

# (Clean) Vehicle Data

Glorija

2024-06-21

## Introduction

This is a presentation of my findings while investigating the dataset car\_prices

The dataset has been cleaned and filtered:

- Removed irrelevant columns such as: trim, vin, body, state, interior, body, seller, and sell date;
- Cleared non-sensical data, missing values, and NAs;
- Made unique makes and colors uniform;
- Converted variables into factors & normalised

## Summary & Structure

```
summary(data)
```

```
##      year      make      odometer      color
## Min.   :1984  Length:511711  Min.   :    1  Length:511711
## 1st Qu.:2008  Class :character  1st Qu.: 28430  Class :character
## Median :2012  Mode   :character  Median : 51838  Mode   :character
## Mean   :2010                           Mean   : 67566
## 3rd Qu.:2013                           3rd Qu.: 97762
## Max.   :2015                           Max.   :999999
##      mmr      sellingprice      transmission      condition
## Min.   : 25   Min.   :    1   Length:511711  Min.   : 1.00
## 1st Qu.: 7300 1st Qu.: 7000  Class :character  1st Qu.:23.00
## Median :12300 Median :12200  Mode   :character  Median :34.00
## Mean   :13806 Mean   :13651                           Mean   :30.57
## 3rd Qu.:18300 3rd Qu.:18200                           3rd Qu.:41.00
## Max.   :182000 Max.   :230000                           Max.   :49.00
```

```
head(data)
```

```
##   year   make odometer color   mmr sellingprice transmission condition
## 1 2015    kia   16639 white 20500     21500  automatic      5
## 2 2015    kia    9393 white 20800     21500  automatic      5
## 3 2014    bmw   1331  gray 31900     30000  automatic     45
## 4 2015   volvo  14282 white 27500     27750  automatic     41
## 5 2014    bmw   2641  gray 66000     67000  automatic     43
## 6 2015   nissan  5554  gray 15350     10900  automatic      1
```

```

str(data)

## 'data.frame': 511711 obs. of 8 variables:
## $ year      : int 2015 2015 2014 2015 2014 2015 2014 2014 2014 ...
## $ make      : chr "kia" "kia" "bmw" "volvo" ...
## $ odometer   : int 16639 9393 1331 14282 2641 5554 14943 28617 9557 4809 ...
## $ color     : chr "white" "white" "gray" "white" ...
## $ mmr       : int 20500 20800 31900 27500 66000 15350 69000 11900 32100 26300 ...
## $ sellingprice: int 21500 21500 30000 27750 67000 10900 65000 9800 32250 17500 ...
## $ transmission: chr "automatic" "automatic" "automatic" "automatic" ...
## $ condition  : int 5 5 45 41 43 1 34 2 42 3 ...

```

## Correlation & Feature Selection

### Correlation Matrix (char & num)

- Cramer's V for Categorical vs Categorical comparison
- Pearson's for Numerical vs Numerical comparison
- Cramer's V for Categorical vs Numerical comparison

```

##           year  make  odometer  color      mmr sellingprice transmission
## year      1.000 0.403 -0.773 0.337  0.593      0.583      0.244
## make      0.403 1.000  0.214 0.282  0.635      0.623      0.403
## odometer -0.773 0.214  1.000 0.187 -0.584     -0.578      0.120
## color     0.337 0.282   0.187 1.000  0.194      0.161      0.245
## mmr       0.593 0.635  -0.584 0.194  1.000      0.984      0.117
## sellingprice 0.583 0.623 -0.578 0.161  0.984      1.000      0.091
## transmission 0.244 0.403  0.120 0.245  0.117      0.091      1.000
## condition   0.537 0.221  0.316 0.189  0.284      0.239      0.208
##           condition
## year          0.537
## make          0.221
## odometer      0.316
## color          0.189
## mmr            0.284
## sellingprice   0.239
## transmission    0.208
## condition      1.000

