Used Car Price Predictions

Reese Bottorff, rbottorff@bellarmine.edu

I. INTRODUCTION

I will be using a car dataset from Kaggle. The dataset will be helping me predict the price of the used car. I will be using linear regression to find this price. In this project, I will be going over the background, Data processing methods, Visuals, Splitting the data, Training and Testing the data, and Predictions.

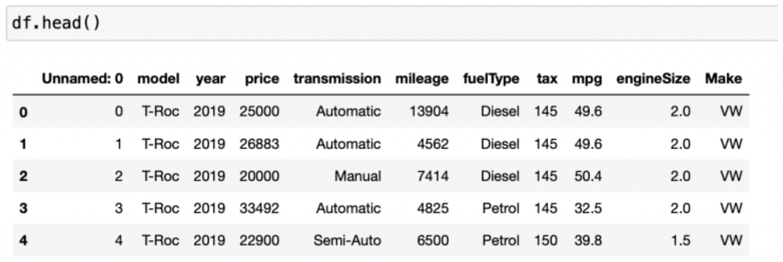
II. BACKGROUND

Using this data, I will create a predictive model to accurately estimate the selling price of a used car based on its features.

The columns in this dataset are:

Model	name of the car models
Year	the year the car was manufactured
Price	the price of the car
Transmission	the type of transmission
Mileage	the number of miles the car has been driven
FuelType	type of fuel the car uses
Tax	the amount of tax that is due on the car
Mpg	the miles per gallon that the car gets
EngineSize	the size of the car's engine in
Make	the manufacturer of the car

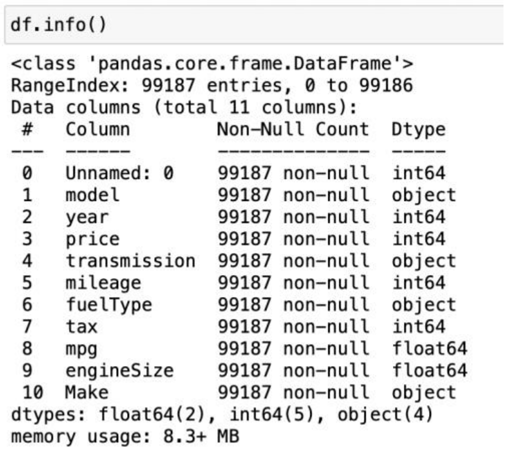
III. EXPLORATORY ANALYSIS



```
df.head()
```

	Unnamed: 0	model	year	price	transmission	mileage	fuelType	tax	mpg	engineSize	Make
0	0	T-Roc	2019	25000	Automatic	13904	Diesel	145	49.6	2.0	VW
1	1	T-Roc	2019	26883	Automatic	4562	Diesel	145	49.6	2.0	VW
2	2	T-Roc	2019	20000	Manual	7414	Diesel	145	50.4	2.0	VW
3	3	T-Roc	2019	33492	Automatic	4825	Petrol	145	32.5	2.0	VW
4	4	T-Roc	2019	22900	Semi-Auto	6500	Petrol	150	39.8	1.5	VW

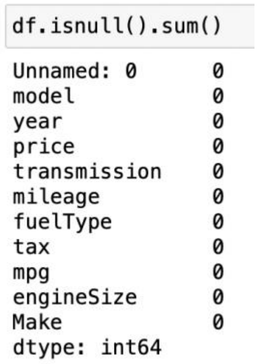
This table shows what the dataset looks like before changing anything



```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 99187 entries, 0 to 99186
Data columns (total 11 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Unnamed: 0    99187 non-null  int64
 1   model         99187 non-null  object
 2   year          99187 non-null  int64
 3   price         99187 non-null  int64
 4   transmission  99187 non-null  object
 5   mileage       99187 non-null  int64
 6   fuelType      99187 non-null  object
 7   tax           99187 non-null  int64
 8   mpg           99187 non-null  float64
 9   engineSize    99187 non-null  float64
 10  Make          99187 non-null  object
dtypes: float64(2), int64(5), object(4)
memory usage: 8.3+ MB
```

This table shows the type each column is. There are multiple categorical variables, therefore I need to change them to int.



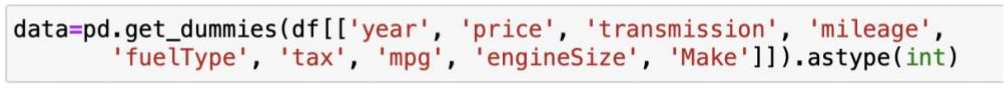
```
df.isnull().sum()
```

```
Unnamed: 0      0
model           0
year            0
price           0
transmission    0
mileage         0
fuelType        0
tax             0
mpg             0
engineSize      0
Make            0
dtype: int64
```

This table shows that there are no missing variables therefore I won't have to fill in any data before starting my training and testing.

