Supervised Learning: Regression on UK Used Car Data Set

NARIMAN PASHAYEV

Main Objectives

- The main objective of this analysis is to predict price(£) of used Ford cars using a Linear Regression and different regularization regressions.

- This analysis attempts to try both train-test-split and cross-validation to have an overview of how these two methods can lead to different decisions in terms of model selection.

- Data Source: Ford Data set from UK used car data set

About the Data

- The data set used in this analysis is a part of 100,000 UK Used Car Data Set published on Kaggle in July 2020 by a member (Aditya).
- The author scraped the data from 100,000 listings, which have been separated into files corresponding to each car manufacturer
- The cleaned data set contains information of price, transmission, mileage, fuel type, road tax, miles per gallon (mpg), and engine size.
- Duplicate listings removed and cleaned the columns
- The cleaned data were then separated into .csv files corresponding with each car manufacturer.
- The Ford data set was selected for this analysis. This data set has 17,965 records and 9 variables. During the analysis, some duplicates were detected and removed, and also there was a row which car year was 2060, so this row was removed as well: remaining 17,810 records.

Variable Name	Type	Description
Model	String	Model of car
Year	integer	Manufacture year
Price	Integer	Selling price
Transmission	String	Transmission type
Mileage	Integer	Car mileage
Fuel type	Integer	Fuel type
Tax	Integer	Current tax
MPG	Float	Miles per gallon
Engine Size	float	Size of car engine

	model	year	price	transmission	mileage	fuelType	tax	mpg	engineSize
0	Fiesta	2017	12000	Automatic	15944	Petrol	150	57.7	1.0
1	Focus	2018	14000	Manual	9083	Petrol	150	57.7	1.0
2	Focus	2017	13000	Manual	12456	Petrol	150	57.7	1.0
3	Fiesta	2019	17500	Manual	10460	Petrol	145	40.3	1.5
4	Fiesta	2019	16500	Automatic	1482	Petrol	145	48.7	1.0
...	...	...	...	...	...	...	...	...	...
17960	Fiesta	2016	7999	Manual	31348	Petrol	125	54.3	1.2
17961	B-MAX	2017	8999	Manual	16700	Petrol	150	47.1	1.4
17962	B-MAX	2014	7499	Manual	40700	Petrol	30	57.7	1.0
17963	Focus	2015	9999	Manual	7010	Diesel	20	67.3	1.6
17964	KA	2018	8299	Manual	5007	Petrol	145	57.7	1.2

Data Exploration

- After removing the duplicates and 1 non-realistic row (year =2060), Exploratory Data Analysis was carried out on the data set
- Total 17810 rows left after removing
- All of the 9 columns: 4 columns are integers, 3 are string and only 2 columns are float.
- There are 23 unique models, 3 unique transmission types and 5 fuel types in the set

```
data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 17810 entries, 0 to 17809
Data columns (total 9 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   model         17810 non-null  object
 1   price         17810 non-null  int64
 2   transmission  17810 non-null  object
 3   mileage       17810 non-null  int64
 4   fuelType      17810 non-null  object
 5   tax           17810 non-null  int64
 6   mpg           17810 non-null  float64
 7   engineSize    17810 non-null  float64
 8   age           17810 non-null  int64
dtypes: float64(2), int64(4), object(3)
memory usage: 1.2+ MB
```

```
data.dtypes.value_counts()

int64      4
object     3
float64    2
dtype: int64
```

```
data_object=data.columns[data.dtypes==object].to_list()
data[data_object].nunique()

model           23
transmission     3
fuelType         5
dtype: int64
```

Data Exploration

- Then basic statistics obtained of the both categorical and numerical data
- Among the all of the model types, Fiesta is the most sold one and total 6508 Fiestas sold
- Manual transmission is the most preferred transmission type and total 15382 cars sold with manual transmission
- Out of the 17810 cars, 12079 are using petrol as a fuel type
- Also in this data set, year column is replaced with Age column

```
data.describe()
```

	year	price	mileage	tax	mpg	engineSize
count	17810.000000	17810.000000	17810.000000	17810.000000	17810.000000	17810.000000
mean	2016.860079	12270.103481	23380.413532	113.314992	57.909556	1.350640
std	2.026487	4736.260216	19418.185474	62.030508	10.132632	0.432597
min	1996.000000	495.000000	1.000000	0.000000	20.800000	0.000000
25%	2016.000000	8999.000000	10000.000000	30.000000	52.300000	1.000000
50%	2017.000000	11289.500000	18277.000000	145.000000	58.900000	1.200000
75%	2018.000000	15295.000000	31095.250000	145.000000	65.700000	1.500000
max	2020.000000	54995.000000	177644.000000	580.000000	201.800000	5.000000

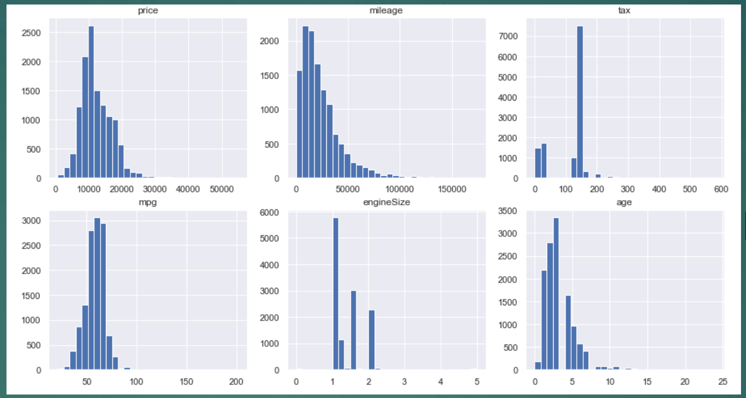
```
data.describe(include=[object])
```

	model	transmission	fuelType
count	17810	17810	17810
unique	23	3	5
top	Fiesta	Manual	Petrol
freq	6508	15382	12079

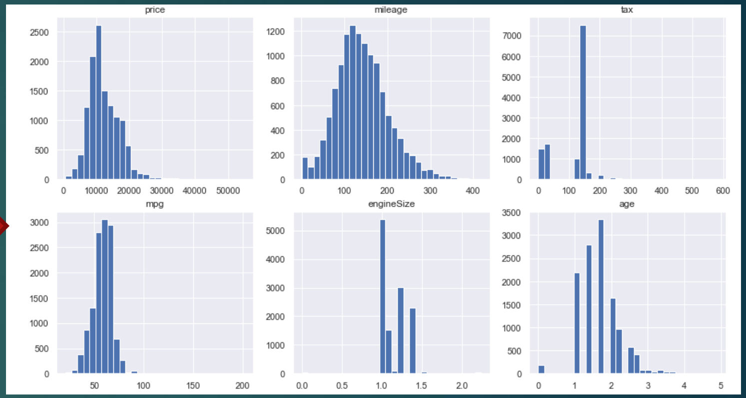
Data Exploration-Determining Skewed features

- