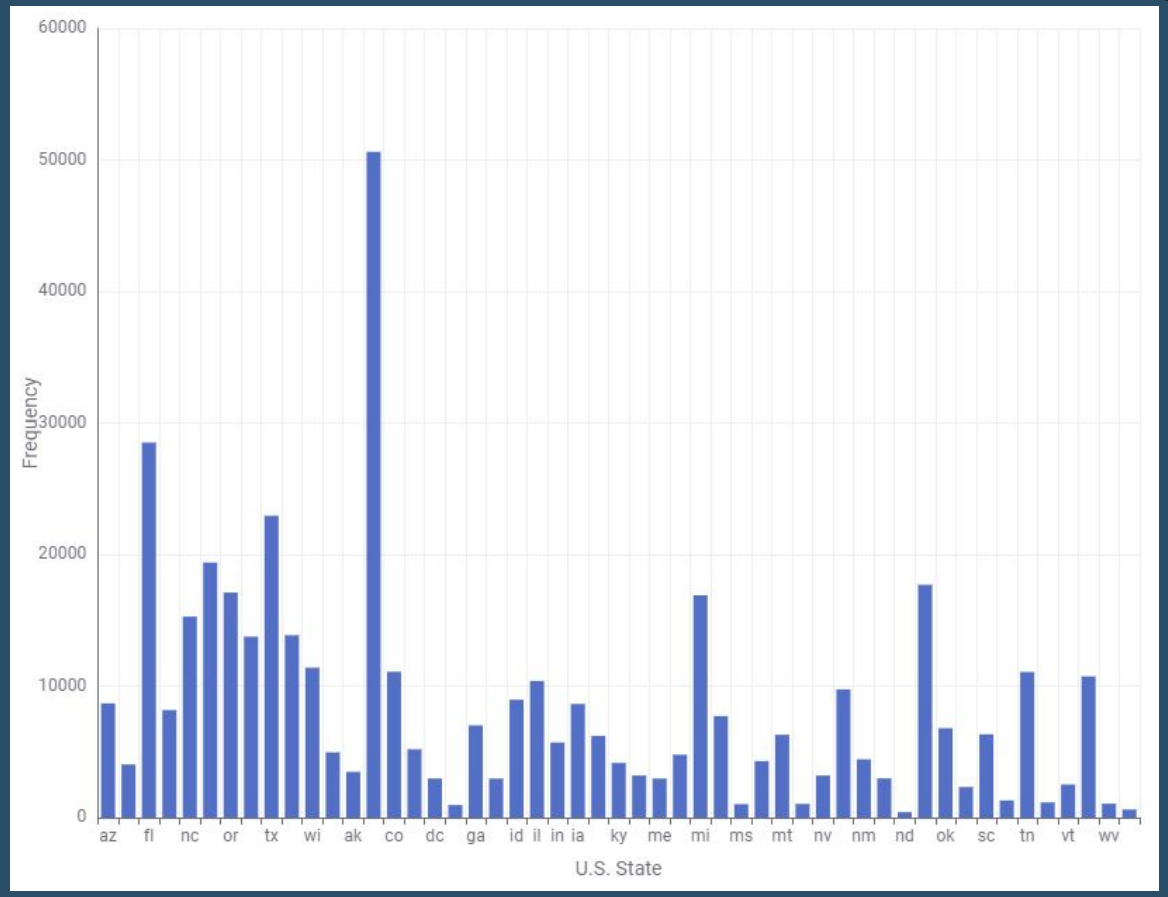


State

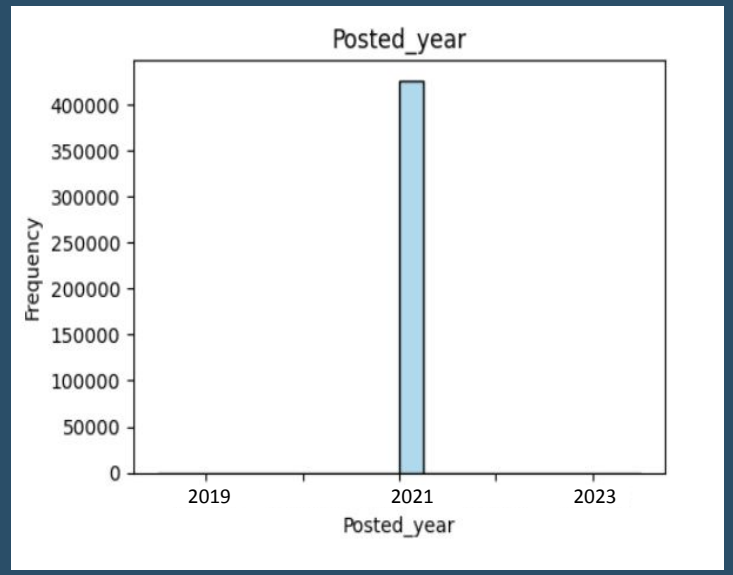
Nature: Categorical

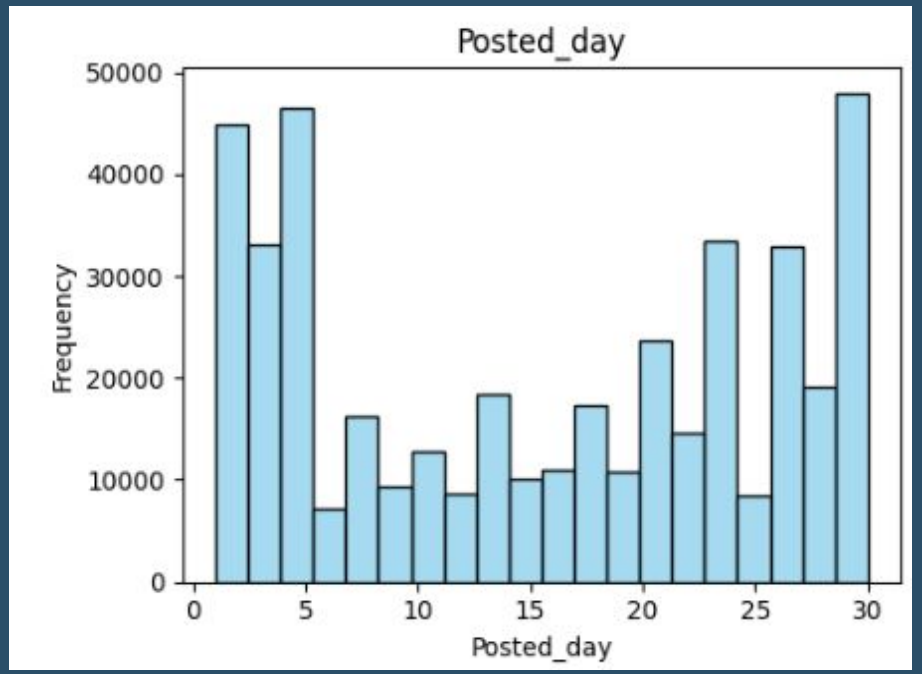
Description: The 50 US states and the District of Columbia in which the car has been listed

Insight: The amount of listings in each state reflects the population and expected sales, like California having the highest number of listings



Post Date





Nature: Numerical, discrete

Description: All the postings were done during the month of April 2021. We observe a concentration of postings at the start and end of the month.

Insight: This variable does not carry much information as the postings are very concentrated. It is much more insightful to look at car age, which is going to be in relation to 2021 for all postings.

ID

Nature: Numerical
Description: ID given to each listing in the dataset
Insight: There are 426880 unique IDs

URL

Nature: String
Description: URL from each listing in the dataset
Insight: There are 426880 unique URLs

Region

Nature: String
Description: Region of the car's seller location
Insight: There are 404 unique Regions

Region URL

Nature: String
Description: URL given to the region of the car's seller location
Insight: There are 413 Region URL, meaning there are datapoints with a Region URL but a missing value in the Region column

VIN

Nature: String
Description: The identifier of the vehicle
Insight: There are 118264 unique VIN, majority is missing one.

Paint Color

Nature: Categorical
Description: Color of the vehicle
Insight: There are 12 colors

Image URL

Nature: String
Description: URL of the image in the listing
Insight: There are 241899 image URLs

Data Preparation

Data Preparation

Dropping Variables and Rows

- We will identify variables and rows not useful for our analysis.

Car Model Mapping

- We will use NLP to identify actual car models from scraped car model descriptions.

Imputation

- We will leverage the actual car models for data imputation.

Feature Engineering

- We will create new features useful for accurate prediction of car prices.

Outlier Analysis

- We will identify and clean the outliers in our target variable price.

Encoding

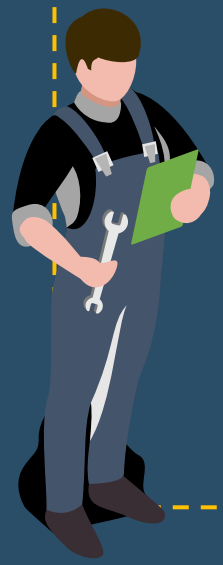
- We will transform the categorical variables in order to actually use them.

Dropping Rows

Throughout the Data Preparation chapter, we drop a lot of observations - a total of 20% of original dataset. While we understand its a lot and observations in general should not be dropped, we elaborate on why such decision was made at every step.

The choice to drop observations was crucial for our specific application. We intend to provide car dealerships with a tool predicting car prices as accurately as possible, not to predict new unseen Craigslist car listings. The dealers would input the car's specifications to out model to output the predicted price.

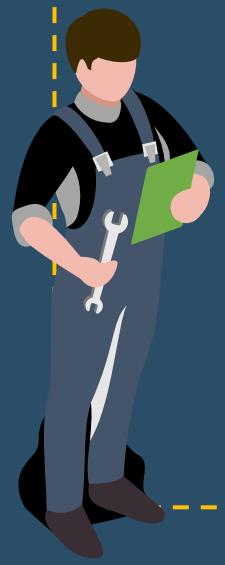
With that in mind, we delete observations that introduce more noise than information into the overall model.



Dropping Variables

- We removed the variables URL and VIN since they accomplish the same job as ID, and Region URL which is the same as the Region variable.
- Our models also do not need to use of Image URL, since it is not in our scope.
- County is completely empty.
- The post date are all on the same month, providing no valuable information.

These variables do not add any value or accuracy to our model. Adding them would be unnecessary.



URL

Region URL

Vehicle Identification Number (VIN)

Image URL

County

Post Date

Car Model Mapping Idea

To address the issue of having 30 thousand unique car model descriptions, we decided to use a secondary data source found on OpenDataSoft. We will refer to this dataset as the Car Model Dataset (CMD). It contains all car models including their specifications, manufactured since 1984. We will try to match our raw car model descriptions to actual car models.

If we identify the car models correctly, then we can additionally use specific information included in the Car Model Dataset, such as whether a specific model has front-wheel drive or its number of cylinders, accurately imputing the missing values.

We decided to identify all unique car models and use a NLP model – Sentence Transformer – to predict which car model the seller most likely meant in the car model description.

Car Model Mapping Steps

Installing the model library

Obtaining unique car model descriptions

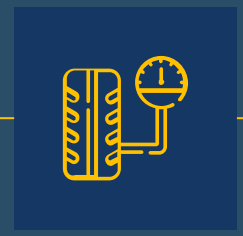
Cleaning car model data for training

Creating the manufacturer+model strings

Creating a model mapping dictionary

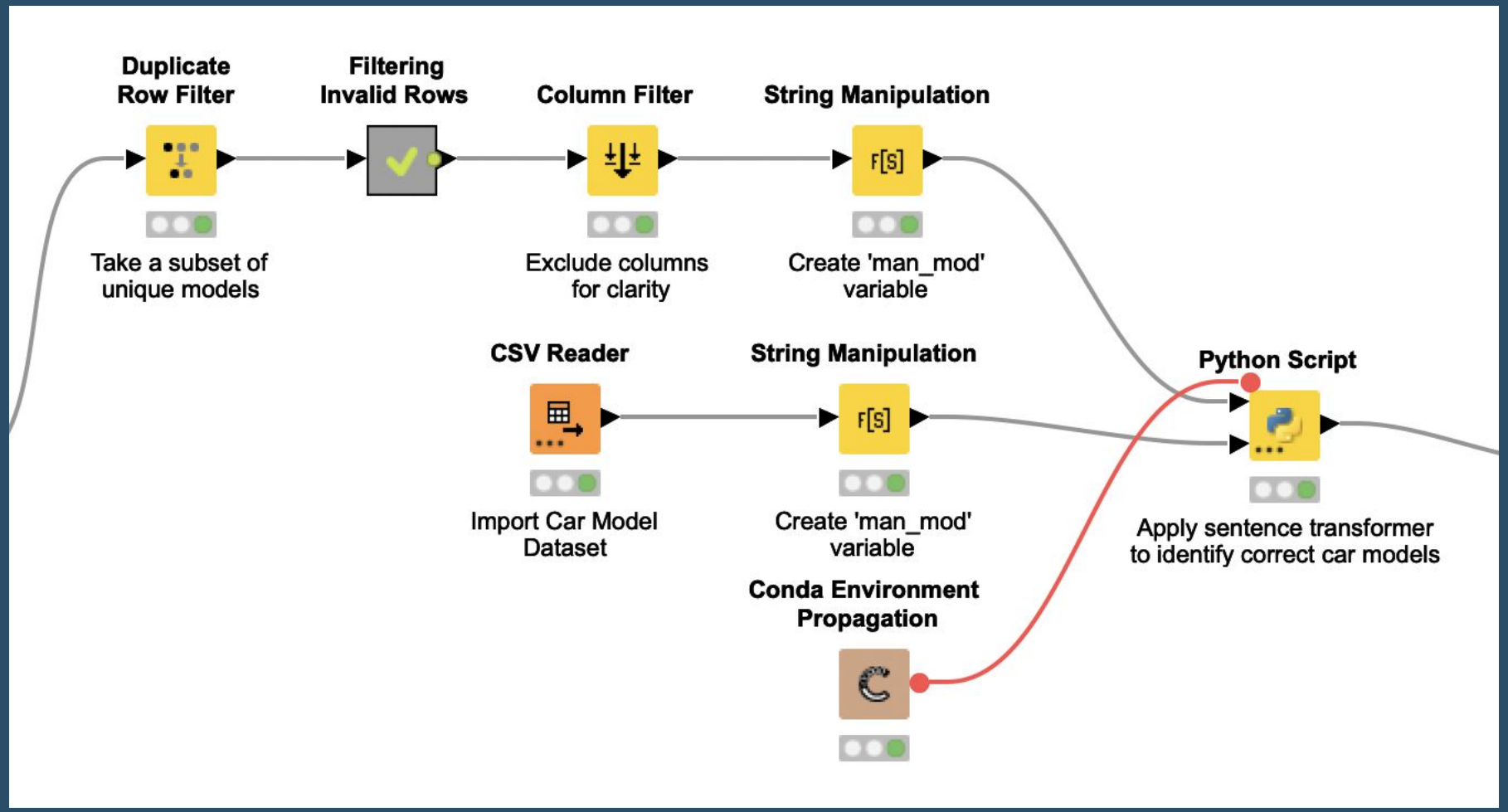
Choosing the model with highest predicted probability

Creating the similarity matrix for the car models



Encoding the car models in the sentence transformer

Car Model Mapping Steps



Car Model Mapping

We begin with obtaining a subset of our dataset, with only the unique 'model' variables – the 30 thousand unique car model descriptions – to make it computationally feasible on 30K rows rather than the 400K of the whole dataset (Figure 1).

We proceed with some basic cleaning (Figure 2). We get rid of all the cars without manufacturers, as they're crucial in identifying correct models, as well as cars manufactured before 1984, because these models will not get identified either way, as they're not in the Car Model Dataset (CMD). We also get rid of all the motorcycles accidentally scraped to our dataset and unnecessary columns for clarity.

At last we get rid of unnecessary columns just for this process (Figure 1).

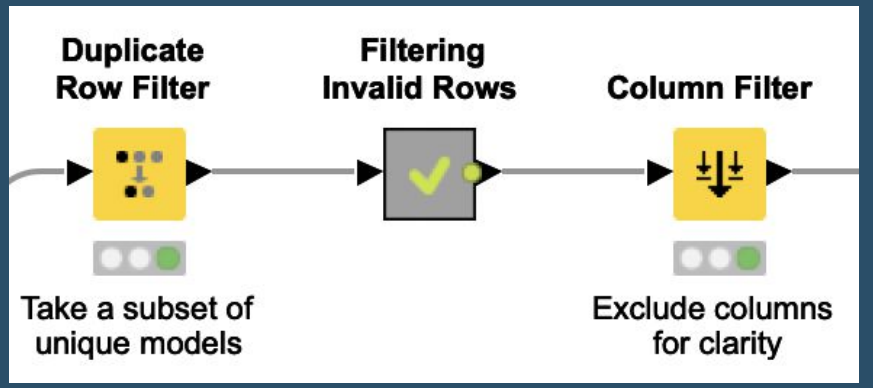


Figure 1.

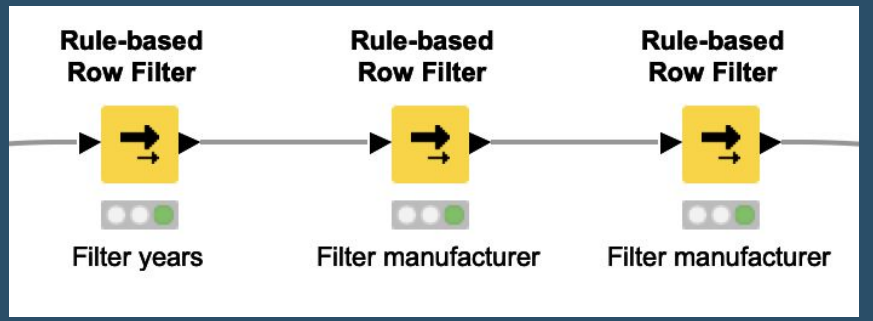
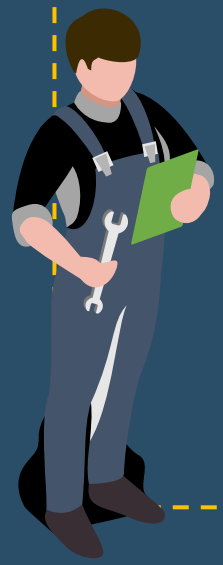


Figure 2.



Car Model Mapping

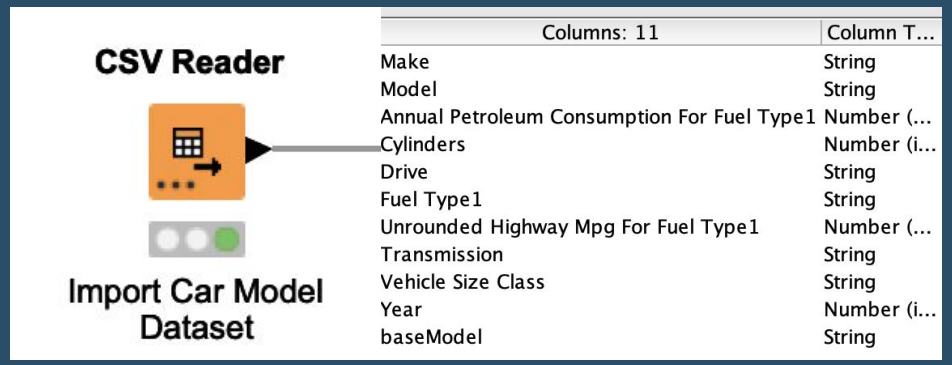


Figure 1.

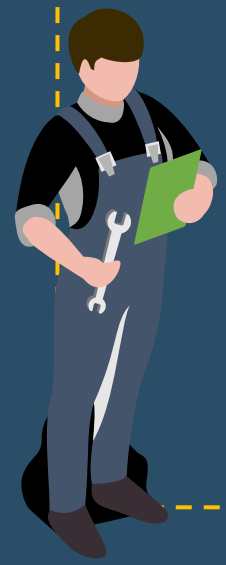
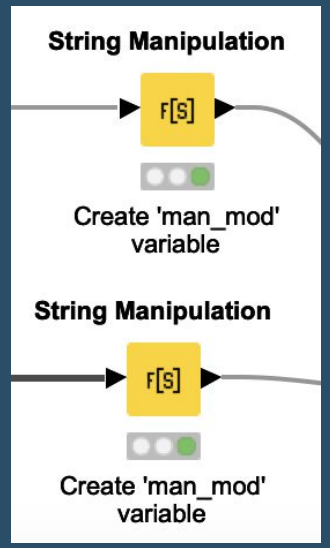


Figure 2.



The next step was to import the CMD (Figure 1). We can see that it also includes data like vehicle size, cylinders and other, which can be used later for imputation. We then prepare the inputs for the Sentence Transformer model. They are joined strings of manufacturer and car model (Figure 2). This was done to correctly assign model names like 300z – which can be difficult without knowing the brand. Creating variables called 'man_mod'.

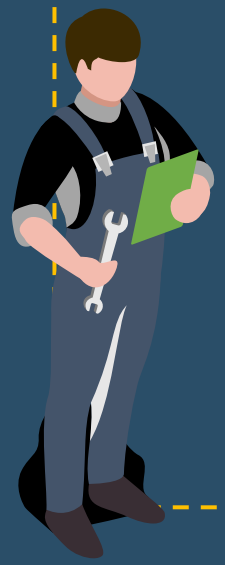
Finally, we continue with encoding the unique 'man_mod's from our dataset in the model, and 'man_mod's from CMD (thus, true actual combinations of manufacturer and model). This is followed by calculating the dot product between them resulting in a matrix with cosine similarities between all the different car models (Figure 3).