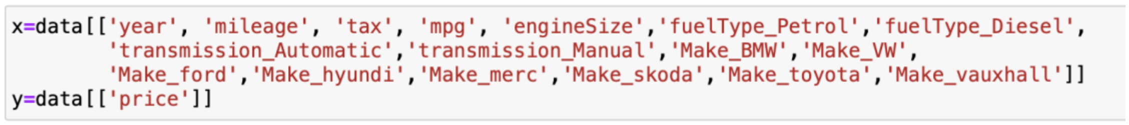
IV. METHODS

A. Data Preparation



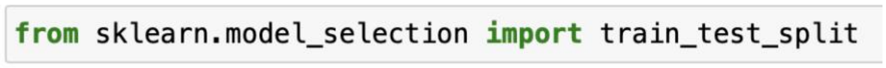
```
data=pd.get_dummies(df[['year', 'price', 'transmission', 'mileage',
        'fuelType', 'tax', 'mpg', 'engineSize', 'Make']]).astype(int)
```

I needed to use pd.get_dummies to change the categorical datas into integers so I could continue my model.

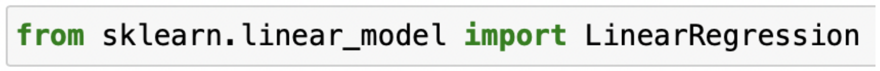


```
x=data[['year', 'mileage', 'tax', 'mpg', 'engineSize','fuelType_Petrol','fuelType_Diesel',
        'transmission_Automatic','transmission_Manual','Make_BMW','Make_VW',
        'Make_ford','Make_hyundi','Make_merc','Make_skoda','Make_toyota','Make_vauxhall']]
y=data[['price']]
```

After changing the types, I created independent and dependent variables, making price the dependent because that is what I will be predicting.

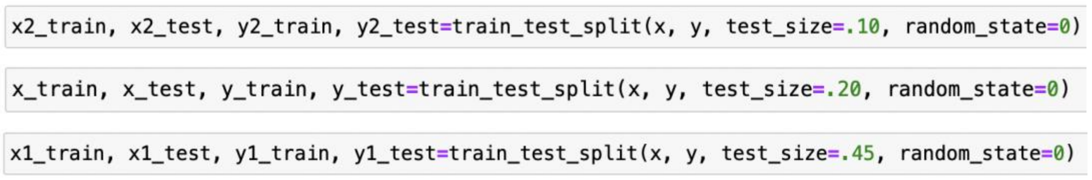


```
from sklearn.model_selection import train_test_split
```



```
from sklearn.linear_model import LinearRegression
```

These packages were used in my model to split, train, and test the dataset.



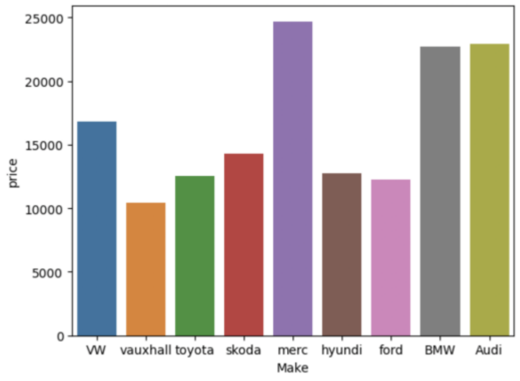
```
x2_train, x2_test, y2_train, y2_test=train_test_split(x, y, test_size=.10, random_state=0)
```

```
x_train, x_test, y_train, y_test=train_test_split(x, y, test_size=.20, random_state=0)
```

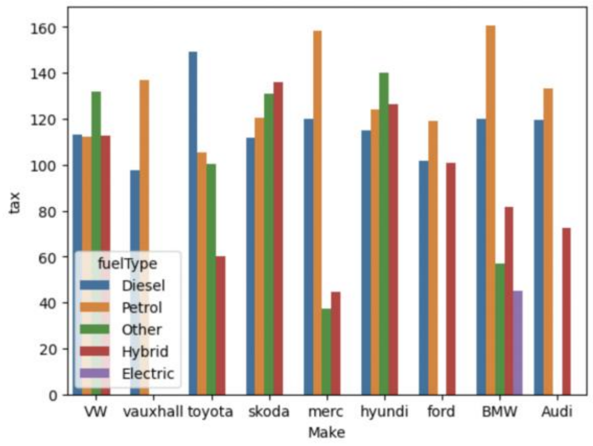
```
x1_train, x1_test, y1_train, y1_test=train_test_split(x, y, test_size=.45, random_state=0)
```

I split the dataset into 3 test sizes to see which one had the greatest accuracy. They all had around the same accuracy, but the 45% split had the biggest at 75% accuracy. Because of this, I used the 45% split for my predictions.