```

### Correlation Matrix Continued (char & num)

- Some of the correlation coefficient's are below 0.2, so I increased the threshold and filtered out the small correlations

```

## [1] "year"        "make"         "odometer"      "mmr"          "sellingprice"
## [6] "condition"

##           year  make  odometer      mmr condition sellingprice
## year      1.000 -0.011 -0.773  0.593   0.356      0.583
## make     -0.011  1.000 -0.022 -0.065   0.033     -0.063
## odometer -0.773 -0.022  1.000 -0.584  -0.338     -0.578

```

```

## mmr          0.593 -0.065  -0.584  1.000      0.301      0.984
## condition    0.356  0.033  -0.338  0.301      1.000      0.344
## sellingprice 0.583 -0.063  -0.578  0.984      0.344      1.000

```

### PCA (char & num)

```

## Importance of components:
##                               PC1     PC2     PC3     PC4     PC5     PC6
## Standard deviation     1.8032 1.0149 0.8953 0.8223 0.47512 0.1225
## Proportion of Variance 0.5419 0.1717 0.1336 0.1127 0.03762 0.0025
## Cumulative Proportion  0.5419 0.7136 0.8472 0.9599 0.99750 1.0000

##                               PC1     PC2     PC3     PC4     PC5     PC6
## year            -0.462  0.080  0.032 -0.521  0.712  0.021
## make             0.022  0.937 -0.315  0.147  0.031  0.000
## odometer         0.458 -0.115  0.008  0.535  0.701 -0.001
## mmr              -0.497 -0.147 -0.290  0.385  0.009 -0.707
## condition        -0.284  0.251  0.870  0.313 -0.022 -0.038
## sellingprice     -0.498 -0.135 -0.244  0.418 -0.012  0.706

## Principal_Component Variance_Explained
## 1                  PC1      54.1918858
## 2                  PC2      17.1673249
## 3                  PC3      13.3601557
## 4                  PC4      11.2683595
## 5                  PC5      3.7622722
## 6                  PC6      0.2500018

```

### Correlation Matrix (num)

- Seeing as there are many PCs and it takes about 4 of them to explain most of the data, I have decided to do correlation with just numerical variables (low correlation) and see if PCA improves

```

##                               year  odometer   mmr  sellingprice
## year            1.000 -0.773  0.593       0.583
## odometer        -0.773  1.000 -0.584      -0.578
## mmr              0.593 -0.584  1.000       0.984
## sellingprice    0.583 -0.578  0.984      1.000

```

### PCA (num)

```

## Importance of components:
##                               PC1     PC2     PC3     PC4
## Standard deviation     1.7471 0.8395 0.47614 0.12739
## Proportion of Variance 0.7631 0.1762 0.05668 0.00406
## Cumulative Proportion  0.7631 0.9393 0.99594 1.00000

##                               PC1     PC2     PC3     PC4
## year            -0.478 -0.515  0.711 -0.013
## odometer        0.476  0.529  0.703 -0.003
## mmr              -0.523  0.471  0.003  0.710
## sellingprice   -0.521  0.483 -0.014 -0.704

```

```

## Principal_Component Variance_Explained
## 1 PC1 76.3060869
## 2 PC2 17.6203358
## 3 PC3 5.6678479
## 4 PC4 0.4057294

```

## Correlation & Feature Selection Conclusion

- Numerical variables explain the data better and the dimensions are reduced therefore, categorical variables have been filtered out for a better model

## Regression Models

### Linear Regression

```

##
## Call:
## lm(formula = response ~ ., data = as.data.frame(predictors))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.9777 -0.0675  0.0100  0.0822 21.4964
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 9.189e-16 2.513e-04    0.00      1
## year       -8.716e-03 4.122e-04   -21.15  <2e-16 ***
## odometer    -1.169e-02 4.088e-04   -28.59  <2e-16 ***
## mmr         9.820e-01 3.220e-04 3050.17  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1798 on 511707 degrees of freedom
## Multiple R-squared:  0.9677, Adjusted R-squared:  0.9677
## F-statistic: 5.108e+06 on 3 and 511707 DF, p-value: < 2.2e-16

```

### Linear Regression Model Evaluation & Cross-Validation

```

## Linear Regression RMSE: 0.1749224

## Linear Regression R-squared: 0.9676856