Car Model Mapping

Finally, we use a python script to apply the Sentence Transformer model.

We begin with encoding the unique 'man_mod's from our dataset in the model, and 'man_mod's from CMD (thus, true actual combinations of manufacturer and model). This is followed by calculating the dot product between them resulting in a matrix with cosine similarities between all the different car models.

For the last steps, we identify the manufacturer-model combination with the biggest cosine similarity.



```python
import knime.scripting.io as knio
import numpy as np
import pandas as pd

from nltk.corpus import stopwords
import nltk
nltk.download('vader_lexicon')
from nltk.sentiment.vader import SentimentIntensityAnalyzer
from sentence_transformers import SentenceTransformer
model = SentenceTransformer('all-MiniLM-L12-v2')

x = knio.input_tables[0].to_pandas()
y = knio.input_tables[1].to_pandas()
y = y.reset_index(drop=True)

querry_x = x["man_mod_x"]
answer = model.encode(y["man_mod_y"])
#new chat addition
querry_x_reset = querry_x.reset_index(drop=True)
querry = model.encode(querry_x_reset)

matrix = np.dot(querry, answer.T)
pandas_matrix = pd.DataFrame(matrix)
predicted_indices = pandas_matrix.idxmax(axis=1)
predicted_models = y.loc[predicted_indices, 'man_mod_y']
x['predicted_model'] = predicted_models.values
x[['predicted_manufacturer', 'predicted_model']] = x['predicted_model'].str.split(';', n=1, expand=True)

knio.output_tables[0] = knio.Table.from_pandas(x)
```

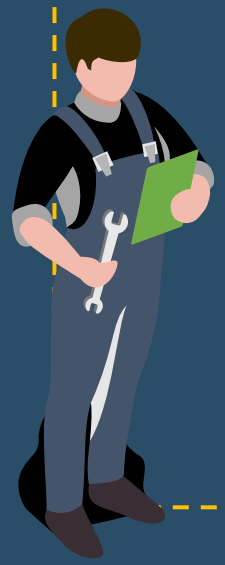
Car Model Mapping

After the splitting the string into predicted manufacturer and predicted model, we obtain the dataframe showing the results of our model prediction (Figure 1). We can observe how the minor differences in wording are picked up and generalized into actual model types contained in the Car Model Dataset.

The following procedure, managed to reduce the cardinality of model variable from 30 thousand to 2658 (Figure 4), making it possible to actually use the 'model' variable in our predictive models.

manuf...	model	man_mod_x	predicted_...	predic...
gmc	sierra 1500 crew cab slt	gmc;sierra 1500 crew cab slt	Sierra 1500 AWD	GMC
chevrolet	silverado 1500	chevrolet;silverado 1500	Silverado 1500 ...	Chevrolet
chevrolet	silverado 1500 crew	chevrolet;silverado 1500 cr...	Silverado 1500 ...	Chevrolet
toyota	tundra double cab sr	toyota;tundra double cab sr	Tundra 2WD	Toyota

Figure 1.



Car Model Identification

Next we proceed to match the car models based on the newly created mapping dataframe obtained in the previous slide.

We use the joiner node (Figure 1) to match on the ‘model’ variable, and when a match happens we add the ‘predicted_model’ and ‘predicted_manufacturer’ from the other dataframe.

Unfortunately not all car models can be identified in our original database due to the cleaning we performed in the Car Model Mapping chapter.

In the end, we drop 33411 the cars that did not map with actual car models (Figure 2), as for the purpose of our predictive model they do not carry valuable information.

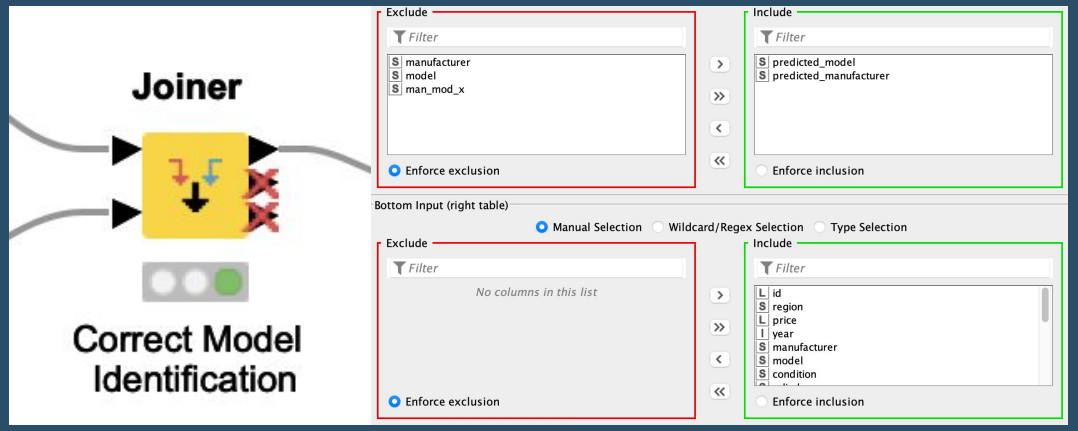
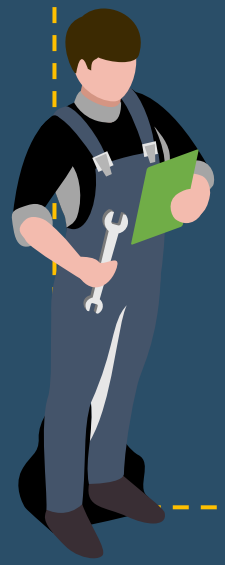


Figure 1.

```
X['actual_model'].isna().sum()
```

```
33411
```

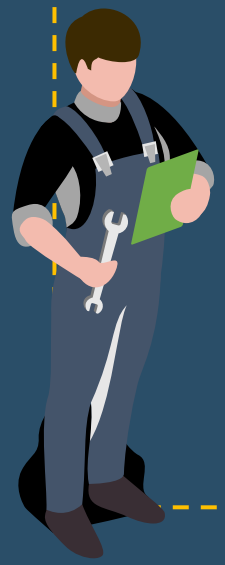
Figure 2.



Car Model Identification

Allowing ourselves to drop the old 'model' and 'manufacturer' columns, we are left with cars with correct actual model and manufacturers. The following procedure allowed for a decrease in the cardinality of model from 29667 to just 2658 models, and actually increased the cardinality of manufacturers from 42 to 50. This is because thanks to the Car Model Dataset, we could identify sub-brands of companies, such as Plymouth which is a brand developed under Chrysler.

In the general summary of the data, we can observe that we now have no missing data for model and manufacturer – bottom of the table. However, this has reduced the number of rows from the initial 426880 to 388051 – roughly 40 thousand observations. While this number is quite high at around 10% of our data, we are confident that the inclusion of additional 40K cars without manufacturers or models, would introduce more noise than information for our model's application.



Dataframe shape: (388051, 20)
Duplicates in Train Dataset: 0, (0.0)%

	dtype	missing%	unique
id	int64	0.000000	388051
region	object	0.000000	404
price	int64	0.000000	14978
year	float64	0.002319	92
condition	object	40.340316	6
cylinders	object	41.323692	8
fuel	object	0.643730	5
odometer	float64	1.017650	99358
title_status	object	1.972679	6
transmission	object	0.564101	3
drive	object	30.091921	3
size	object	72.238701	4
type	object	20.505552	13
paint_color	object	29.881897	12
description	object	0.000515	329224
state	object	0.000000	51
lat	float64	1.503153	47456
long	float64	1.503153	47979
actual_model	object	0.000000	2658
act_manufacturer	object	0.000000	50

Imputation

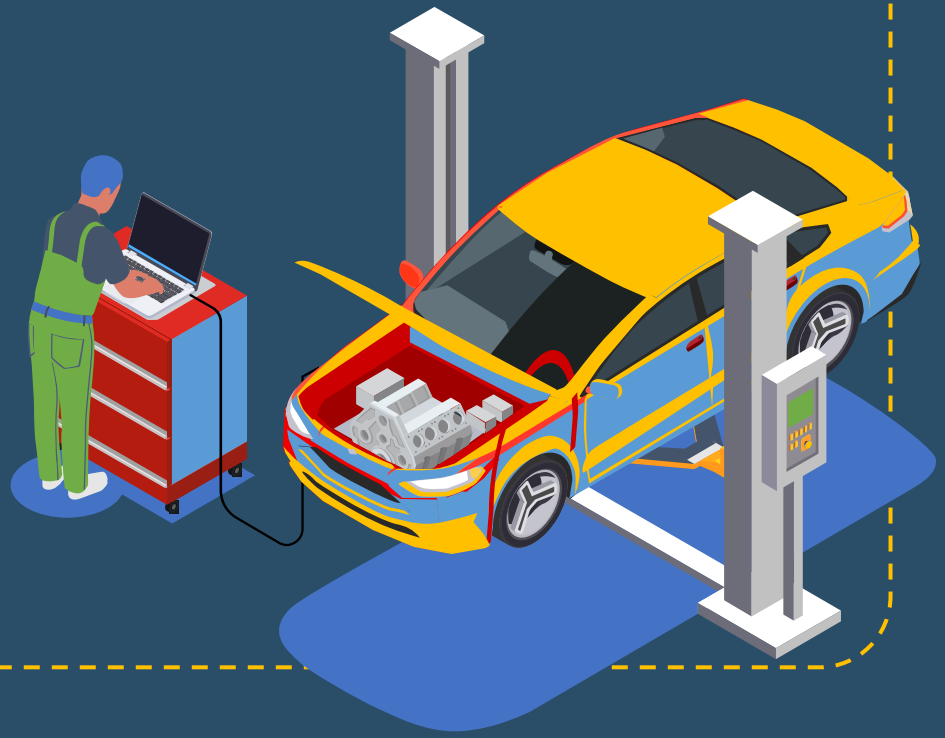
As presented on the previous slide there is a lot of missing values in our data to be addressed. We will split the imputation process into two parts:

Car Model Dataset Imputation

General Imputation

The CMD Imputation will leverage the additionally available information in our secondary dataset. Since we now have all the cars correctly labeled with the actual models, we will impute the actual model's specification whenever possible.

The General Imputation part will cover more usual imputing practices dealing with all variables that we were unable to fill with model's specifications.



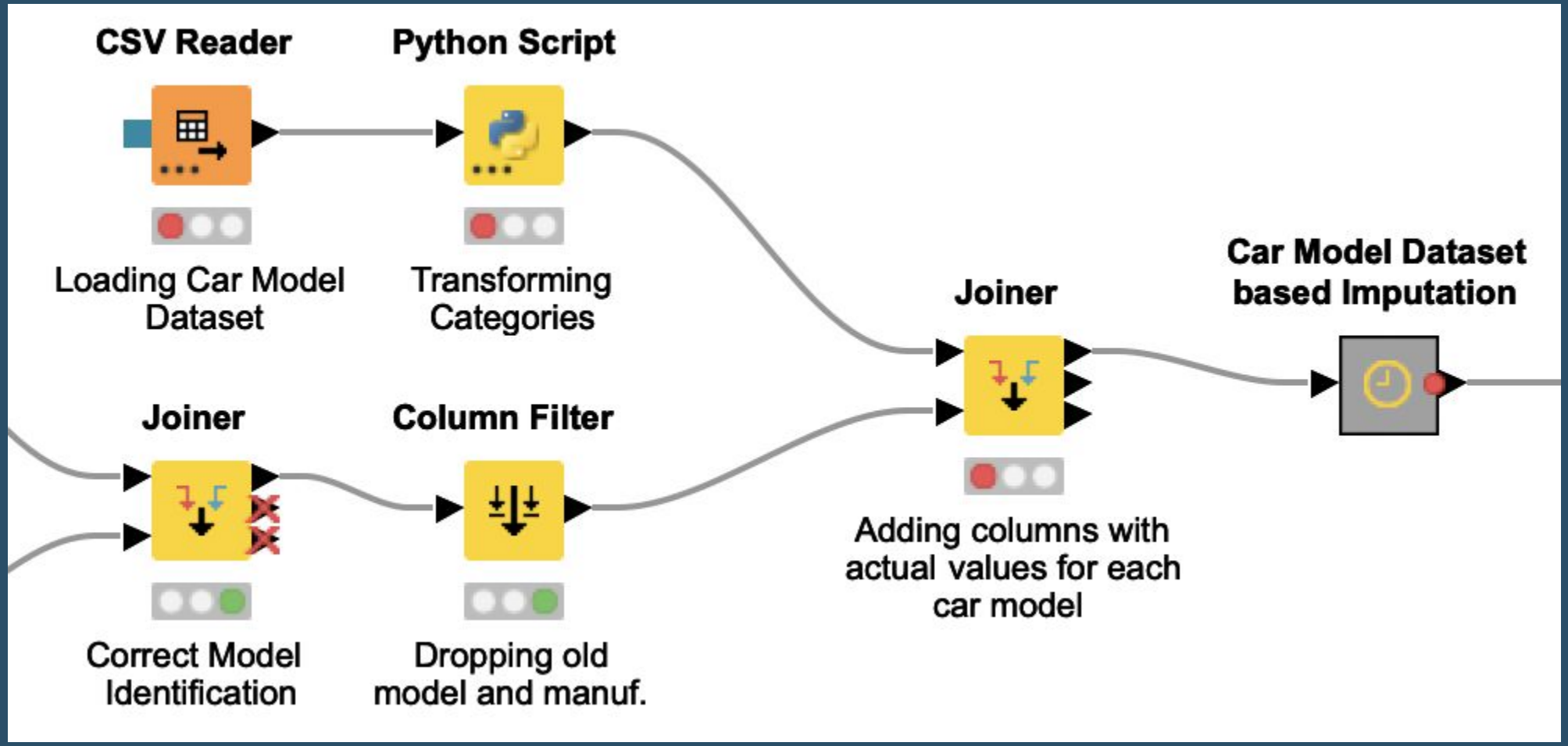
Car Model Dataset Imputation

Preparing the Car Model Dataset to include our main dataset categories

Creating dictionaries for mapping the car models to their specifications

Applying the mapping to input all the missing data

Car Model Dataset Imputation



Car Model Dataset Imputation

We begin by preparing the CMD so that the categories are matched with our original dataset. For that we have used a python script, to map using dictionaries the CMD categories to the our dataset categories and decrease the cardinality in the Car Model Dataset. We can see that our dataset is way less precise with just 4 different categories for the drive variable, while the CMD has 8 unique values (Figure 1).

This is applied to 5 variables that can be identified using the CMD: ‘drive’, ‘fuel’, ‘transmission’, ‘size’, ‘type’ and ‘cylinders’. With the only exception of ‘size’ and ‘type’ in which we combine the variables into just ‘type’, as they carried the same information. Thus, we created 5 new columns (Figure 2).

```
car_models_final = knio.input_tables[0].to_pandas()

#here we transform all of the categories to match the categories of the original dataset
drive_mapping = {
    'Front-Wheel Drive': 'fwd',
    'All-Wheel Drive': '4wd',
    'Rear-Wheel Drive': 'rwd',
    '4-Wheel Drive': '4wd',
    'Part-time 4-Wheel Drive': '4wd',
    '4-Wheel or All-Wheel Drive': '4wd',
    '2-Wheel Drive': 'rwd'
}

car_models_final['TrueDrive'] = car_models_final['Drive'].map(drive_mapping)
car_models_final = car_models_final.drop(['Drive'],axis=1)
```

Figure 1.

TrueD...	TrueF...	TrueType	TrueT...	TrueC...
fwd	gas	sub-compact	manual	4 cylinders
4wd	gas	compact	automatic	6 cylinders
rwd	gas	compact	manual	8 cylinders
4wd	gas	SUV	automatic	6 cylinders
4wd	gas	SUV	automatic	6 cylinders

Figure 2.

Car Model Dataset Imputation

The next step was to join the CMD and our dataset based on matching ‘model’ again (now with already reduced dimensionality in our dataset) and adding the 5 ‘true’ value columns from previous slide. However, we didn’t want to just disregard the previous data and map to the values from CMD, but only impute when the values are missing.

Therefore, we enter the metanode ‘Car Model Dataset based Imputation’, which performs the following process (Figure 2). We begin by using a Rule Engine to create a new column when based on the ‘true’ value columns, only when a data point is missing. Then we use column merger to merge the newly created column and the original as they are exactly complementary. Finally, as the joiner node can only join 2 tables, we wrap it all back together using several of them.

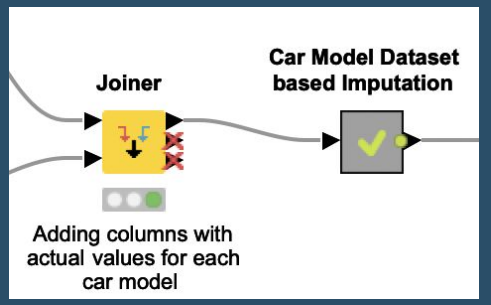


Figure 1.

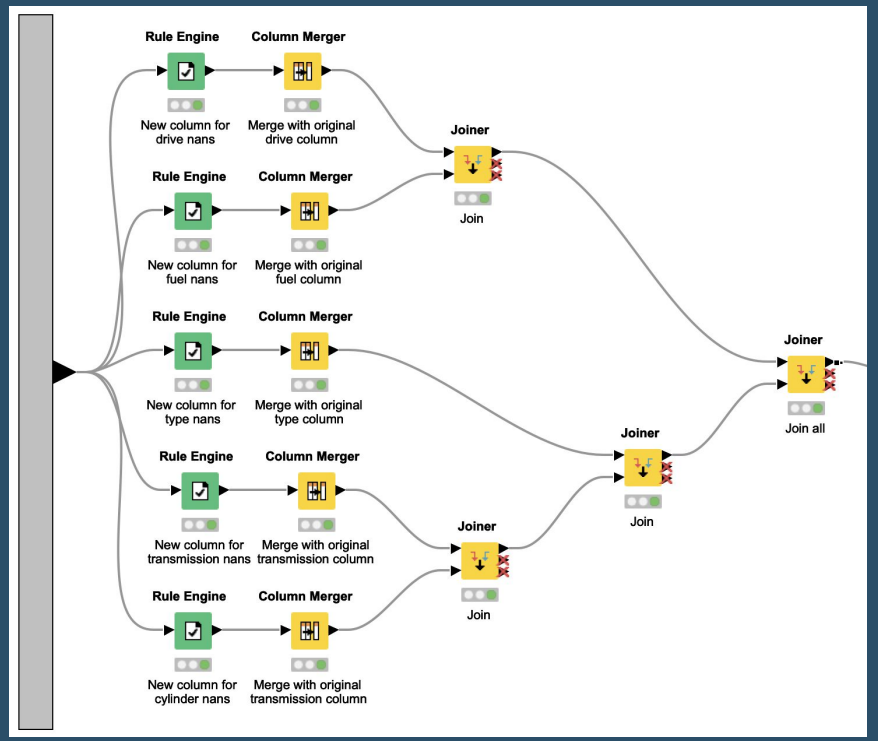


Figure 2.

Car Model Dataset Imputation

While almost all of the missing values from ‘drive’, ‘fuel’, ‘transmission’, ‘type’ and ‘cylinders’ were imputed we still missed a couple, as even the CMD was not ideally clean. Frankly: 1.06% of cylinder, 0.15% of drive, and 0.73% of type (Figure 1).

We decided that even though we could likely perform imputation with median values, given the small amount of cars that require imputation for these variables, we will get rid of these cars completely (Figure 2). By doing so we can guaranteed the following - all cars have corect models and their model specifications. We believe this ensured that the difference between having 4 and 6 cylinders in a car, will be spotted easier by our model. For the rest of the variables we proceeded with regular imputation practices.

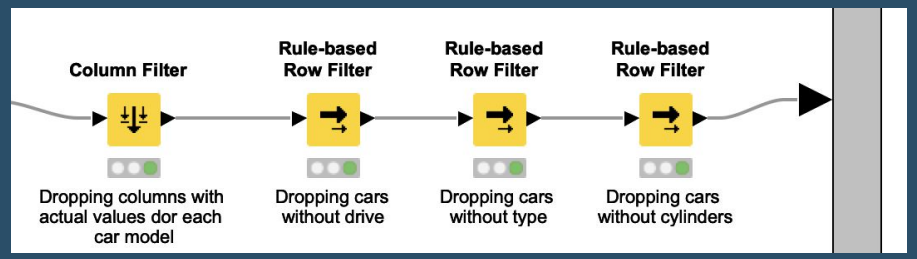


Figure 2.

Dataframe shape: (388051, 19)
Duplicates in Train Dataset: 0, (0.0)%

	dtype	missing%	unique
id	int64	0.000000	388051
region	object	0.000000	404
price	int64	0.000000	14978
year	float64	0.002319	92
condition	object	40.340316	6
cylinders	object	1.063778	8
fuel	object	0.000000	5
odometer	float64	1.017650	99358
title_status	object	1.972679	6
transmission	object	0.000000	3
drive	object	0.152815	3
type	object	0.731089	17
paint_color	object	29.881897	12
description	object	0.000515	329224
state	object	0.000000	51
lat	float64	1.503153	47456
long	float64	1.503153	47979
actual_model	object	0.000000	2658
act_manufacturer	object	0.000000	50

Figure 1.

General Imputation

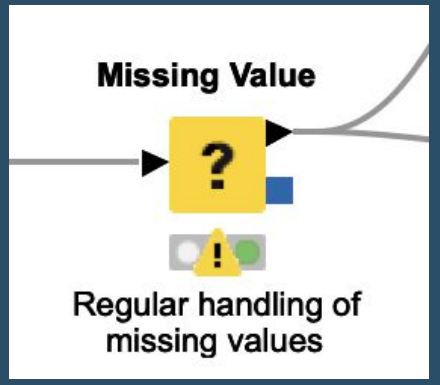


Figure 1.