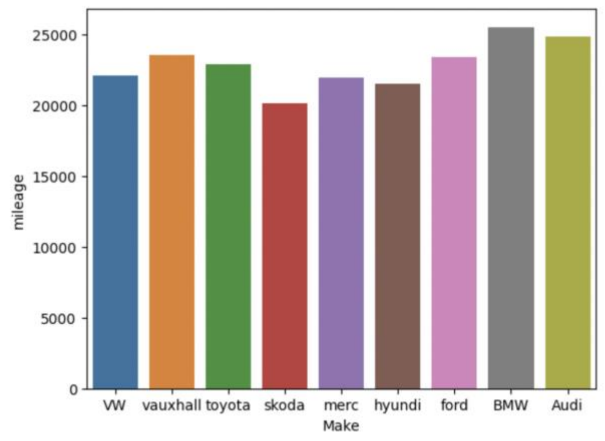
B. *Experimental Design*



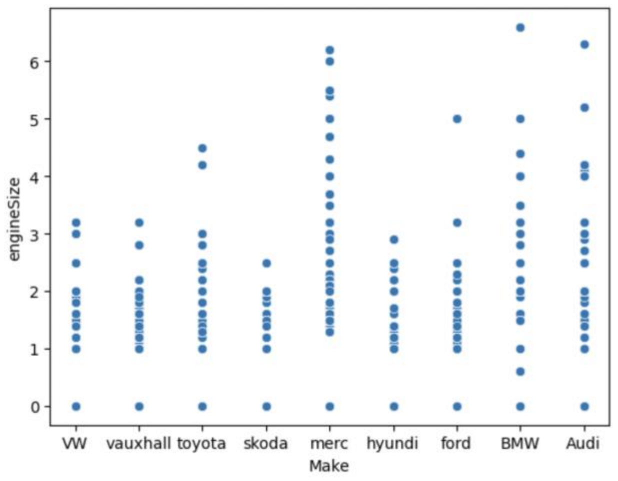
This graph is comparing the make of the car and the price. This shows that the Mercedes has the highest price and Vauxhall has the lowest price.



This graph is comparing the make of the car and the tax of the car with the hue as the fueltype. The results show a consistency of high tax with petrol as the fueltype and a consistency of low tax with hybrid as the fueltype.



This graph is comparing the make of the car and the mileage used. The results show that every make is over 20,000 miles except Skoda. This also shows that BMW has the most miles used.



This graph compares the make of the car and the engine size. The results show that the Mercedes has the widest variety of engine sizes but BMW has the biggest engine