```

### Polynomial Regression

```

##
## Call:
## lm(formula = sellingprice ~ poly(year, 2) + poly(odometer, 2) +
##     poly(mmr, 2), data = as.data.frame(train_data))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.1700 -0.0500  0.0000  0.0500  1.1700
## 
```

```

## -8.8690 -0.0673  0.0110  0.0824 21.4976
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)           7.101e-04  2.827e-04   2.512   0.012 *
## poly(year, 2)1      -7.827e+00  3.265e-01  -23.970 < 2e-16 ***
## poly(year, 2)2      -2.336e+00  1.928e-01  -12.114 < 2e-16 ***
## poly(odometer, 2)1 -7.853e+00  3.201e-01  -24.535 < 2e-16 ***
## poly(odometer, 2)2  8.533e-01  1.976e-01   4.319 1.57e-05 ***
## poly(mmr, 2)1        6.311e+02  2.471e-01 2553.583 < 2e-16 ***
## poly(mmr, 2)2      -3.148e+00  2.055e-01  -15.318 < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1809 on 409363 degrees of freedom
## Multiple R-squared:  0.9675, Adjusted R-squared:  0.9675
## F-statistic: 2.028e+06 on 6 and 409363 DF,  p-value: < 2.2e-16

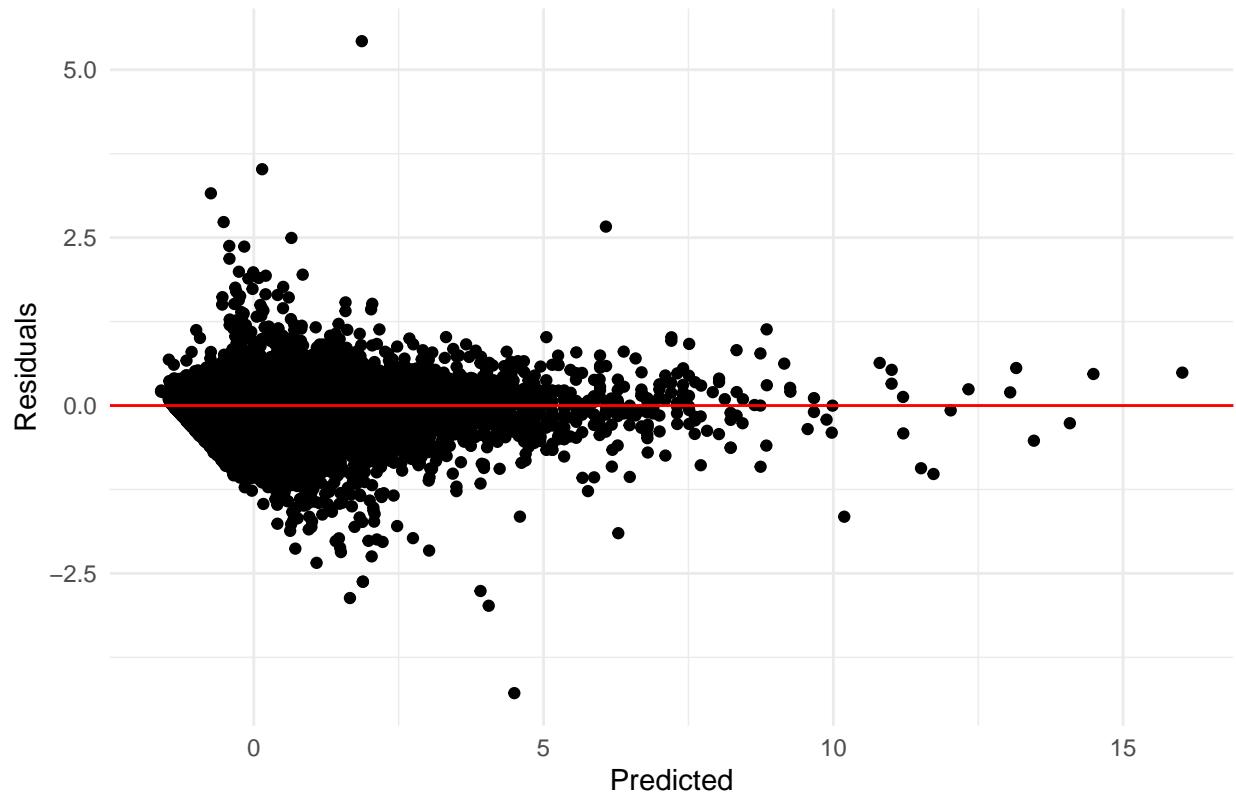
## Polynomial Regression RMSE: 0.1749021

## Polynomial Regression R-squared: 0.9674528

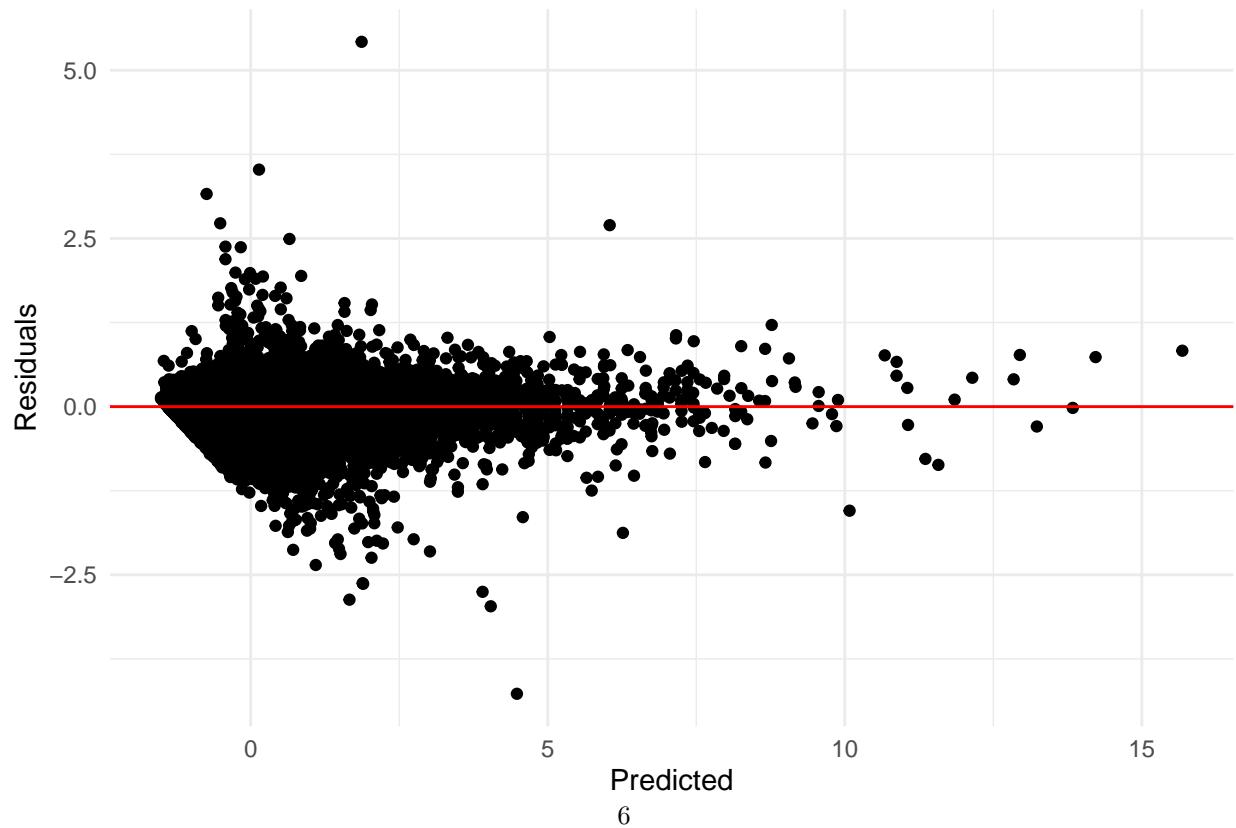
```

## Model Comparison

Linear Regression Residuals Plot



Polynomial Regression Residuals Plot



## **Conclusion**

- Given that the RMSE for the linear regression and the R-squared are very close but slightly better than the polynomial regression, the linear model is likely the better choice. It's simpler, less prone to overfitting, and has better predictive performance based on RMSE