Now we are left with the following columns with missing data: ‘year’, ‘condition’, ‘odometer’, ‘title_status’, ‘paint_color’ and ‘lat’/’long’. We deal with them using the Missing Value node.

We will first deal with the variables ‘year’ and ‘odometer’, as they are the most reliable data points to imput. For the ‘year’ there is less than a 0.01% of missing values, while for ‘odometer’ there is only 1.01% (previous slide). Given their distributions described in the Data Description chapter, we decided to fill both with the median values for their respective variables (Figure 1).

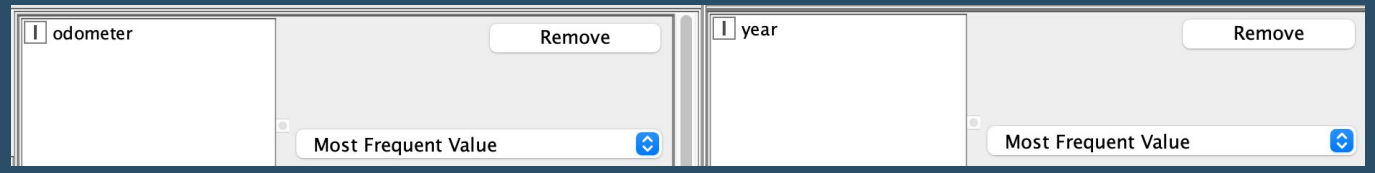


Figure 2.

General Imputation

Condition is an important factor of a car's value, therefore, we decided not to drop it. Given that 40% of condition is missing, we need to be careful with the imputation technique. We can't really do median imputation, due to the over-representation of positive condition in the dataset (Figure 1).

We suspect that car owners with questionable condition may be hesitant to put the condition in the car posting. We will create another category - unknown - which we believe might capture that phenomena (Figure 2).

For the 'description' variable, which we will use for an NLP application, we just decided to impute an empty string, signaling neutral sentiment (Figure 3).

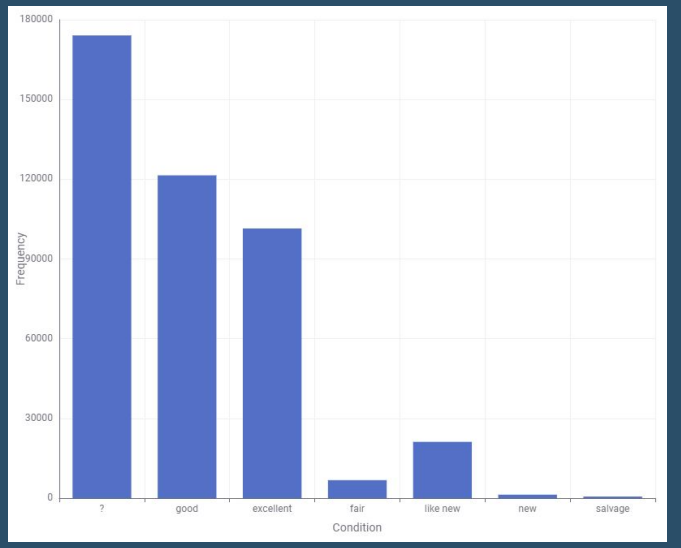


Figure 1.

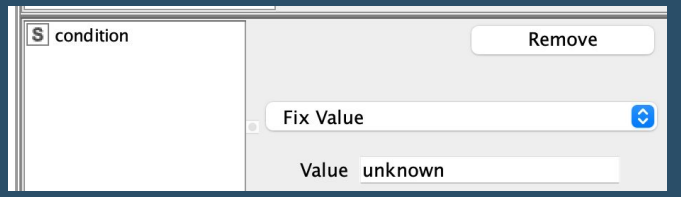


Figure 2.

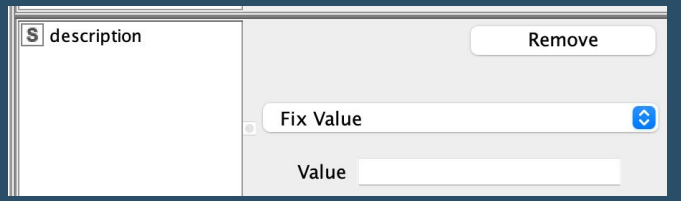
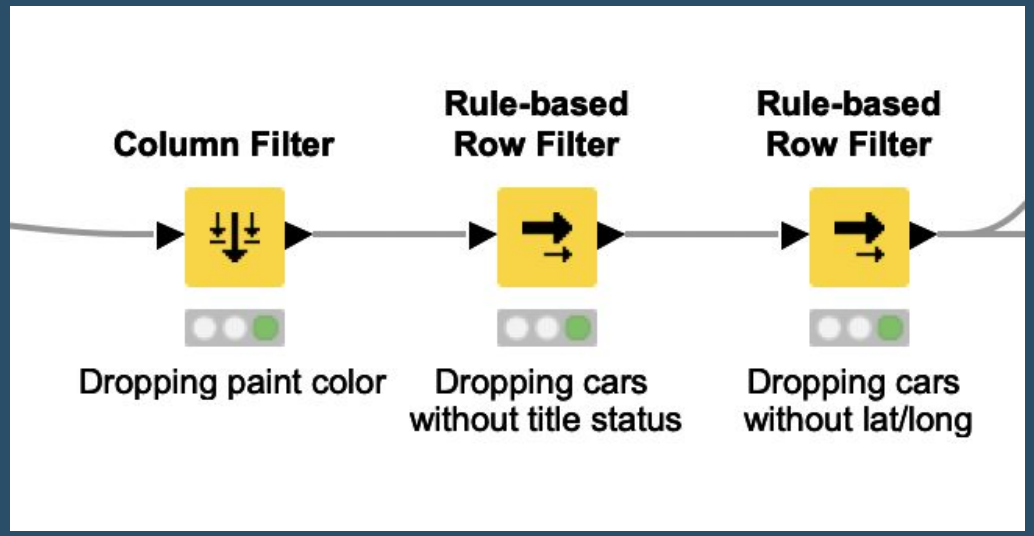


Figure 3.

General Imputation

Even though some people believe painting their car red will make it faster, the amount of missing data (30%) is to big to base imputing on anything. While at first we wanted to include ‘paint_color’ in the model, we decided to get rid the variable and drop the column.



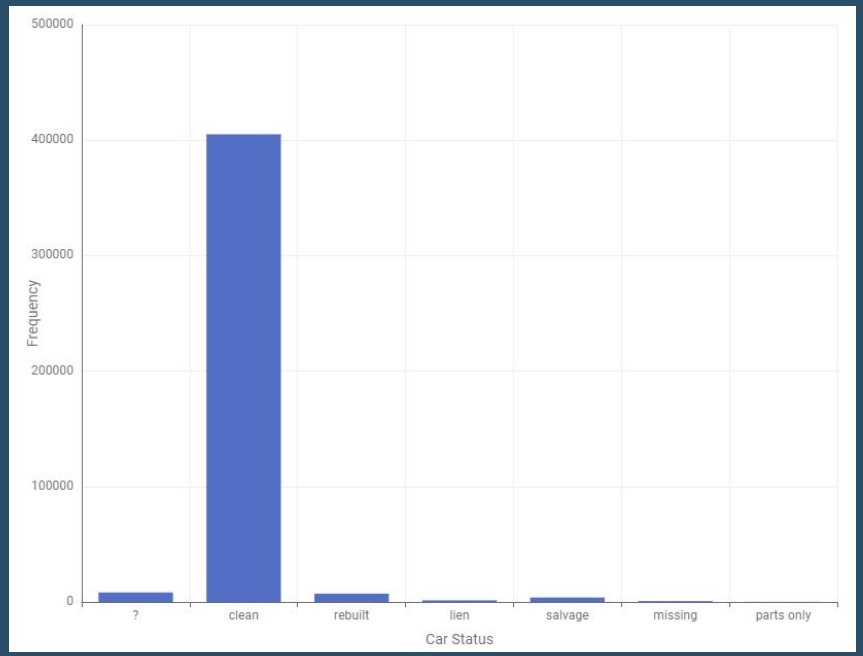
For ‘lat’ and ‘long’ we identified that when one is missing the other is as well.

We wanted to use the location of the sale along with KNN to found out regional differences between car prices. With 1.5% of missing location data, we believed that with slight differences between the regional prices, random imputation of the location of sale could already influence the significance of regional differences. Once again we leverage the fact of having abundant data points in our dataset, and drop the cars without positional data.

General Imputation

The last variable ‘title_status’, has a very unique distribution. We were hesitant to drop the variable completely but we decided it still actually carries important information. Frankly, ‘clean’ is dominating, however, categories ‘salvage’, ‘part only’ or ‘rebuilt’ can effectively signal heavily underpriced cars, which would be hard to identify otherwise. Most people trying to get rid of a salvaged car are not trying to get the market price of said model.

This point led us to the decision of actually dropping the cars without the ‘title status’ (previous slide), as there is only a couple thousand of them. If we tried to impute ‘clean’ for all the missing ones, we are likely to impute it for some cars that are salvaged. This would significantly decrease the information of damaged cars, given the small amount of them in the first place.



Imputation Summary

To summarize, we have now dealt with all the missing values. We are aware that a lot of observations were dropped from out dataset, however, as we argued in this chapter, we believe that imputing these values would bring more noise than information to our dataset.

We now stand at 367267 observations and 18 variables, reducing the number of observations by about 15%, from the initial point. However, we lean on the specific application to our managerial application. We want to build as accurate of a representation of used cars market as possible. While this procedure could not be applied straight to unseen data, the model is meant only for the prediction task by car dealers.

	dtype	missing%	unique
id	int64	0.0	367267
region	object	0.0	404
price	int64	0.0	14311
year	float64	0.0	91
condition	object	0.0	7
cylinders	object	0.0	8
fuel	object	0.0	5
odometer	float64	0.0	96267
title_status	object	0.0	6
transmission	object	0.0	3
drive	object	0.0	3
type	object	0.0	17
description	object	0.0	310816
state	object	0.0	51
lat	float64	0.0	46874
long	float64	0.0	47407
actual_model	object	0.0	2618
act_manufacturer	object	0.0	49

Outlier Analysis

	Price
318592	3736928711
356716	3736928711
91576	3024942282
257840	3024942282
184704	1410065407
...	
151878	0
313692	0
313691	0
13979	0
241913	0

Figure 1.

Some quick analysis shows that there are listing for over 3,000,000,000 dollars which are obviously wrong (Figure 1), as this is Craigslist. They are mostly likely phone numbers, so they are not reliable data points.

As for the cars with 0 price, the prices are probably just ads with “contact me for price” so we will have to delete these observations as well, because they are not actual pricings affecting our model’s prediction power.

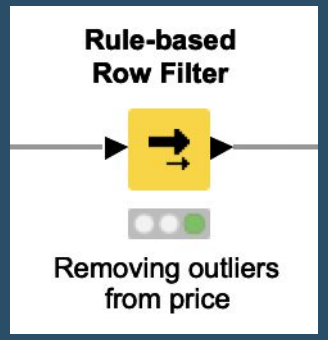


Figure 2.

We decided to draw our boundary for outliers at 200,000 not only to eliminate extremes but because our aim is to model typical used cars not the luxury market. We are also removing all zero price cars, reducing the datapoint number by 25962 (Figure 3). The new total is 341305 cars.

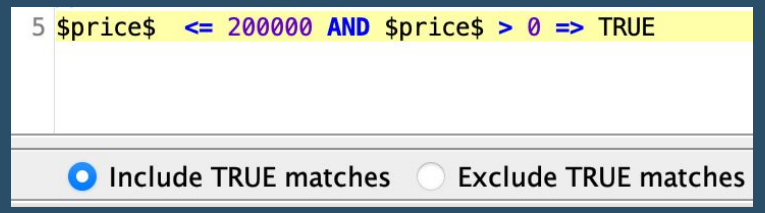
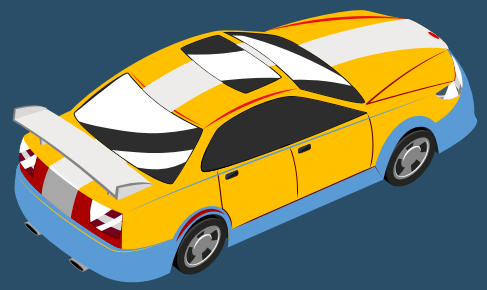
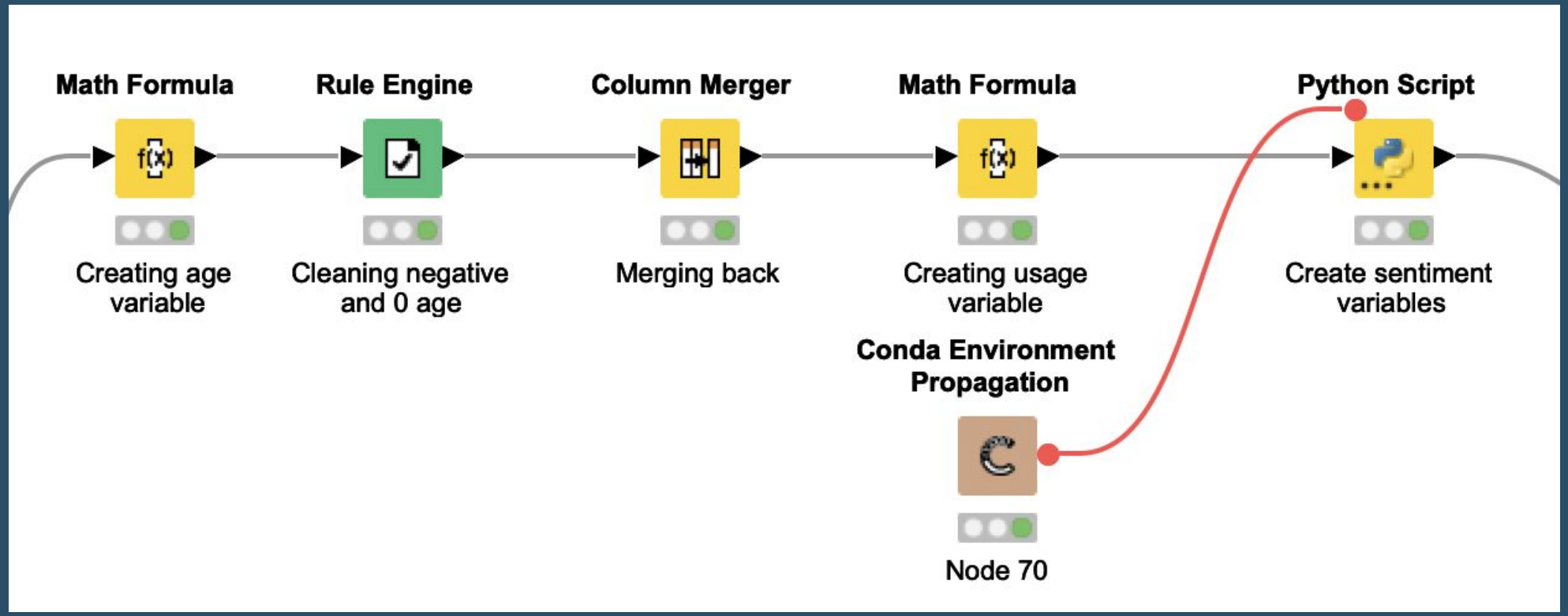


Figure 3.



Feature Engineering



Feature Engineering

Age of a car

We have added a feature called ‘age’ which transforms the variable ‘year’. While from the modelling standpoint it does not change much, it is more understandable to the end user. As the cars were posted in 2021 we choose it as the base year. All the cars from 2021 and 2022 (some are called 2022 models due to how cars are released) are assigned the minimum age of 1. We transform the variable below Math Formula and Rule Engine nodes (Figure 1), and then use Column Merger node to combine the raw and corrected calculations.

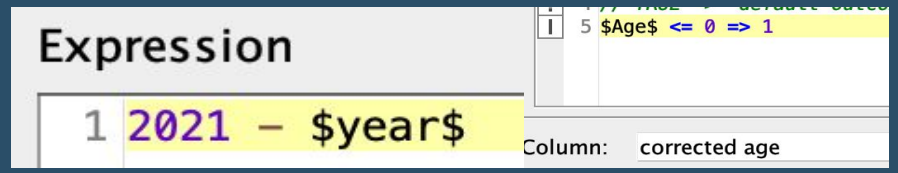


Figure 1.

Usage per year

We also included a new variable called ‘usage’, which utilizes the ‘age’ variable. It is calculated as the total mileage (‘odometer’) divided by the car age. We expect that some cars are used more heavily than others and this variable allows us to capture this effect. A fairly new car but used a lot is not priced the same, as one just sitting in the garage. We calculate it below using a single Math Formula node (Figure 2).

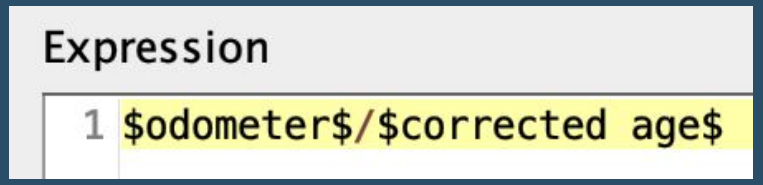
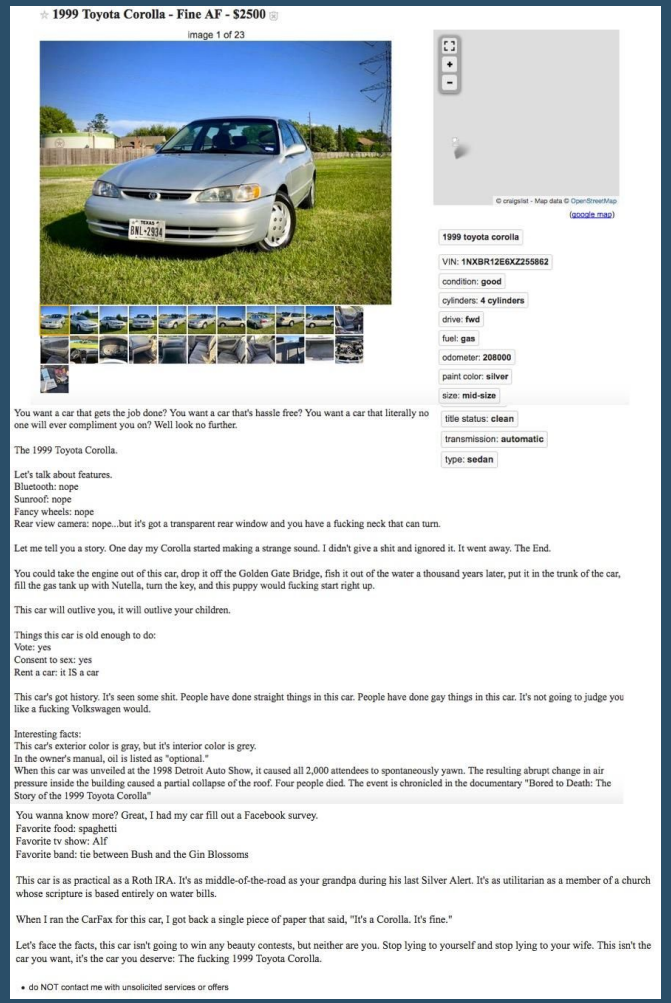


Figure 2.

Feature Engineering - Sentiment

The description in car listings is a goldmine of information that can provide some key insight on a car's value. The following is a real description from a Craigslist car ad:

"Had been driven daily until recently when the **brakes failed**. Selling as is. Good motor. Pretty new transmission. Anyone handy enough to do the work or willing to pay for repair will get a truck with some miles left in it. **Not safe to drive** until the brakes get fixed. Must have a way to transport the vehicle."



☆ 1999 Toyota Corolla - Fine AF - $2500

image 1 of 23

1999 toyota corolla

VIN: 1NXBR12E6XZ255862
condition: good
cylinders: 4 cylinders
drive: fwd
fuel: gas
odometer: 208000
paint color: silver
size: mid-size
title status: clean
transmission: automatic
type: sedan

You want a car that gets the job done? You want a car that's hassle free? You want a car that literally no one will ever compliment you on? Well look no further.

The 1999 Toyota Corolla.

Let's talk about features.
Bluetooth: nope
Sunroof: nope
Fancy wheels: nope
Rear view camera: nope...but it's got a transparent rear window and you have a fucking neck that can turn.

Let me tell you a story. One day my Corolla started making a strange sound. I didn't give a shit and ignored it. It went away. The End.

You could take the engine out of this car, drop it off the Golden Gate Bridge, fish it out of the water a thousand years later, put it in the trunk of the car, fill the gas tank up with Nutella, turn the key, and this puppy would fucking start right up.

This car will outlive you, it will outlive your children.

Things this car is old enough to do:
Vote: yes
Consent to sex: yes
Rent a car: it IS a car

This car's got history. It's seen some shit. People have done straight things in this car. People have done gay things in this car. It's not going to judge you like a fucking Volkswagen would.

Interesting facts:
This car's exterior color is gray, but it's interior color is grey.
In the owner's manual, oil is listed as "optional."
When this car was unveiled at the 1998 Detroit Auto Show, it caused all 2,000 attendees to spontaneously yawn. The resulting abrupt change in air pressure inside the building caused a partial collapse of the roof. Four people died. The event is chronicled in the documentary "Bored to Death: The Story of the 1999 Toyota Corolla"

You wanna know more? Great, I had my car fill out a Facebook survey.
Favorite food: spaghetti
Favorite tv show: Alf
Favorite band: tie between Bush and the Gin Blossoms

This car is as practical as a Roth IRA. It's as middle-of-the-road as your grandpa during his last Silver Alert. It's as utilitarian as a member of a church whose scripture is based entirely on water bills.