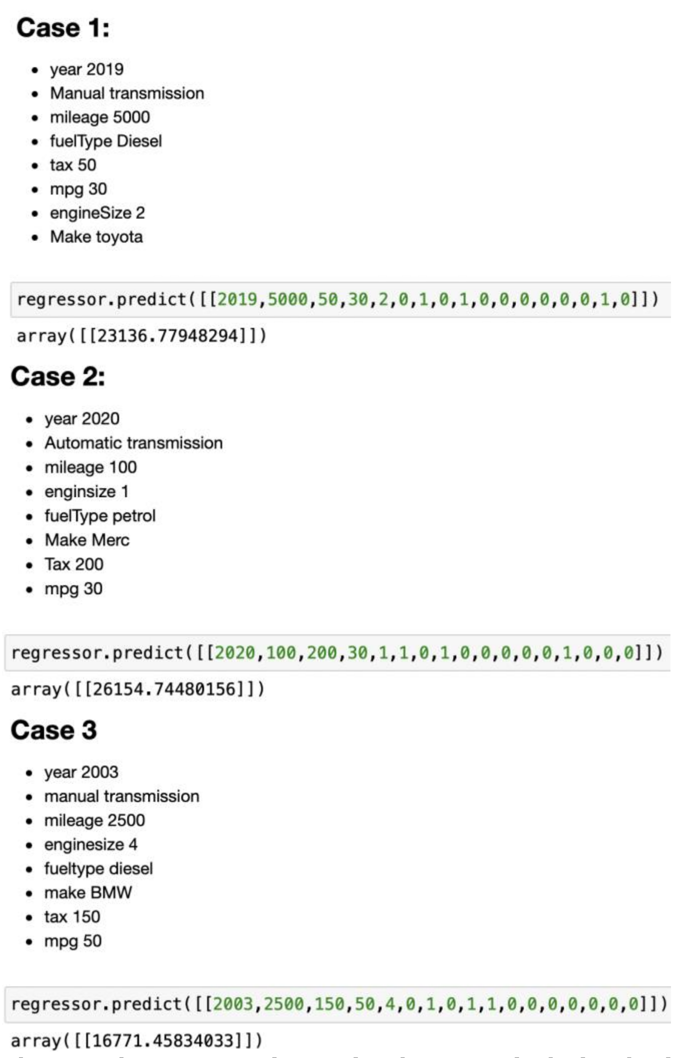
C. Tools Used

The following tools were used for this analysis: Python v3.5.2 running the Anaconda 4.3.22 environment for Apple Macintosh computer was used for all analysis and implementation. In addition to base Python, the following libraries were also used: Pandas 0.18.1, Numpy 1.11.3, Matplotlib 1.5.3, Seaborn 0.7.1, SKLearn 0.18.1, and Patsy 0.41.

These tools were used to help me complete this model. Without these tools, I wouldn’t be able to research and break down the dataset to complete this model.

V. RESULTS

A. Predictions



Case 1:

- year 2019
- Manual transmission
- mileage 5000
- fuelType Diesel
- tax 50
- mpg 30
- engineSize 2
- Make toyota

```
regressor.predict([[2019,5000,50,30,2,0,1,0,1,0,0,0,0,0,0,1,0]])
```

```
array([[23136.77948294]])
```

Case 2:

- year 2020
- Automatic transmission
- mileage 100
- enginsize 1
- fuelType petrol
- Make Merc
- Tax 200
- mpg 30

```
regressor.predict([[2020,100,200,30,1,1,0,1,0,0,0,0,0,1,0,0,0]])
```

```
array([[26154.74480156]])
```

Case 3

- year 2003
- manual transmission
- mileage 2500
- enginesize 4
- fueltype diesel
- make BMW
- tax 150
- mpg 50

```
regressor.predict([[2003,2500,150,50,4,0,1,0,1,1,0,0,0,0,0,0,0]])
```

```
array([[16771.45834033]])
```

These are the cases I used to predict the price. I looked at the different features and randomly picked specific features to try and predict the price. As you can see, the first case has a predicted price of $23,136, the second case has a predicted price of $26,154, and the third case has a predicted price of $12,771.

B. Discussion of Results

I think my model was good because it gave a lot of information to work with. That being said, I think it also made it hard to work with. Because there was so much data, I had to delete a column because of how big it make the dataset.

C. Problems Encountered

I faced a couple problems when completing my model. Having to delete the Model column was a problem for me because it was a big factor in my dataset. I had to figure out how to complete the model without one of the main features. Another problem I faced was changing the categorical variables. When I changed them, I had to then write out every sub column (example: Make turned into make_BMW, make_toyota...). It was a very time consuming process. The last problem I faced was finding test sizes that had different accuracy percentages. Most of the numbers I tested out had the same results.

D. Limitations of Implementation

Like I said before, this dataset was very big and had a big variety, therefore it was difficult to work with and needed things to be deleted. Working with a smaller dataset would have been easier and less complicated.

E. Improvements/Future Work

Improvements would include finding a smaller dataset that allows me to work without needing to delete or change any data or columns. Also finding a dataset that interests me would be better because I would be more intrigued by the results

VI. CONCLUSION

Overall, the dataset I used was a very interesting dataset for linear regression. It allowed me to dive into the car world and see how much used cars cost. I found it interesting because the different features didn't make as much of a difference in the price as I thought they would. The 3 predictions I made were close to the same price, even though I used very different features. To make the predictions, I needed to delete the model column because it was too large and had too big of a variety. I also had to change the categorical data into integers before training and testing the data. I thought this linear regression model was easy to read and found this very interesting.

REFERENCES

- https://www.kaggle.com/datasets/harishkumardatalab/used-car-prediction-dataset (Kaggle dataset)
- https://github.com/Reeseab03/Final_LR_RAB/blob/main/Final_LR_RAB.ipynb (Github notebook)