When I ran the CarFax for this car, I got back a single piece of paper that said, "It's a Corolla. It's fine."

Let's face the facts, this car isn't going to win any beauty contests, but neither are you. Stop lying to yourself and stop lying to your wife. This isn't the car you want, it's the car you deserve: The fucking 1999 Toyota Corolla.

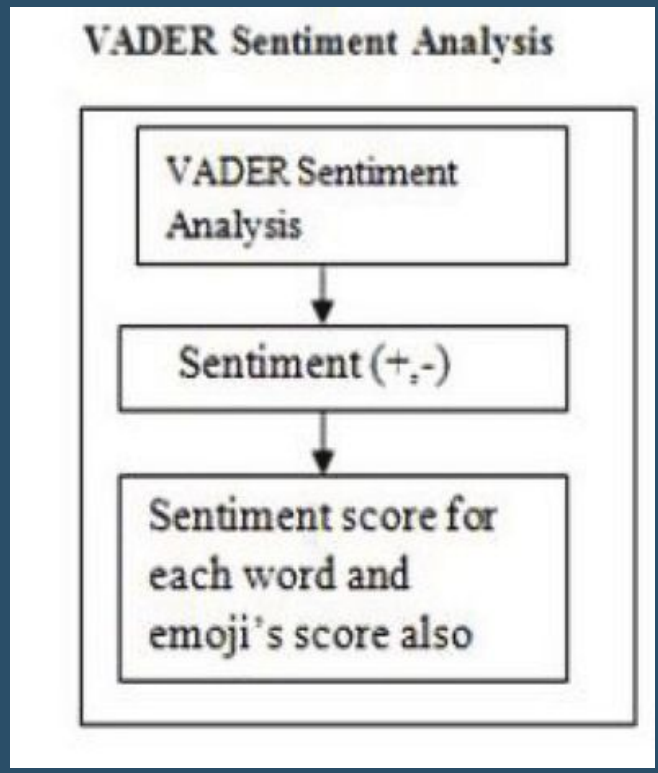
- do NOT contact me with unsolicited services or offers

Another example of a detailed description but with a much more positive sentiment about their vehicle for sale. We aim to capture this value.

Feature Engineering - Sentiment

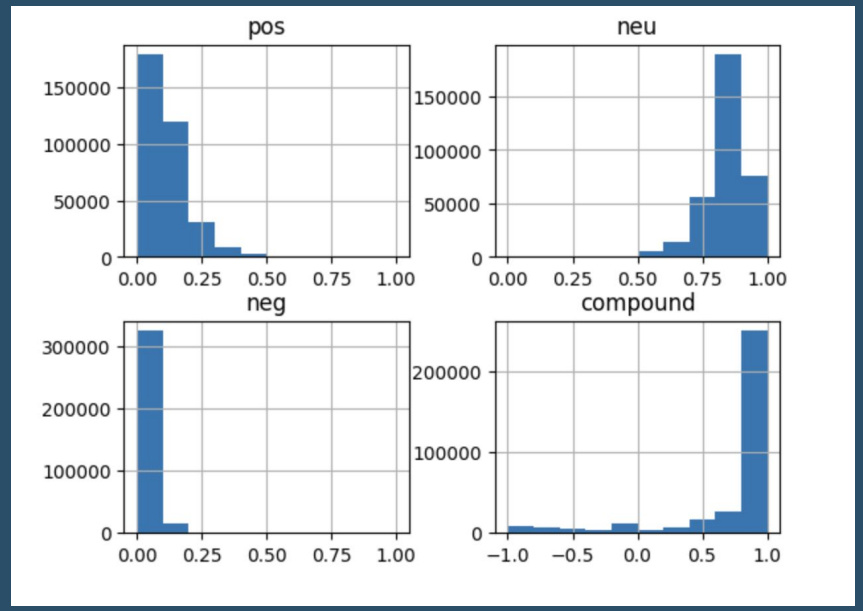
Text data needs to be transformed in order to contribute to the model, so we chose to use Vader Sentiment Analysis from Github as a tool to assess our car descriptions.

VADER (Valence Aware Dictionary and sEntiment Reasoner) is a lexicon and rule-based sentiment analysis extension that is specifically attuned to sentiments expressed in social media, and works well on texts from other domains.



Feature Engineering - Sentiment

We used the following python script to obtain the four unique sentiment variables.




```python
import knime.scripting.io as knio
import pandas as pd
import numpy as np

from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
data = knio.input_tables[0].to_pandas()


from alive_progress import alive_bar
import time
analyzer = SentimentIntensityAnalyzer()
i=-1
pd.options.mode.chained_assignment = None  # default='warn'

data["neg"]=pd.Series()
data['neu']=pd.Series()
data['pos']=pd.Series()
data['compound']=pd.Series()
with alive_bar(len(data["description"]), force_tty=True) as bar:
    for sentence in data["description"]:
        i=i+1
        score = analyzer.polarity_scores(sentence)
        data["neg"][i]=score["neg"]
        data["neu"][i]=score["neu"]
        data["pos"][i]=score["pos"]
        data["compound"][i]=score["compound"]
        bar()


knio.output_tables[0] = knio.Table.from_pandas(data)
```

Feature Engineering - Sentiment

The sentiment analysis tool is set up to consider 4 different aspects to the text in the description (Figure 1).

Positive, Negative, and Neutral aim to measure sentiment of the word in context in their respective emotion. While compound is determined by the sum of the valence scores for the words in the lexicon and estimates a degree of the sentiment rather than the actual value.

	pos	neu	neg	compound
pos	1.000000	-0.922420	0.021528	0.346431
neu	-0.922420	1.000000	-0.404902	-0.190460
neg	0.021528	-0.404902	1.000000	-0.327374
compound	0.346431	-0.190460	-0.327374	1.000000

Figure 2.

- Positive
- Negative
- Neutral
- Compound

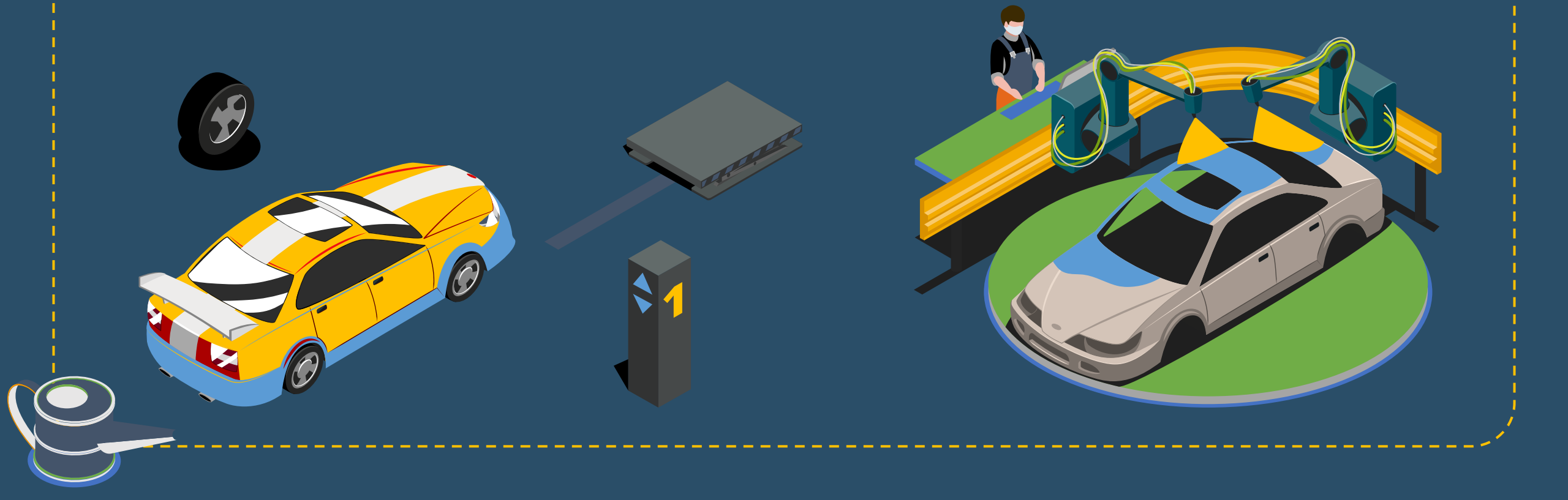
Figure 1.

Encoding

Encoding is a necessary step in using categorical data in regression. While in the initial python version of the project we have used one hot encoding for all of categorical variables, after converting to Knime, we realized that all categorical columns are automatically encoded using OH encoding in the learner nodes. Thus, we did not include encoding explicitly in our data preparation workflow, as it is handled automatically in the models workflow.

Data Descriptions

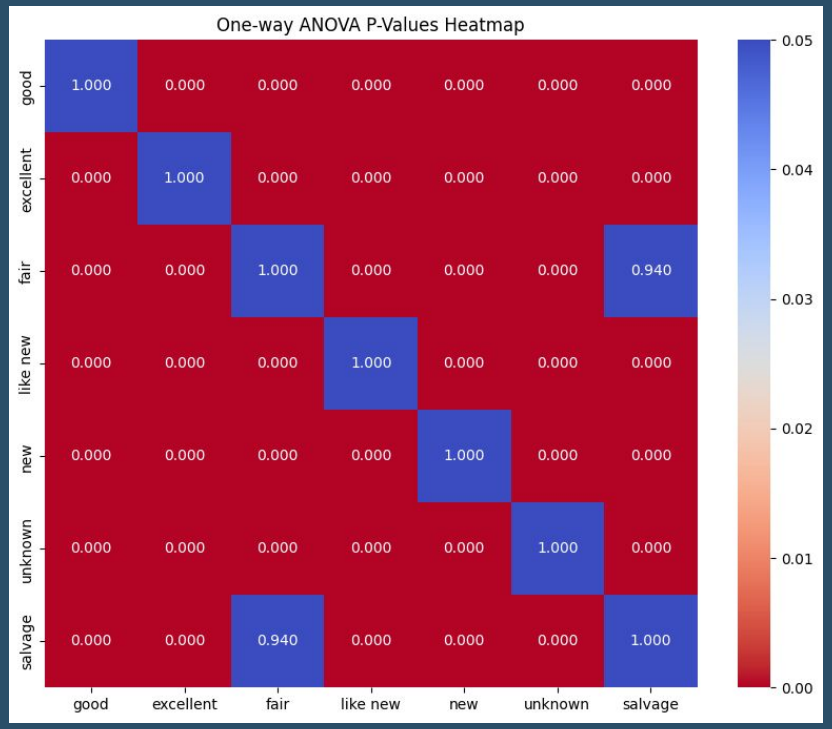
Bivariate Analysis

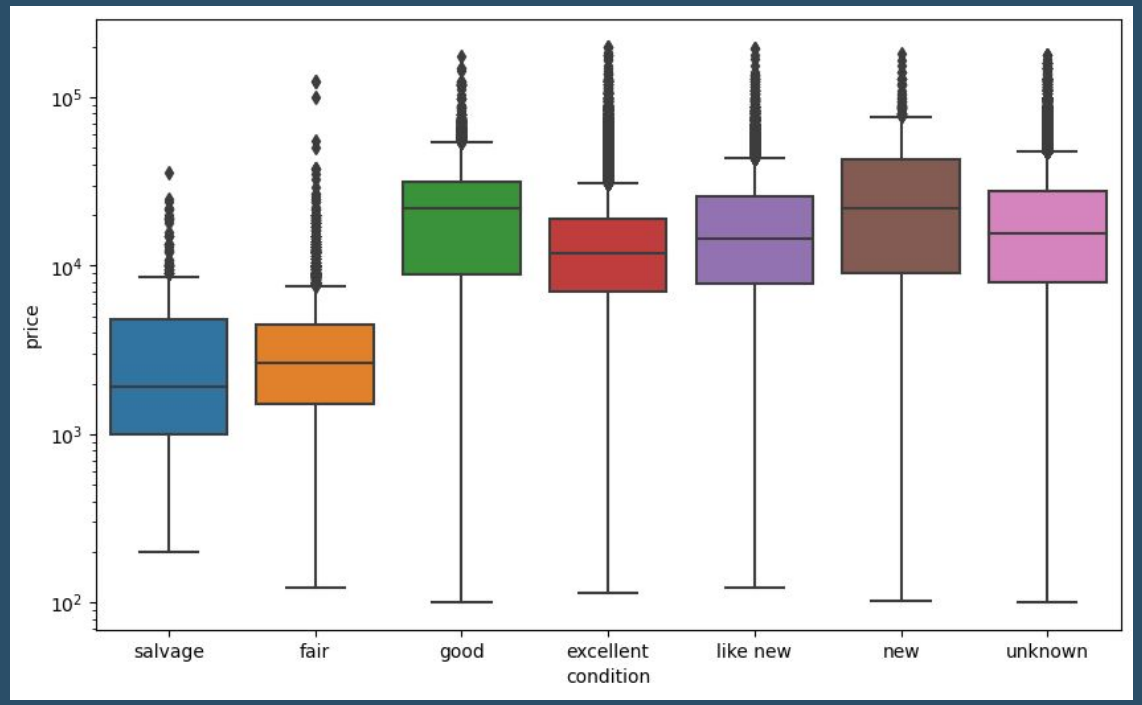


Bivariate Analysis - Condition

The results of ANOVA tests indicate that with high certainty we can assume that salvage and fair cars come from the same price distribution: people probably use “Fair” as an euphemism for a destroyed car.

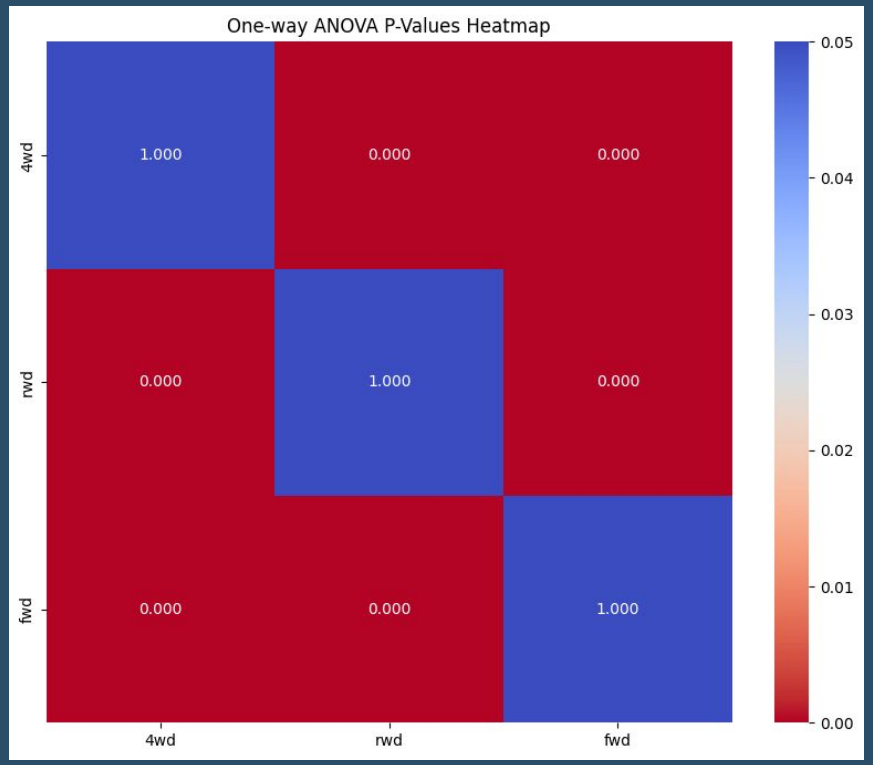
We can clearly see the difference in price distributions among condition differ significantly. We can safely distinguish between 2 groups:
-Salvage and fair (we combine it into one category)
-The rest

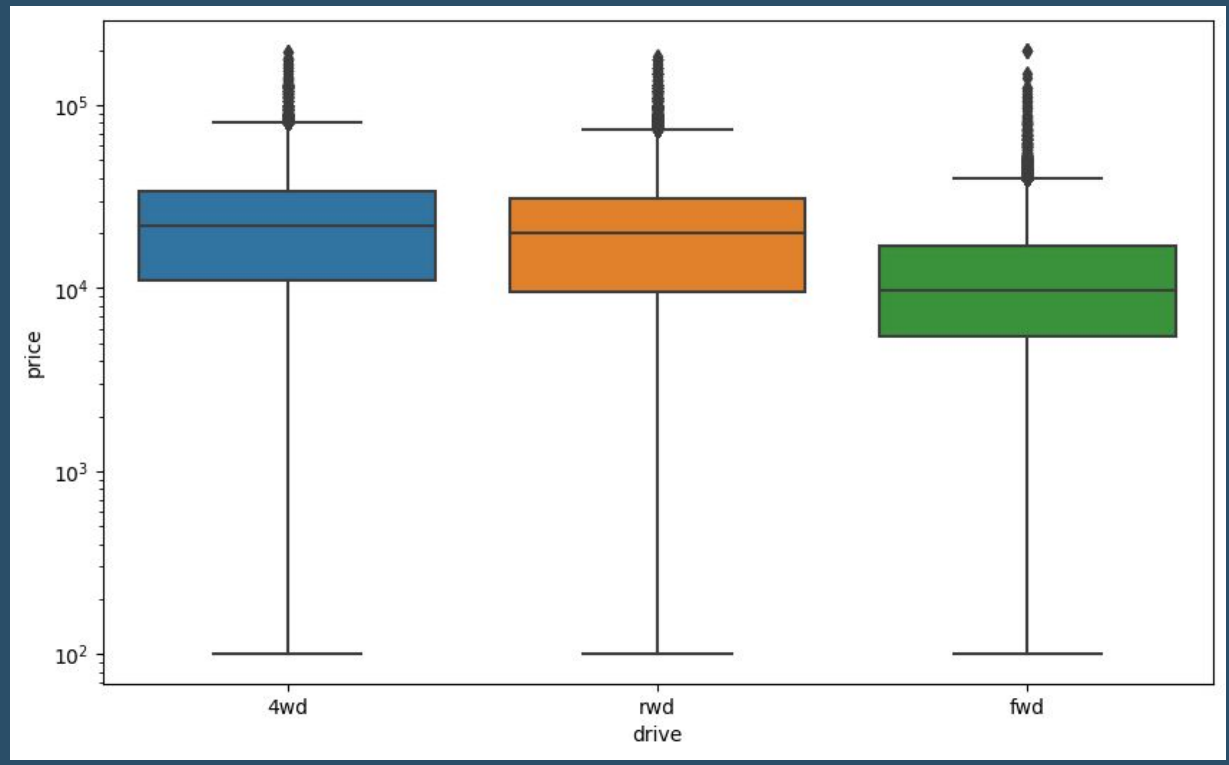




Bivariate Analysis - Drive

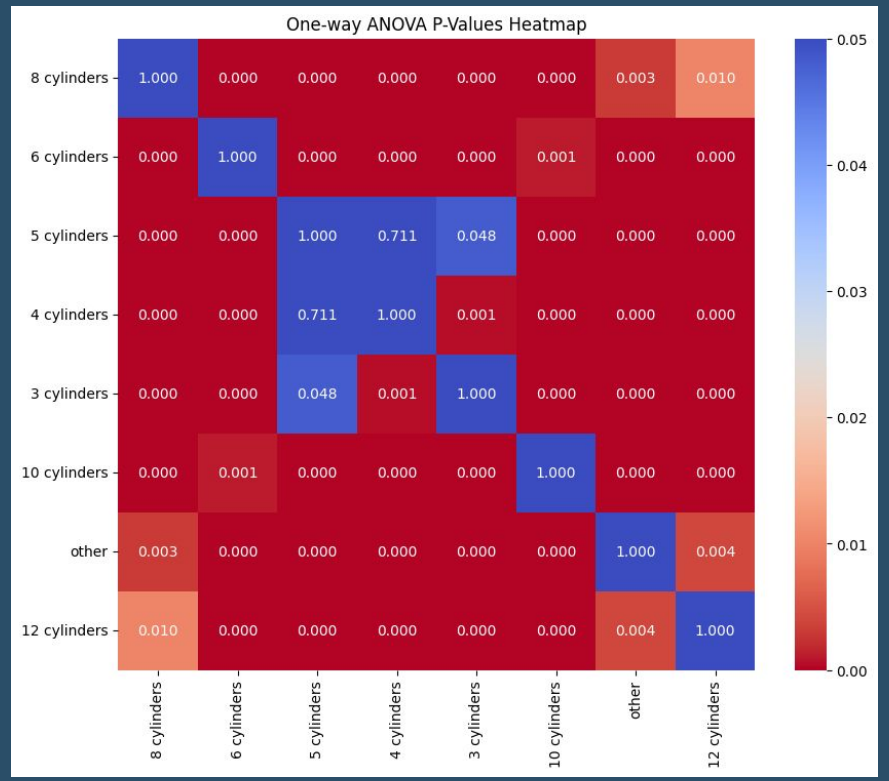
The results of ANOVA show that there are significant differences among all different type of “Drive”. We can observe that forward-wheel drive is in general priced much lower than the other 2 types.

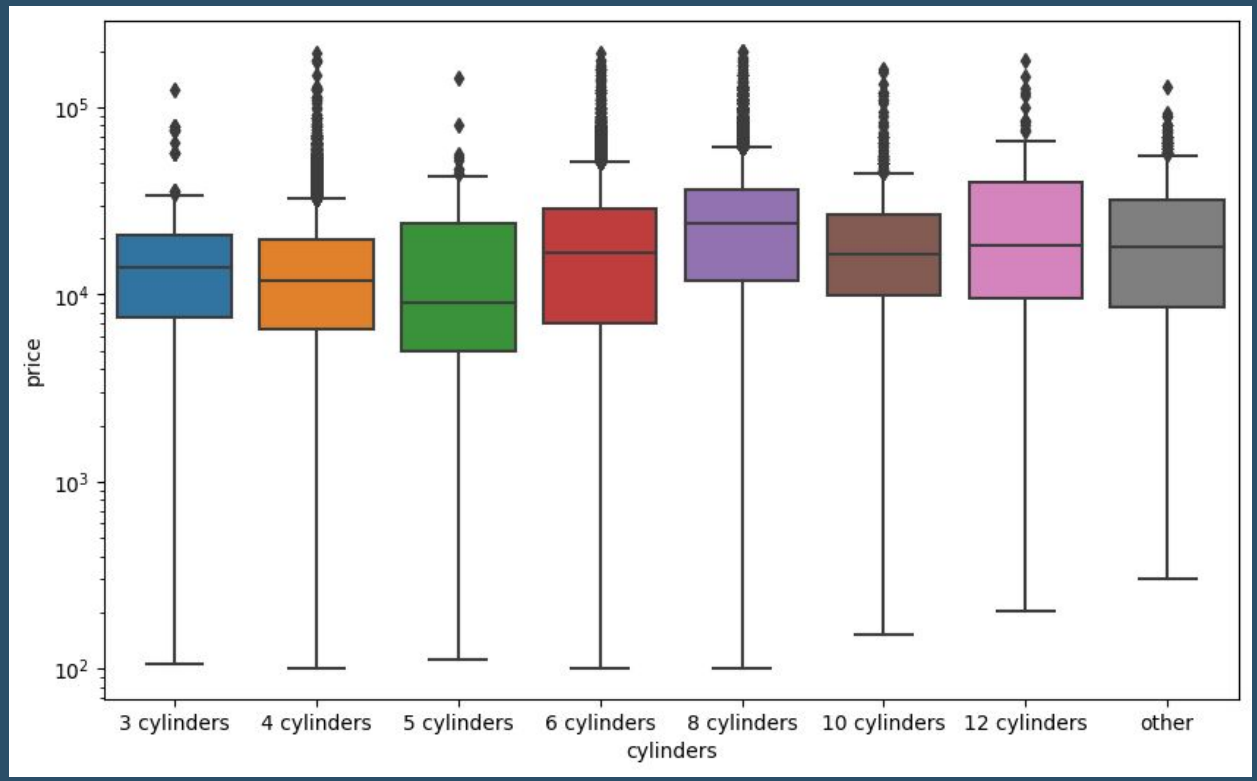




Bivariate Analysis - Cylinder

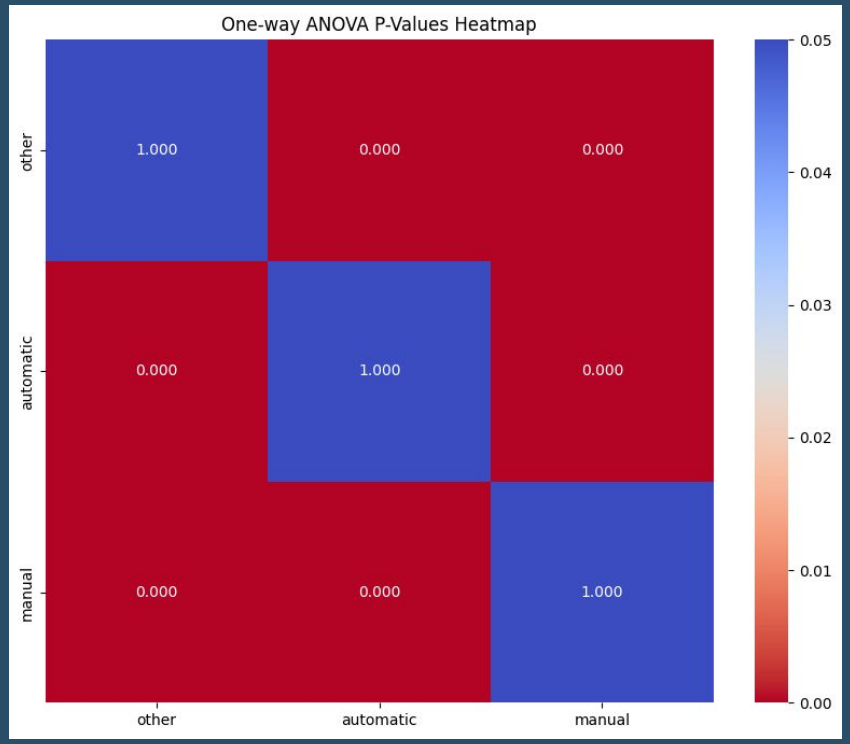
The results of ANOVA show that there are significant differences among most counts of “Cylinders”. However, the differences are not so clear between 3, 4 and 5 cylinders. We suspect that the p-values would change if more cars had 3 and 5 cylinders, as these are not the typical counts.

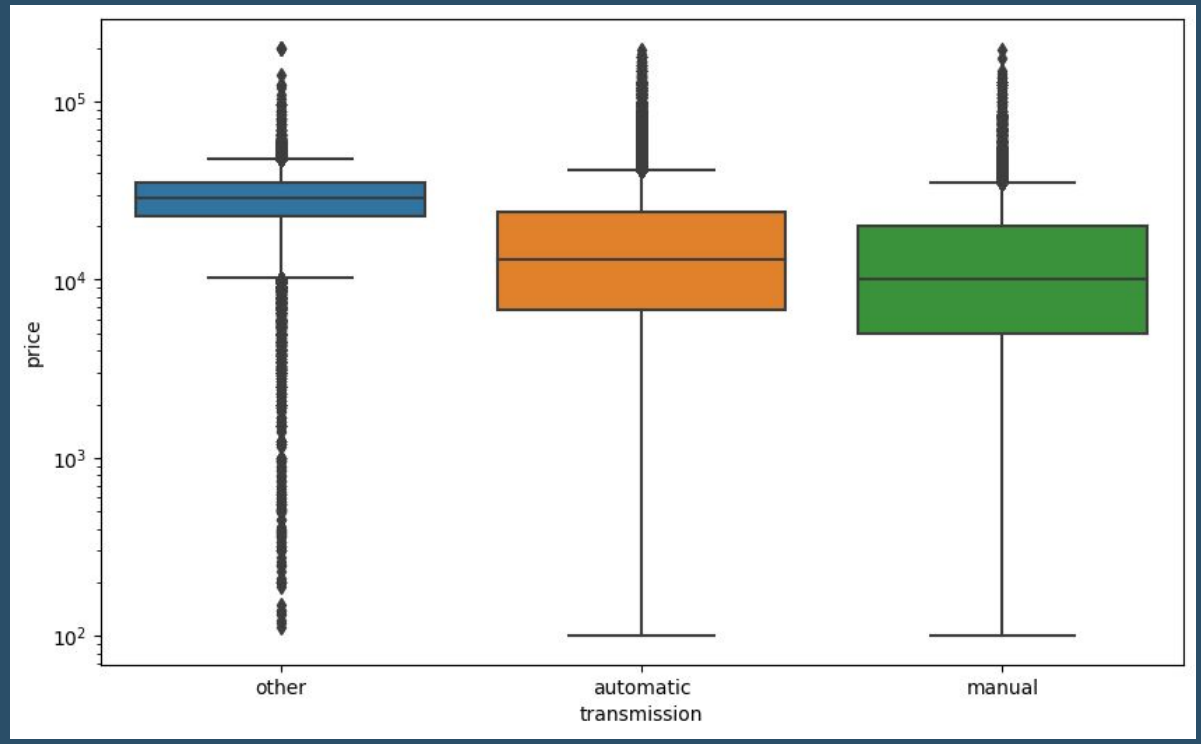




Bivariate Analysis - Transmission

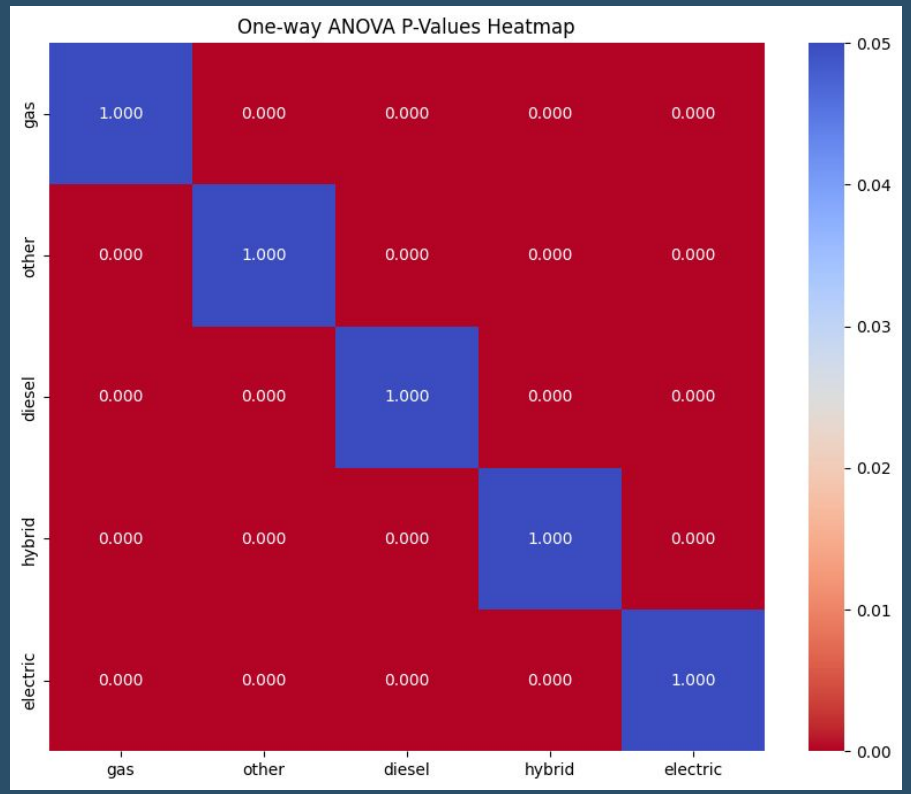
The results of ANOVA show that there are significant differences among “Transmission” types. It makes perfect sense as its common to price manual and automatic cars differently.

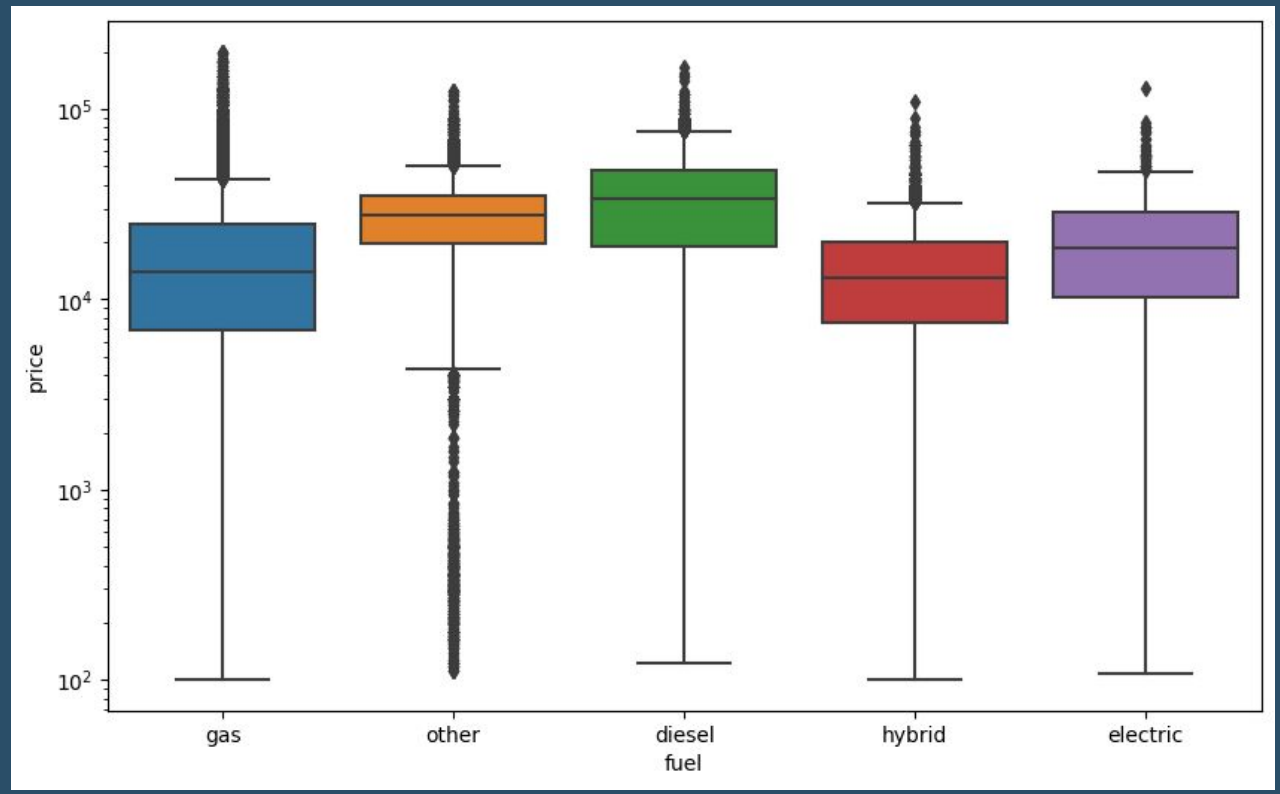




Bivariate Analysis - Fuel

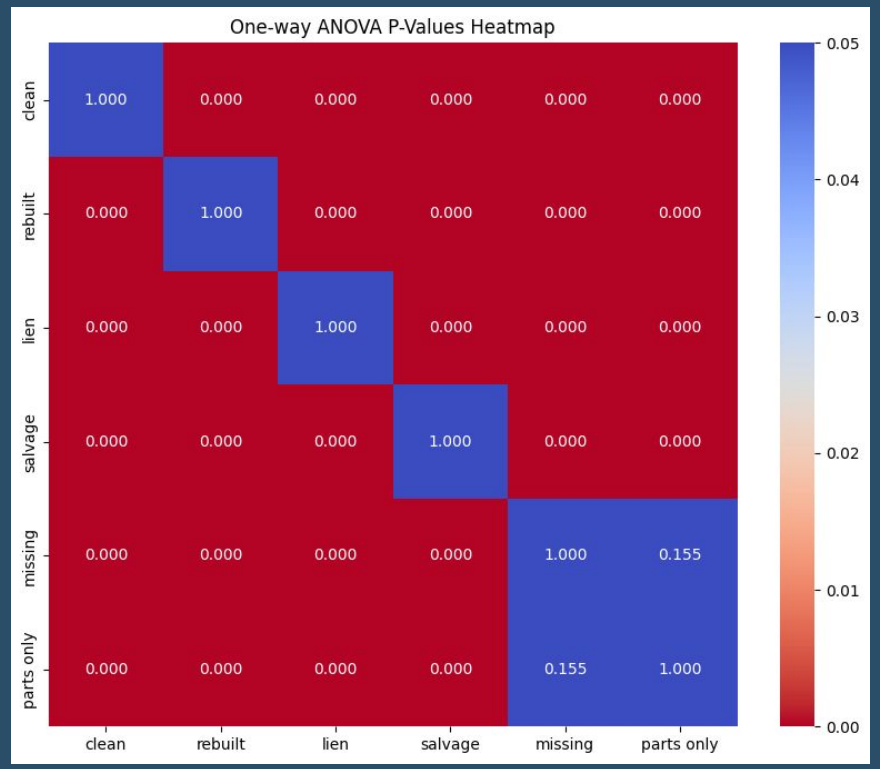
The results of ANOVA show that there are significant differences among all “Fuel” types. It is also an expected result based on the cost of different fuel type.

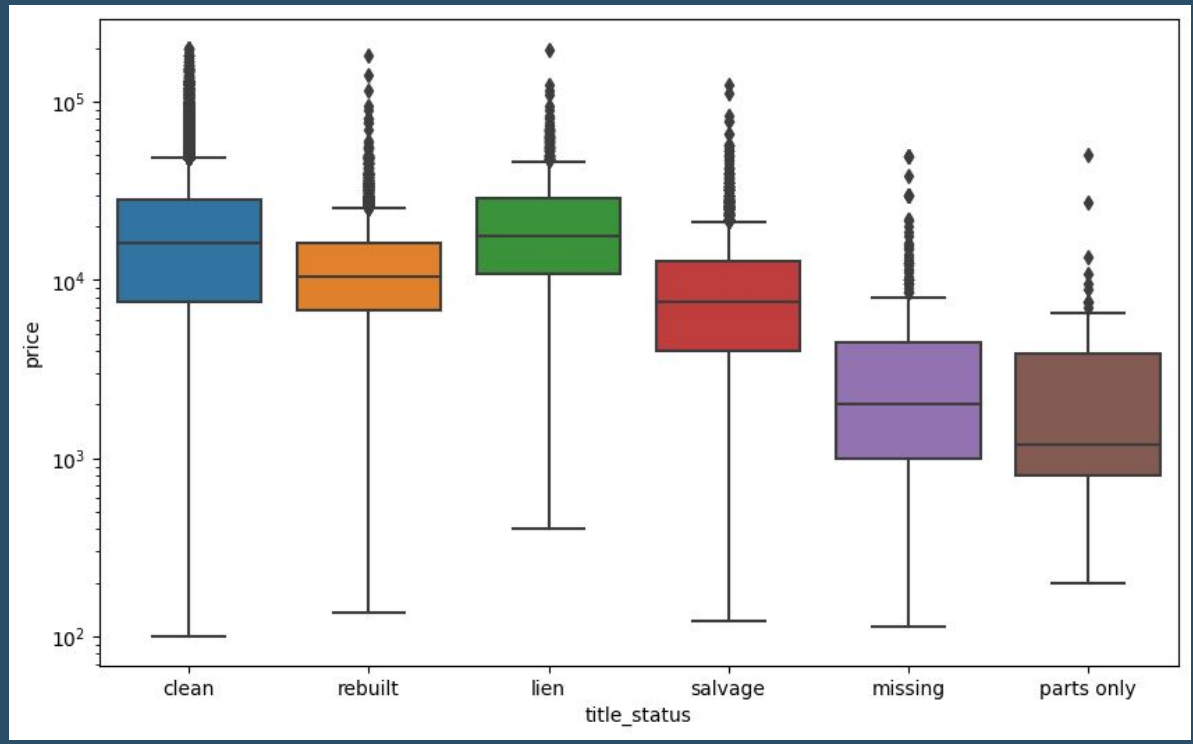




Bivariate Analysis - Title Status

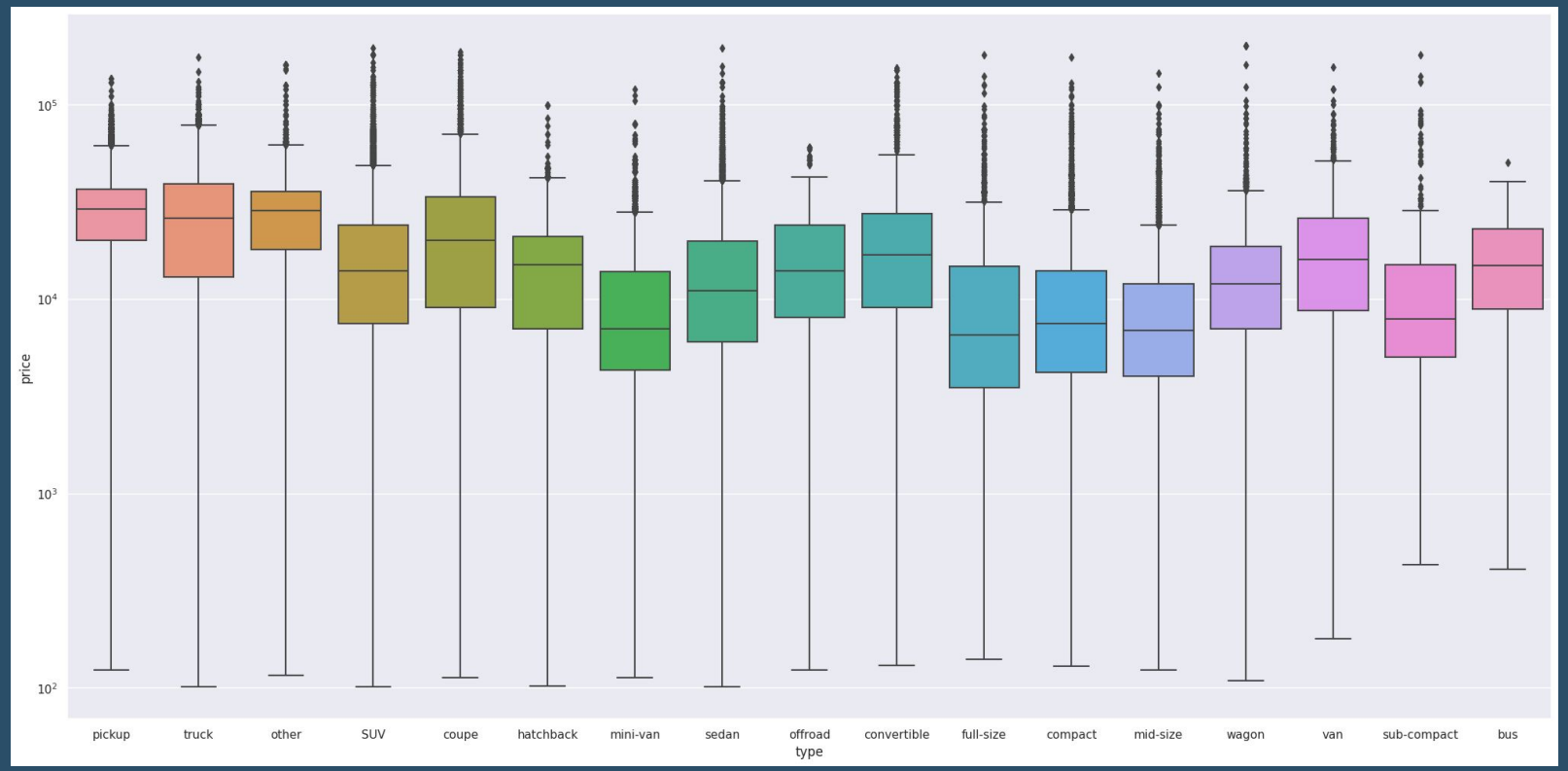
The results of ANOVA show that there are significant differences among most “Title_status” types, with the exception of “Missing” and “Parts only”. It may be the case that people indeed meant the same by them, as in missing (parts). However, we are not confident enough to combine the two categories.



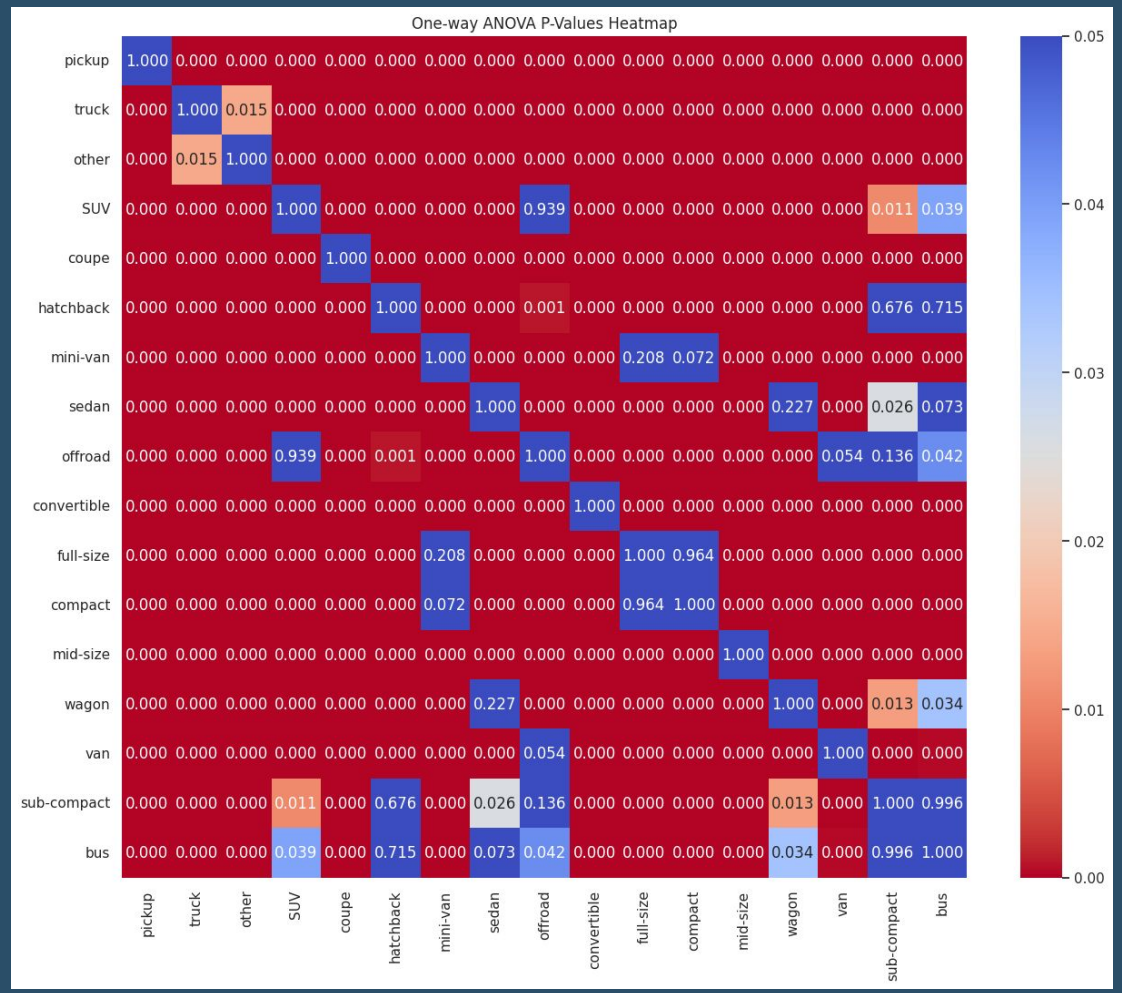


Bivariate Analysis - Type

The box and whisker plots display an interesting shape of long bottom quartile, and a short top quartile with a grouping of outliers above. This is likely due to the fact that very few cars are listed under median but many over the median. It is also showing that each car type has a unique distribution.



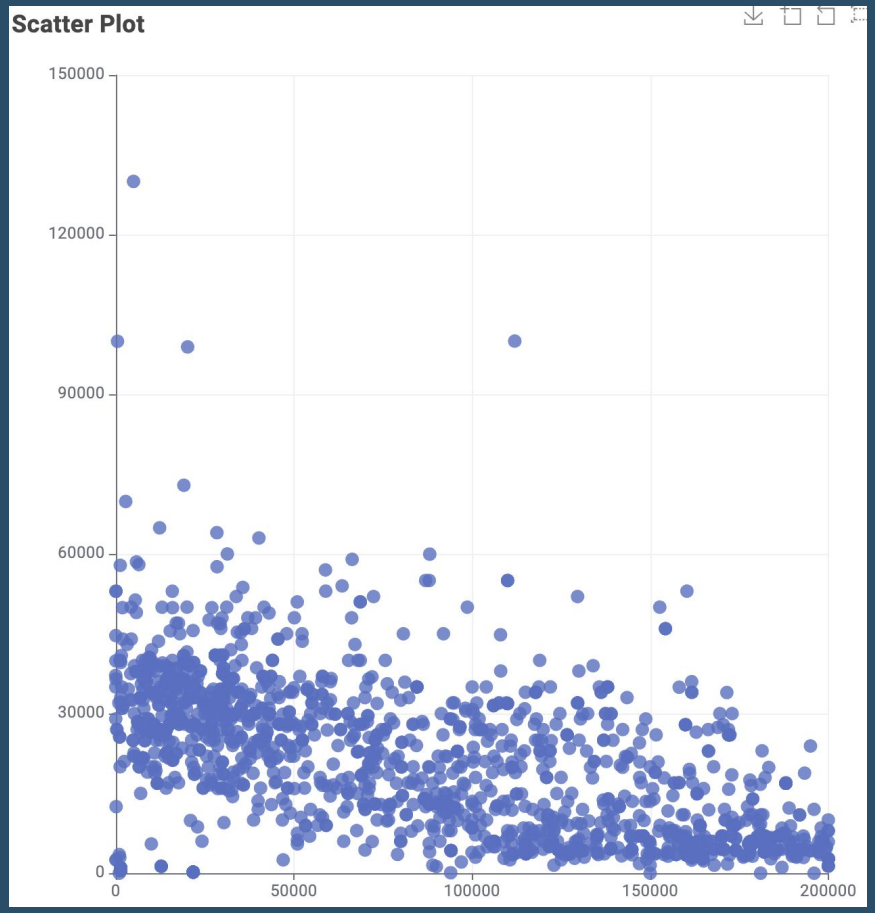
Bivariate Analysis - Type



The results of ANOVA show of car ‘type’ are not so clear at first sight. For example we have a p-value of 0.996 for ‘bus’ and ‘sub-compact’, where they literally target cars at the opposite side of the spectrum. It may be caused by a relatively high cardinality of this variable, thus we will still keep all of them.

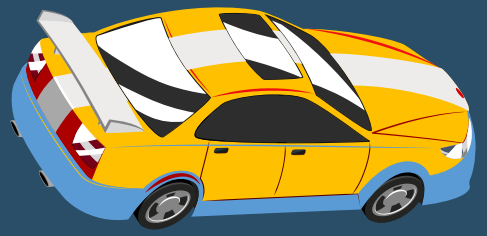
However for the most part, the distributions of each type is different from the others. Some that do have high p-values for ANOVA are Off-Road and Sub-Compact, as they have more than 3 high correlation boxes.

Relationship between Price and Miles



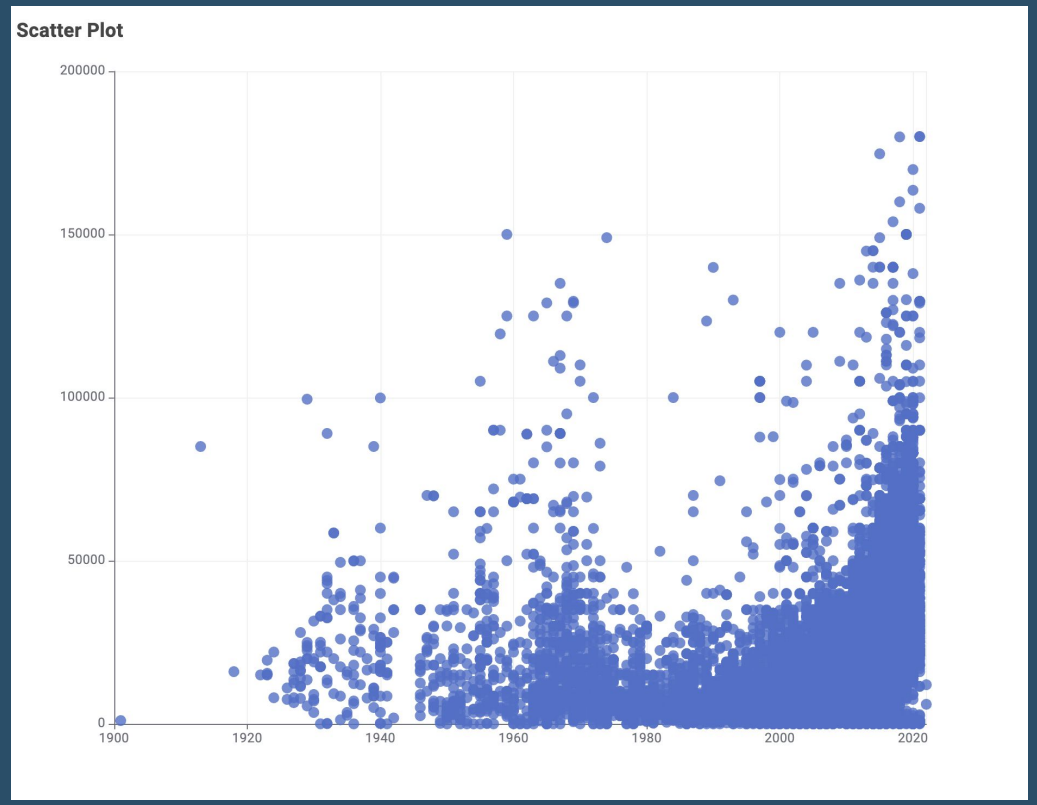
(Subset of Data)

The scatterplot which maps prices and distance driven shows an overall negative slope. Why is this the case? Well as a car is used more and more, it loses value. This is known as mileage depreciation. So it does indeed make logical sense as to why we observe this negative relationship.



While there are indeed outliers that are selling for very expensive, that could be attributed to luxury or collection cars, most points will lie under the boundary that has been formed.

Relationship between Price and Years



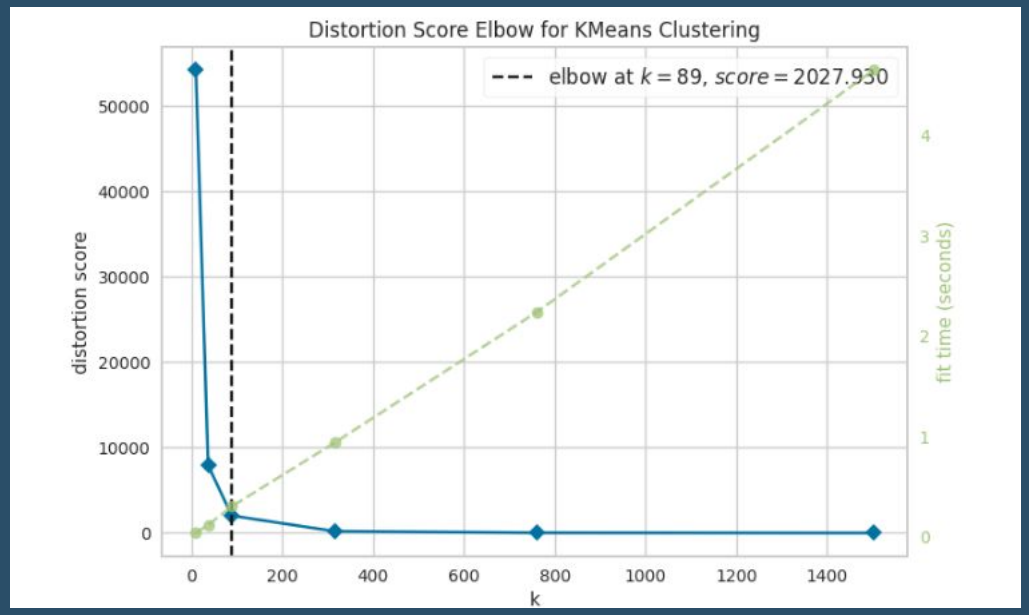
We can see a strong positive correlation between year manufactured and price especially in the models after 1980. It is important to note that some older cars can be considered as vintage, antique, and collector vehicles giving them a high price depending on their conservation state.

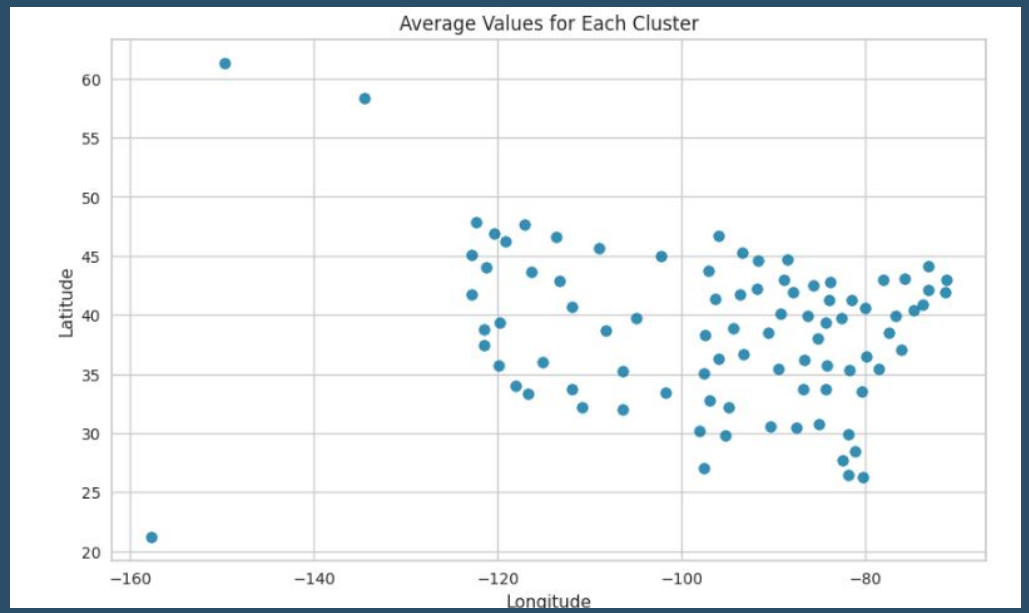
The trend is less clear for the older car models, so we will have to further investigate how it impacts certain models of cars more than other. We can also see other interesting features of the data, like how there barely any cars produced in the mid 1940s due to WW2. So the data is representative of what we predicted would be the case.

Modeling

Clustering

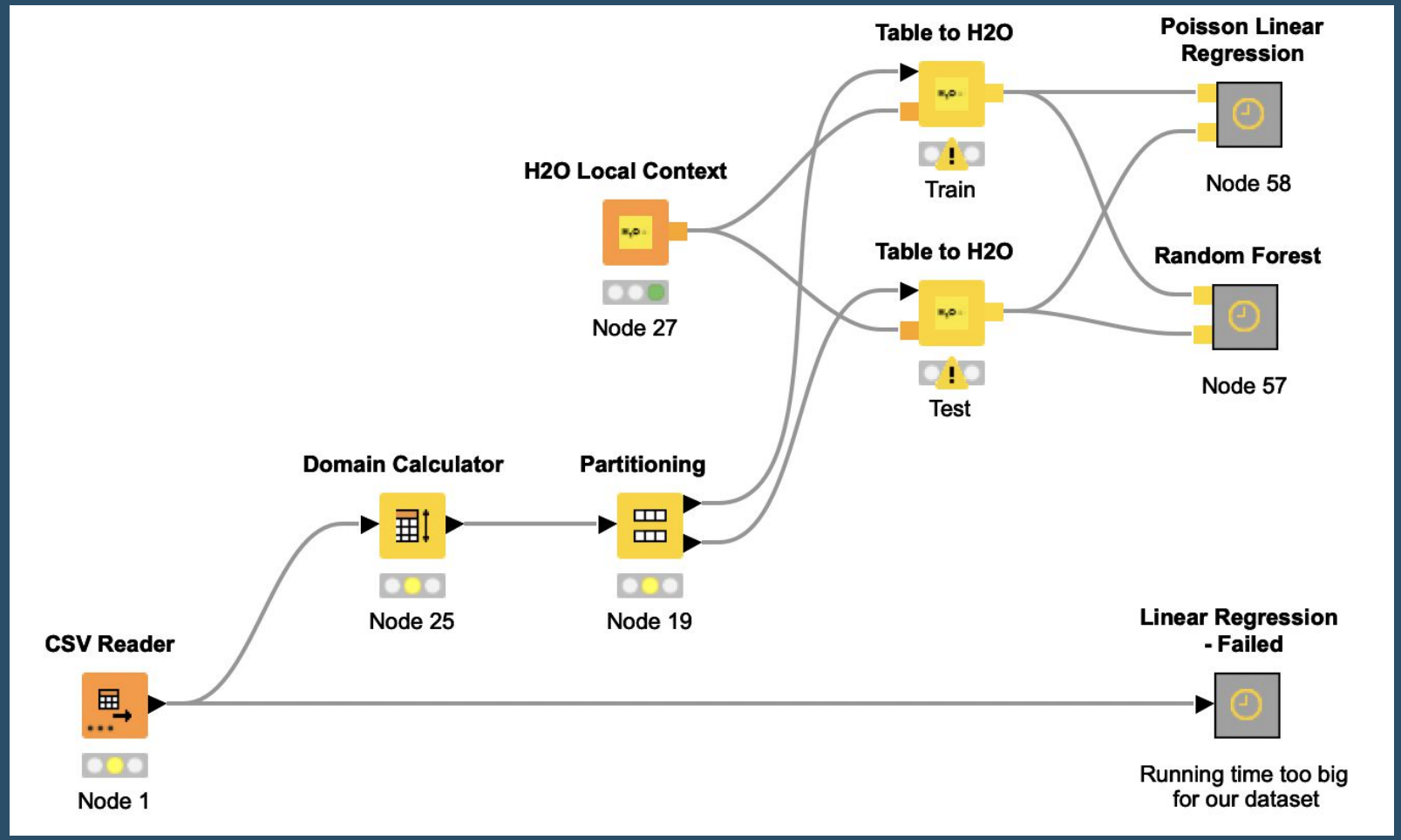
We decided to use clustering to account for places the cars are sold in. We run Knn clustering by longitude and latitude and determine the most optimal K. We choose candidates for K as number of cities above certain number of people in the US. By elbow curve analysis we see the most optimal clusters associated the data with cities of 250000 and above which there are 89 in the United States. We checked the results and only one cluster was not ascribed to a big city: the one between Alaska and US: which is understandable since its a part of a faraway isolated territory bordering Canada. Those clusters will be later assigned to each data point, one-hot encoded and used for analysis.





1. https://www.statista.com/statistics/241695/number-of-us-cities-towns-villages-by-population-size/

Models Overview



Linear Regression - Base Model

```
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 10060 on 341203 degrees of freedom
Multiple R-squared:  0.529,     Adjusted R-squared:  0.5289
F-statistic:  8331 on 46 and 341203 DF,  p-value: < 2.2e-16
```

	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	3.040e+04	4.556e+02	66.720	< 2e-16	***
odometer	-1.295e-02	1.665e-04	-77.735	< 2e-16	***
age	-9.147e+02	3.170e+00	-288.573	< 2e-16	***
usage	2.561e-02	1.701e-03	15.055	< 2e-16	***
compound	1.230e+03	3.875e+01	31.749	< 2e-16	***

We have an multiple R-squared value of 0.529 indicating that the the independent variables collectively provide a reasonable fit to the dependent variable.

The non-categorical data points: Odometer, Age, Usage, and Compound are all significant. Odometer has a negative estimate, as engine usage decreases the value of the vehicle.

Usage, the feature we built that assess how concentrated the use of a car has been is also significant. This is opposite of what we expected to see, but it may be too indicative of age of vehicle and missing the information we aimed to capture.

Compound sentiment increases the price. People might be overly optimistic in the description in hopes to sell a car for a higher price relative to its actual worth.

Linear Regression - Type

type_SUVTrue	1.500e+03	1.050e+02	14.285	< 2e-16	***
type_busTrue	-5.233e+02	6.678e+02	-0.783	0.43334	
type_compactTrue	2.496e+03	1.391e+02	17.941	< 2e-16	***
type_convertibleTrue	9.564e+03	1.650e+02	57.948	< 2e-16	***
type_coupeTrue	7.057e+03	1.288e+02	54.805	< 2e-16	***
type_full.sizeTrue	-1.525e+03	1.942e+02	-7.850	4.16e-15	***
type_hatchbackTrue	8.357e+02	1.326e+02	6.304	2.90e-10	***
type_mid.sizeTrue	4.235e+02	1.387e+02	3.052	0.00227	**
type_mini.vanTrue	6.900e+02	1.609e+02	4.288	1.80e-05	***
type_offroadTrue	8.292e+03	4.737e+02	17.505	< 2e-16	***
type_otherTrue	5.989e+03	1.259e+02	47.584	< 2e-16	***
type_pickupTrue	5.184e+03	1.175e+02	44.135	< 2e-16	***
type_sedanTrue	9.554e+02	1.060e+02	9.011	< 2e-16	***
type_sub.compactTrue	8.389e+03	5.253e+02	15.968	< 2e-16	***
type_truckTrue	5.328e+03	1.186e+02	44.917	< 2e-16	***
type_vanTrue	2.365e+03	1.496e+02	15.805	< 2e-16	***

The estimate of sedan is 9.5 while for SUV it is 1.5. This indicates the average change in price for a particular category relative to the reference category. The estimate for sedans is larger than the estimate for SUVs, suggesting that, on average, sedans tend to have higher predicted prices compared to SUVs in the model.

We can observe the estimates of the one hot encoding as the effect as the change in the predicted value of the dependent variable compared to the reference category. All of the variables are significant except for Bus. We do not have sufficient evidence to claim that the predicted value for cars labeled as Buses is different from the reference category after accounting for other variables in the model.

Linear Regression - Categorical

condition_excellentTrue	-1.902e+03	4.652e+01	-40.882	< 2e-16	***
condition_fairTrue	-4.284e+03	1.472e+02	-29.110	< 2e-16	***
condition_goodTrue	-1.572e+03	5.144e+01	-30.566	< 2e-16	***
condition_like.newTrue	3.767e+01	8.042e+01	0.468	0.63944	
condition_newTrue	6.551e+03	3.363e+02	19.477	< 2e-16	***
condition_salvageTrue	-2.863e+03	4.774e+02	-5.997	2.01e-09	***

All encoded condition variables are significant but Like_new. It is possible that this is a typical slightly over-optimistic condition and not honest.

As expected, being in the category new has the strongest influence on price. Although it is strange to see Salvage with a less negative estimate than Fair, it may be the case than the only salvages worth a significant amount are listed on craigslist versus just sending to scrapyard.

transmission_automaticTrue	-5.322e+03	6.640e+01	-80.151	< 2e-16	***
transmission_manualTrue	-1.872e+03	1.021e+02	-18.329	< 2e-16	***

Both transmission types are significant but have negative intercepts compared to the baseline of Other. It may be that cars labeled under Other transmission are unique or sports cars with a mixed transmission. Manuals are not an overly common feature of cars in the US, so it may be a price boosting feature.

fuel_dieselTrue	9.819e+03	1.042e+02	94.264	< 2e-16	***
fuel_electricTrue	1.002e+03	4.772e+02	2.099	0.03583	*
fuel_gasTrue	-1.951e+03	7.181e+01	-27.161	< 2e-16	***
fuel_hybridTrue	1.756e+02	1.675e+02	1.049	0.29439	

We can see that only two of the variables for fuel are fully significant. When looking back to the distribution of fuels, approximately 95% of data falls into either gas or diesel. Non-significance may be due to lack of enough reliable points to achieve significance.

Linear Regression - Categorical

title_status_cleanTrue	4.577e+03	1.770e+02	25.864	< 2e-16	***
title_status_lienTrue	5.660e+03	3.373e+02	16.779	< 2e-16	***
title_status_missingTrue	7.238e+03	5.805e+02	12.469	< 2e-16	***
title_status_parts.onlyTrue	3.490e+02	9.836e+02	0.355	0.72275	
title_status_rebuiltTrue	1.204e+03	2.155e+02	5.585	2.33e-08	***

The only non-significant variable is Parts Only. Less than one percent of data points are Parts Only so there may have not been enough predictive power behind Parts Only. It is strange to see that the estimate for missing has the strongest influence on price, as it is a mix of bad and good titles.

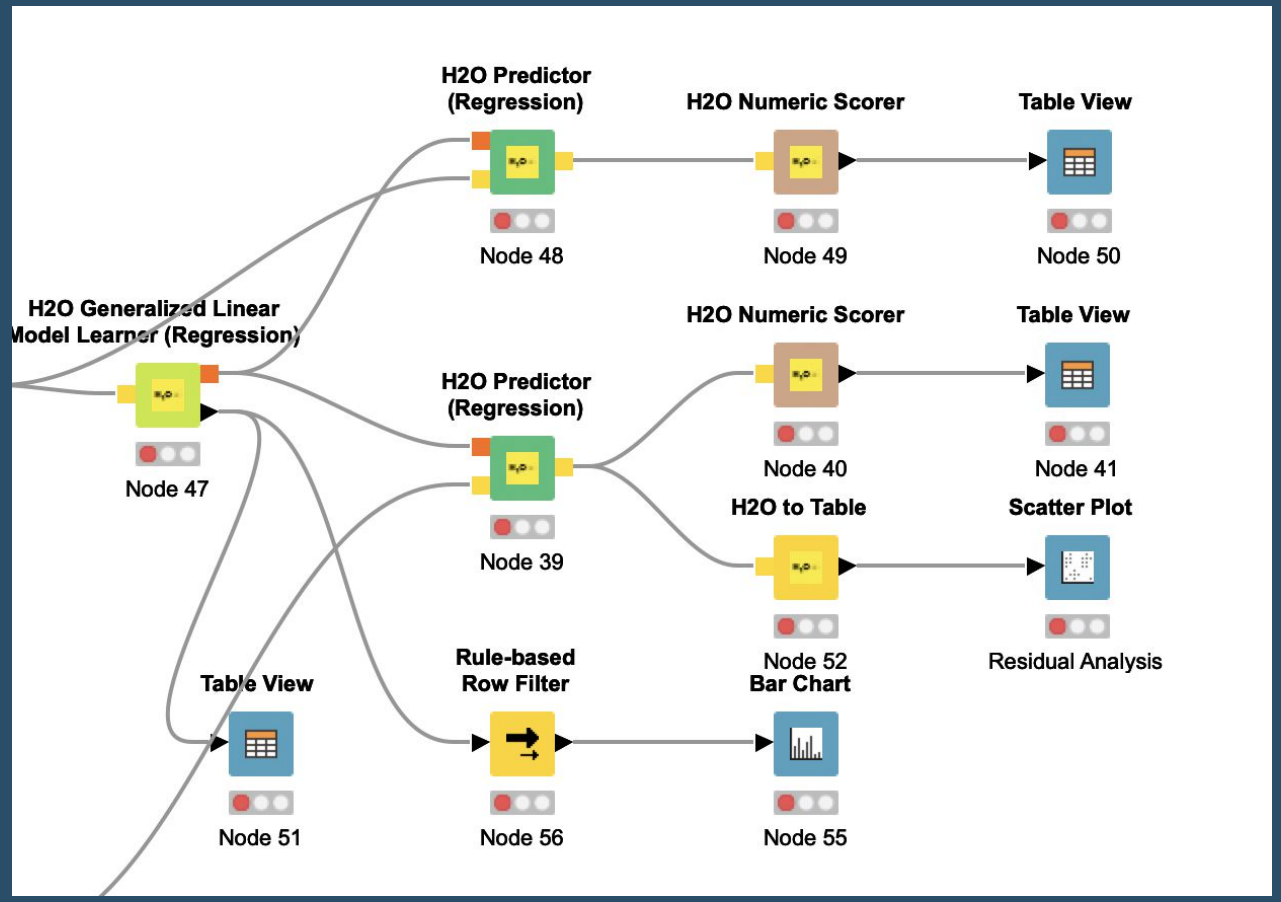
drive_4wdTrue	1.015e+03	5.320e+01	19.074	< 2e-16	***
drive_fwdTrue	-4.239e+03	5.684e+01	-74.589	< 2e-16	***

Both variables are significant while RWD is the baseline comparison. We observe that 4WD cars have a positive estimate while FWD has a much lower negative coefficient. This can be explained by the fact that 4WD cars are typically SUVs while FWD cars are often compact or hatchbacks.

cylinders_10.cylindersTrue	-2.031e+03	4.683e+02	-4.336	1.45e-05	***
cylinders_12.cylindersTrue	1.316e+04	9.076e+02	14.502	< 2e-16	***
cylinders_3.cylindersTrue	-6.497e+03	4.355e+02	-14.917	< 2e-16	***
cylinders_4.cylindersTrue	-6.222e+03	3.959e+02	-15.714	< 2e-16	***
cylinders_5.cylindersTrue	-5.377e+03	4.446e+02	-12.094	< 2e-16	***
cylinders_6.cylindersTrue	-2.851e+03	3.960e+02	-7.199	6.06e-13	***
cylinders_8.cylindersTrue	1.207e+03	3.968e+02	3.043	0.00234	**

All cylinder associated variables are significant and indicate levels of power in determining price. We see 8 and 12 cylinders with the highest coefficient for price, while interestingly 10 is negative. The two most negative are 3 and 4 cylinders. Cars with 3 and 4 cylinder engines are tiny ones with a smaller price.

Poisson Regression

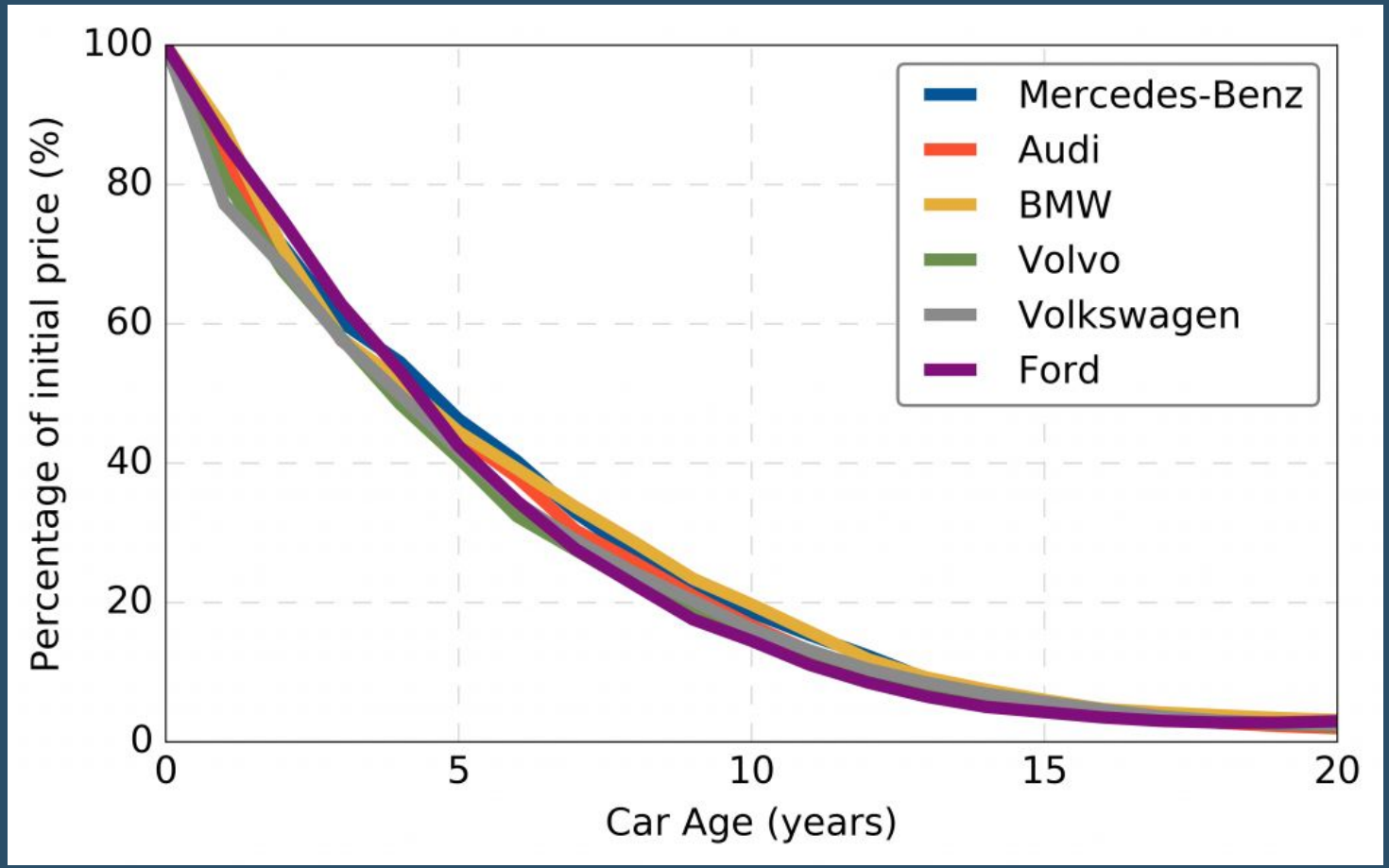


Poisson models are a good fit when there is a lot of count. Poisson regression models the rate of events. If we want to investigate a specific attribute impact on car price then the poisson regression is quite suitable.

We also implemented a Poisson Regression as it is a count based predictor. By that, it will make significantly better use of the count data like cylinders. Our initial assumption is that the Poisson regression will perform better than simple regression but worse than more complex machine learning models like Random Forests

Why Poisson Regression

Car value decreases exponentially over time, where time is a cofounder for change in all the observable characteristics of the car. We suspect that therefore a poisson model which exponentiates the design matrix is more truthful then a simple linear model. Additionally Poisson models perform well with high amount of binary variables which are present in the dataset. Therefore before training ensemble models to account for more complicated relationships in the data we decided to give this statistical model a try.



Model Performance

For a simple generalised linear model it performs very well with both training and test R^2 at 0.715 and 0.705 respectively, although we consider RMSE to still be too high to use it to price cars effectively. However it's a great improvement compared to linear regression model with R^2 of 0.53. The similarity in train and test R^2 indicate that the model is at its full capabilities. There is a clear concentration visible Price=Predicted Price line which graphically indicates good performance.

Train scoring

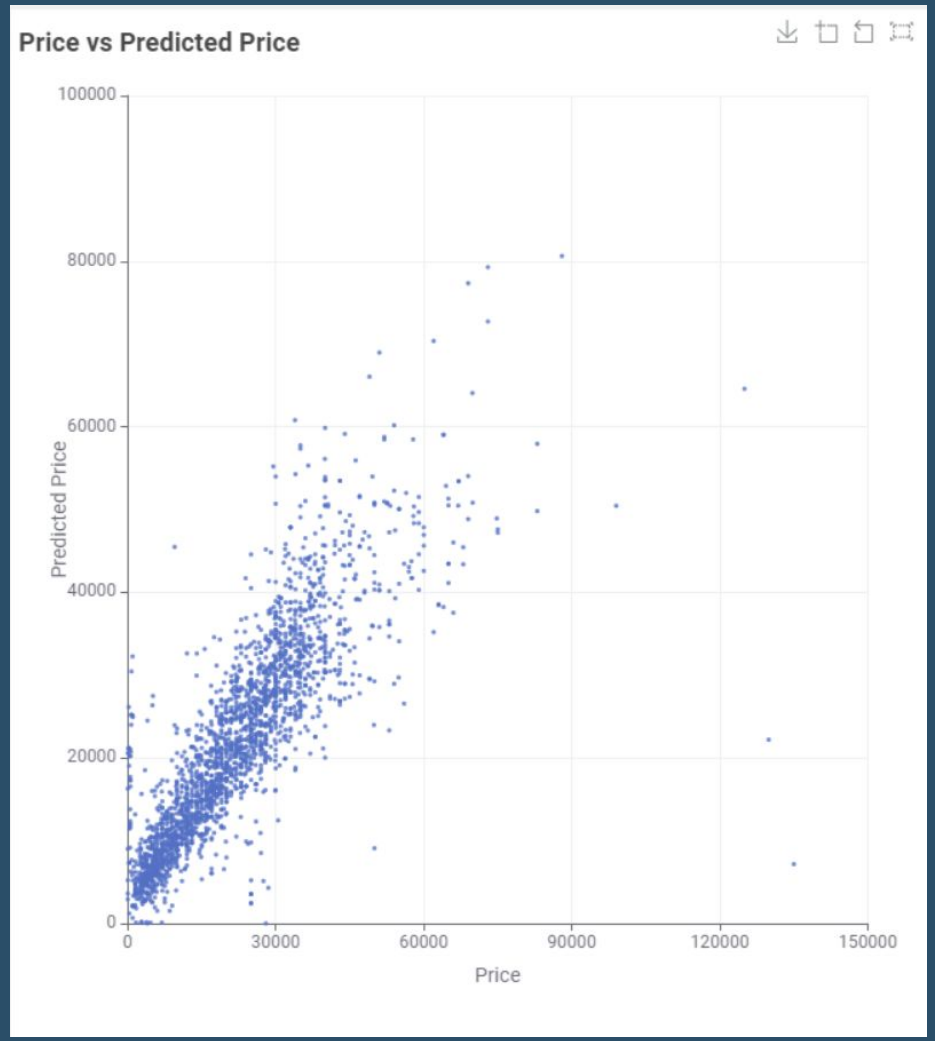
Rows: 1 | Columns: 3

R2 Nu...	RMSE Number...	RMSLE Number (double)
0.715	7,809.257	0.979

Test scoring

Rows: 1 | Columns: 3

R2 Nu...	RMSE Number...	RMSLE Number (double)
0.705	8,031.578	0.976



Coefficient Analysis

Poisson model is a type of linear model where logarithm of the price is being the target variable.The interpretation is such: holding all other variables in the model constant, increasing X by 1 unit (or going from 1 level to the next) multiplies the rate of Y by $e^{coefficient.}$ Most of the dummy variables are zero, nonetheless there are some that are valuable for the prediction.

condition=new	0.02782336039410893
condition=excellent	0.0
condition=like new	0.0
condition=unknown	0.0
condition=salvage	0.0
condition=good	-0.0479319081017735
condition=fair	-0.5388279456317377

Diesel has a big significant multiplier effect on the car, while gas negative, compared to other fuel types which are the baseline (coefficient=0)

fuel=gas	-0.083
fuel=diesel	0.372

3,4,5 cylinders decrease the value of a car and 8 cylinders increased it compared to all other number of cylinders

cylinders=5 cylinders	-0.034
cylinders=3 cylinders	-0.132
cylinders=4 cylinders	-0.169
cylinders=8 cylinders	0.206

Coefficient Analysis

Manuals are more valuable than automatic: that makes sense since there is a premium for rarity in the United States market.

transmission=manual	0.071
transmission=automatic	-0.076

Fwd drive significantly decreases the listing price of the car compared to rwd and 4wd. Fwd car are typically cheaper small vehicles so this makes sense. The coefficients are similar to that of the simple regression.

drive=4wd	0.031
drive=rwd	0
drive=fwd	-0.216

Most of dummy variables indicating the specific model are equal to 0

Once again usage is surprising: the coefficient is positive which indicates that with increase of usage, listing price increases. That might be people putting more sentimental\subjective value on the car since they grew accustomed to it

usage	6.677603447206968E-6

Age as well as odometer are both negatively associated with the price listing. Compound sentiment score has positive impact on the price as expected

odometer	-5.157088549317581E-6
age	-0.039845387186444634

compound	0.021

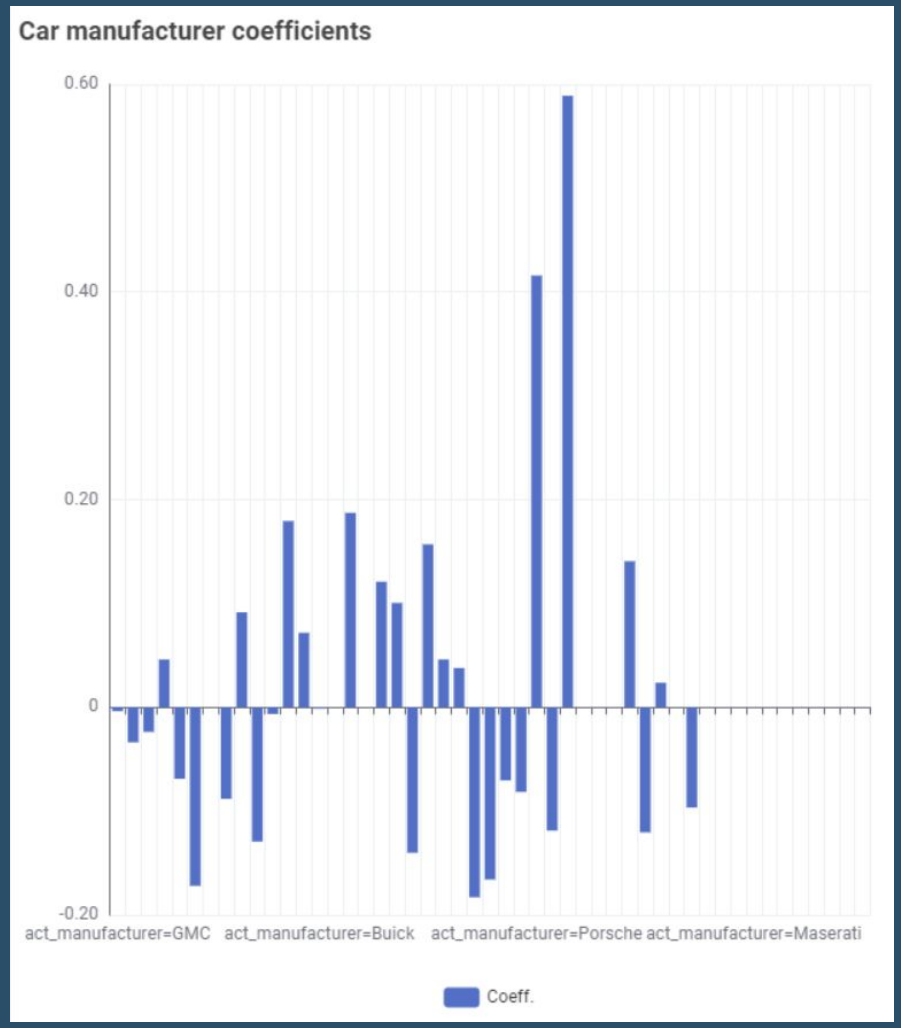
Difference among manufacturers

Luxury manufacturers tend to have higher prices keeping other factors constant: especially Ferrari and Porsche have increased prices ~1.8 and ~1.5 times respectively

Some manufacturers tend to have lower listing prices, despite some of them not being usually considered cheap

act_manufacturer=Ferrari	0.589
act_manufacturer=Porsche	0.416
act_manufacturer=Audi	0.187
act_manufacturer=Lexus	0.179
act_manufacturer=Mercedes-Benz	0.157
act_manufacturer=Land Rover	0.141
act_manufacturer=Lincoln	0.121
act_manufacturer=Alfa Romeo	0.101
act_manufacturer=Cadillac	0.092
act_manufacturer=Jaguar	0.072
act_manufacturer=Acura	0.046
act_manufacturer=Toyota	0.046
act_manufacturer=BMW	0.038
act_manufacturer=Tecstar, LP	0.024

act_manufacturer=Buick	-0.001
act_manufacturer=GMC	-0.004
act_manufacturer=Honda	-0.006
act_manufacturer=Ford	-0.024
act_manufacturer=Chevrolet	-0.034
act_manufacturer=Jeep	-0.069
act_manufacturer=Subaru	-0.07
act_manufacturer=Volkswagen	-0.081
act_manufacturer=Mazda	-0.088
act_manufacturer=Saturn	-0.097
act_manufacturer=Kia	-0.119
act_manufacturer=Mercury	-0.12
act_manufacturer=Dodge	-0.129
act_manufacturer=Hyundai	-0.14
act_manufacturer=Mitsubishi	-0.166
act_manufacturer=Nissan	-0.172
act_manufacturer=Chrysler	-0.183

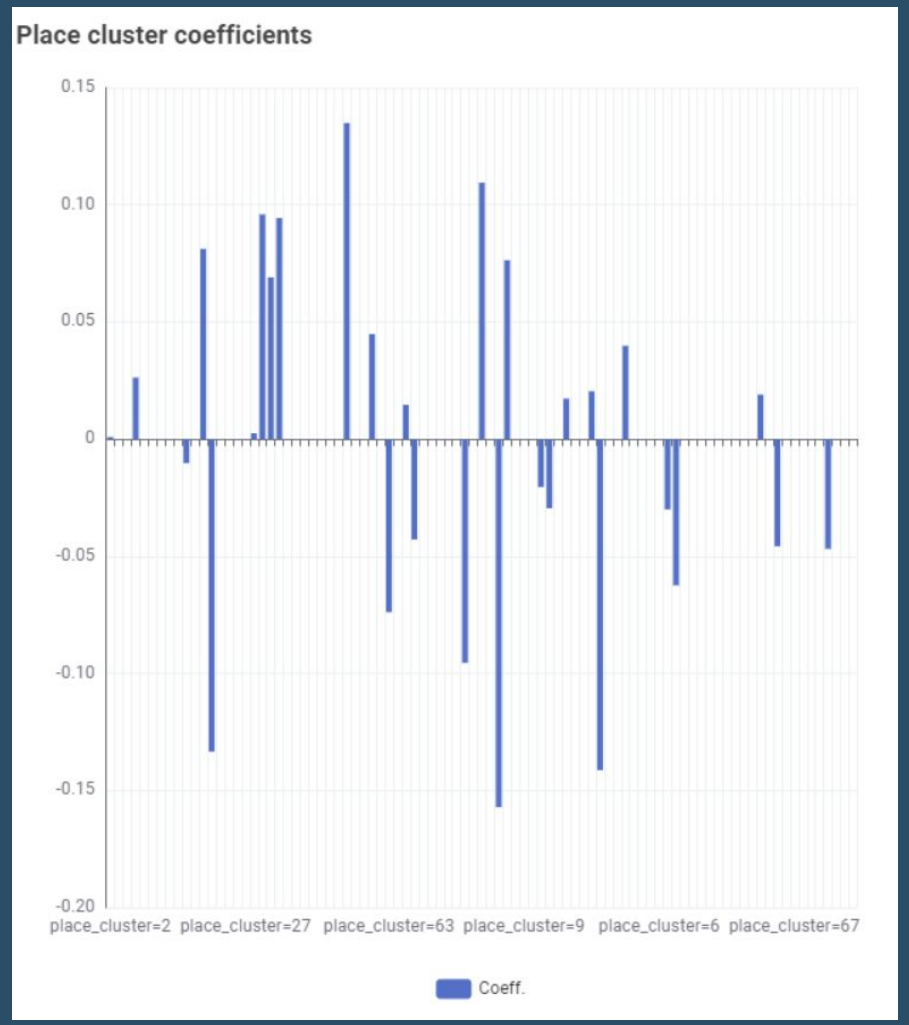


Impact of place of listing

There are certain places that tend to have prices higher up to 1.14 than normal and 0.86 times lower.

place_cluster=19	-0.01
place_cluster=84	-0.02
place_cluster=36	-0.03
place_cluster=76	-0.03
place_cluster=81	-0.043
place_cluster=47	-0.046
place_cluster=28	-0.047
place_cluster=58	-0.062
place_cluster=63	-0.074
place_cluster=1	-0.096
place_cluster=32	-0.133
place_cluster=43	-0.141
place_cluster=18	-0.157

place_cluster=60	0.135
place_cluster=38	0.109
place_cluster=5	0.096
place_cluster=51	0.094
place_cluster=61	0.081
place_cluster=29	0.076
place_cluster=8	0.069
place_cluster=12	0.045
place_cluster=30	0.04
place_cluster=78	0.026
place_cluster=14	0.02
place_cluster=83	0.019
place_cluster=24	0.017
place_cluster=7	0.015
place_cluster=0	0.003
place_cluster=2	0.001



Models and feature engineering impact

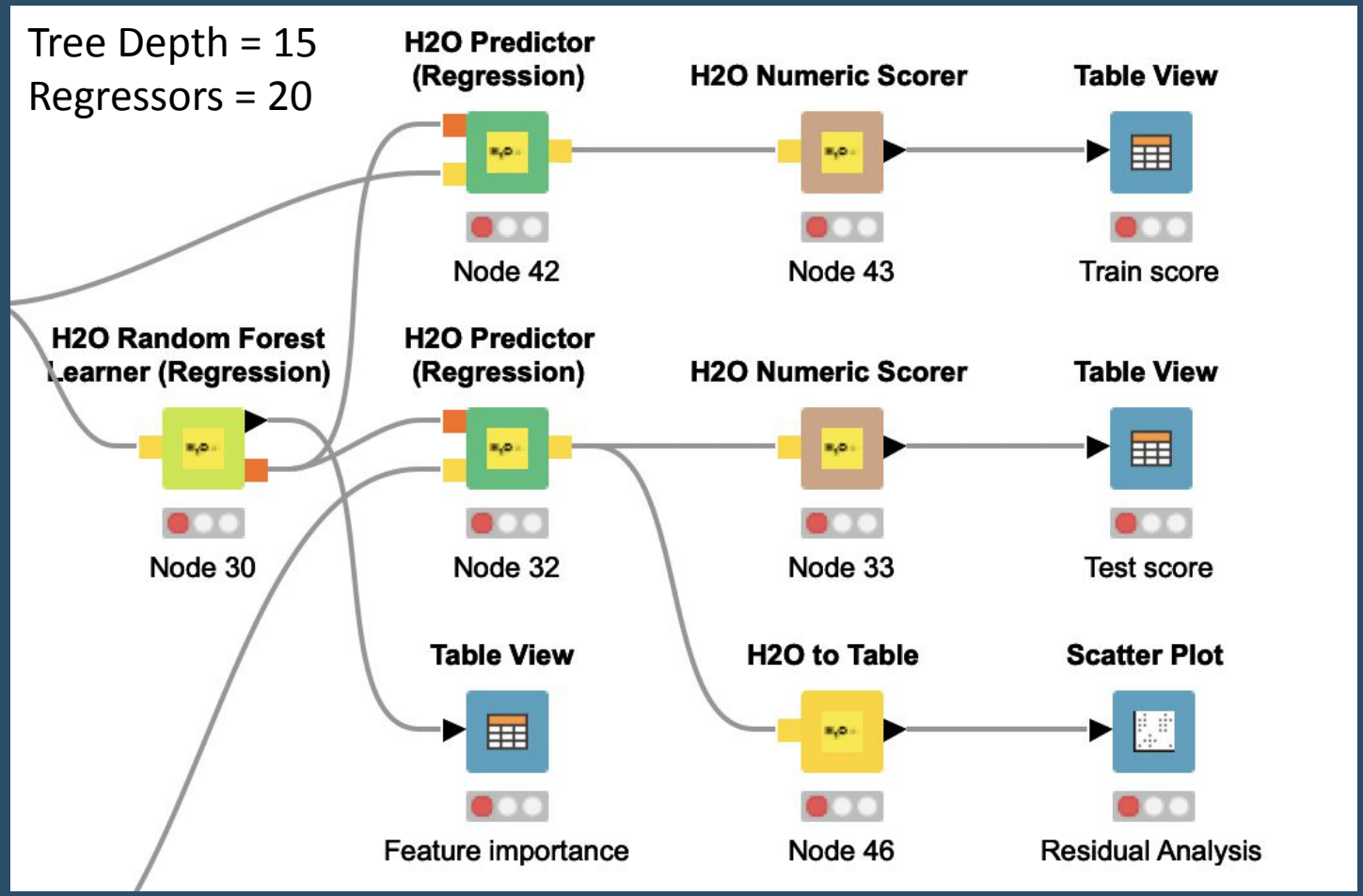
We have run lazy regression in Python to decide which ML models has the best potential to be used in Knime. Lazy regression performs many fits with training time-accuracy tradeoff more on the advantage of time, to scan for the would perform the best. We decide to run it before and after performing feature engineering to also see the effects of our work.

Model	Adjusted R-Squared	R-Squared	RMSE	Time Taken	RMAE
ExtraTreesRegressor	0.62	0.62	0.05	27.20	0.22
BaggingRegressor	0.62	0.62	0.05	6.82	0.22
KNeighborsRegressor	0.57	0.58	0.05	1.13	0.22
ExtraTreeRegressor	0.56	0.57	0.05	0.34	0.22
DecisionTreeRegressor	0.56	0.56	0.05	1.03	0.22
HistGradientBoostingRegressor	0.41	0.41	0.06	2.90	0.24
GradientBoostingRegressor	0.39	0.40	0.06	20.13	0.24
LassoLarsCV	0.12	0.12	0.07	0.73	0.27
LarsCV	0.12	0.12	0.07	0.75	0.27
Lars	0.12	0.12	0.07	0.09	0.27
LassoLarsIC	0.12	0.12	0.07	0.35	0.27
BayesianRidge	0.12	0.12	0.07	0.16	0.27
ElasticNetCV	0.12	0.12	0.07	2.16	0.27
LassoCV	0.12	0.12	0.07	2.16	0.27
AdaBoostRegressor	0.11	0.12	0.07	3.04	0.27

Model	Adjusted R-Squared	R-Squared	RMSE	Time Taken	RMAE
ExtraTreesRegressor	0.76	0.77	0.04	40.26	0.19
BaggingRegressor	0.73	0.74	0.04	13.08	0.19
KNeighborsRegressor	0.59	0.61	0.05	2.22	0.21
DecisionTreeRegressor	0.55	0.57	0.05	2.10	0.22
HistGradientBoostingRegressor	0.52	0.55	0.05	2.26	0.22
ExtraTreeRegressor	0.52	0.54	0.05	0.49	0.22
GradientBoostingRegressor	0.47	0.49	0.05	36.21	0.23
LassoLarsCV	0.24	0.28	0.06	0.68	0.25
LarsCV	0.24	0.28	0.06	0.66	0.25
Lars	0.24	0.28	0.06	0.09	0.25
LassoLarsIC	0.24	0.28	0.06	0.35	0.25
BayesianRidge	0.24	0.28	0.06	0.17	0.25
LassoCV	0.24	0.28	0.06	2.13	0.25
ElasticNetCV	0.24	0.28	0.06	2.34	0.25
HuberRegressor	-0.02	0.04	0.07	3.04	0.27

The increases in the Pseudo R Squared score are very noticeable, with neary 0.1 increase for all models tested. We can also gain an understanding of which models are best suitable for our specific case. The top 3 in both cases is Extra Trees Regressor, Bagging Regressor, and K Neighbors Regressor. We decided to go with Extra Tree Regressor called here Extra Tree Regressor

Random Forest Regressor



Poisson models are a good fit when there is a lot of count. Poisson regression models the rate of events. If we want to investigate a specific attribute impact on car price then the poisson regression is quite suitable.

We also implemented a Poisson Regression as it is a count based predictor. By that, it will make significantly better use of the count data like cylinders. Our initial assumption is that the Poisson regression will perform better than simple regression but worse than more complex machine learning models like Random Forests

Performance Analysis

Random Forest Regressor performs almost ideally: it is able to achieve 0.927 and 0.881 R^2 respectively. Despite the fact that it is a somewhat a black box tool, we are still able to extract feature importance. We see that the model takes advantage first and foremost of data regarding age of the car and odometer with 0.295 and 0.175 importance respectively, accounting for 0.47 of the decision process. The next 9 variables make up approximately equal amount of decision process, the remaining 6 only approximately 0.06. The low importance of usage is highly unexpected, but, since this is a variable obviously correlated with odometer and age, it might indeed provide little additional information. The same can be said about the variable condition: it is possible that most of the information it provides can also be inferred from odometer and age.

Train score

Train scoring

Rows: 1 | Columns: 3

R2 *Nu...*	RMSE *Number...*	RMSLE *Number (double)*
0.927	3,964.54	0.766

Test score

Test scoring

Rows: 1 | Columns: 3

R2 *Nu...*	RMSE *Number...*	RMSLE *Number (double)*
0.881	5,065.905	0.797

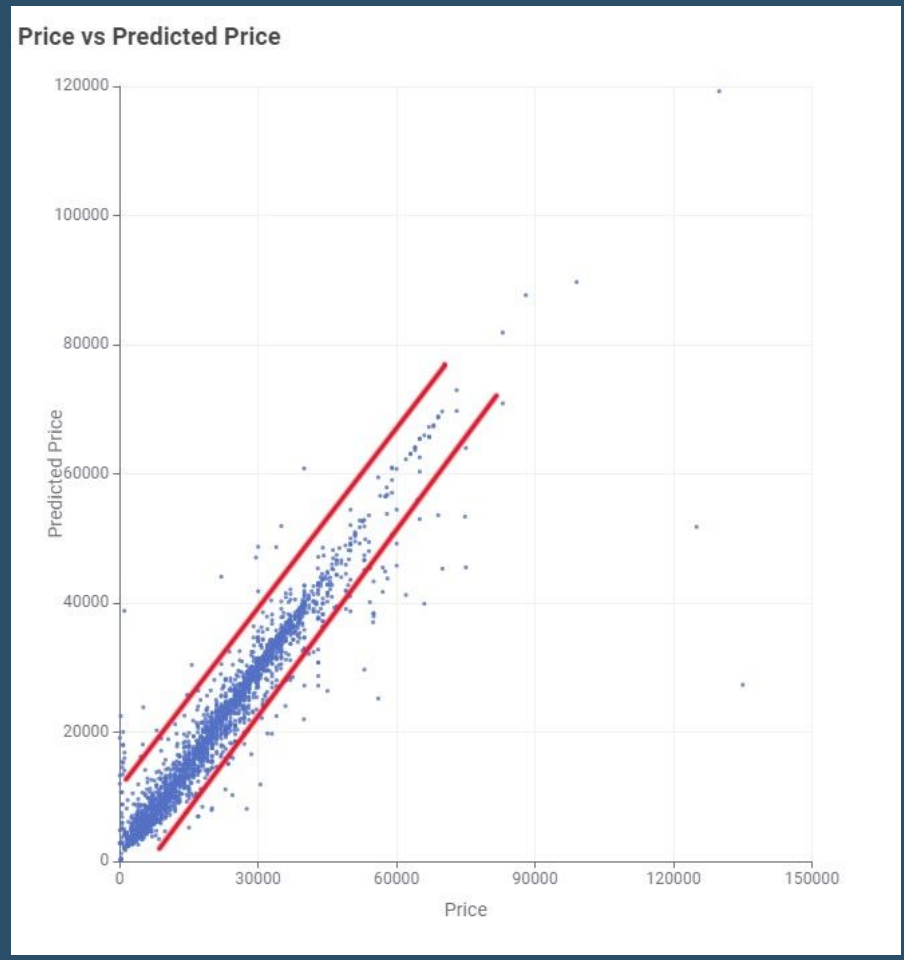
RowID	Scaled Importance *Number (double)*	Percentage *Number (double)*
age	1	0.295
odometer	0.592	0.175
type	0.317	0.094
place_cluster	0.244	0.072
drive	0.238	0.07
act_manufacturer	0.214	0.063
fuel	0.187	0.055
cylinders	0.171	0.05
actual_model	0.106	0.031
compound	0.078	0.023
transmission	0.055	0.016
neu	0.044	0.013
condition	0.044	0.013
pos	0.035	0.01
usage	0.033	0.01
neg	0.026	0.008
title_status	0.004	0.001

Residual analysis for opportunity detection

As we can see from the graph, the model performs very well: we can see a clear straight line where with high precision Price=~Predicted Price. Outliers find themselves roughly outside the area marked by red lines.

Apart from checking for fair price, the model can be used to identify overpriced and underpriced offers. If the predicted price is greater than actual price, then it is very much possible that the car is undervalued and could be a purchase opportunity (points above upper red line)

However, we would be very careful setting the price according to the model. The prices predict here reflect private people's subjective value of given cars on a private market. The predicted prices do not necessarily maximise profits for the business if used for selling own cars. There have been a number of cases where, especially real estate agencies, tried to use ML models to do that and failed. One of those cases is Zillow, where their "Zestimates" failed to account for that and were one of the factors that led to their financial troubles.



1. https://fortune.com/education/articles/what-zillows-failed-algorithm-means-for-the-future-of-data-science/

Comparing Models

Linear Regression

Simplest model we created as baseline. It still performed relatively well as it explained slightly above half of the variance. Almost all coefficients were significant and their direction and magnitude made sense.

Poisson Regression

A model that corrected the linear model assumptions by a fact that cars depreciate exponentially. It explained more variance and provided better, more interpretable results. This model can be used primarily to investigate the effects of given features on the price, and can be somewhat used to estimate fair price and under/overpriced cars.

Random Forest

This model was created with maximum predictive power in mind and this is its purpose: to predict the price. Additionally feature importance might help to infer whether there exists more non poisson and non linear impact of some variables on price. The residuals of this model can definitely be used to detect overpriced and underpriced cars.

Managerial Applications

Understanding the pricing

This predictive model model assists in identifying the most profitable cars for purchasing while guiding the dealership in setting prices. It is important to note that this should help as a tool for people, not an automated system, otherwise it poses threats as described in residual analysis.

While this model is based on cars from one period, the model can be adapted to continuously assess the market and adjusting prices based on the model's insights to avoid overpricing or underpricing.

Competition Analysis

The model can provide valuable insights for competitor analysis, allowing the given dealership to understand and adapt to various competitors' pricing strategies.

This will advantage the dealership as they can identify opportunities to offer superior prices in certain areas of the market where competition underestimates or overestimates pricing compared to private used car market rates.

Customer Experience

The model can enhance overall customer satisfaction by ensure fair and competitive prices for vehicles. With such a complex model that captures differences between the editions of the same model car, customers can have a more transparent experience that also caters to their needs.

The data-driven approach also helps smooth the negotiating process by enabling the buyer or seller to feel more informed.

Managerial Applications II

Know what to look for

We provided a thorough analysis on influence of most features of a car on its price. Our client, combining this information with their business experience and acumen can easily create business plans and strategies, for example target certain types of cars to buy if they think market currently underpriced them or try to sell parts of their stock they think the market is currently overpricing.

Insight into psychology of listings

From the impact of description sentiment to analysing distributions of price by assigned state we can hypothesise about tricks and manipulation people use in their listings. We can leverage the fact that certain listings have high compound sentiment core and thus correct for it when considering listing's price. The results of ANOVA between fair and salvage state imply that people price those cars in a similar way: they just might try to play down how badly damaged the car really is.

Potential future insight

The current model is based on data from a single period. This limits the use of the models continuously as trends change and cars values as well. By fitting the model to update with new datasets, we will be able to better understand car prices and how they fluctuate. The potential to predict when trends in the car prices will allow dealerships to prepare for changes in the market and in good cases, purchase cars which will increase in value in the short future.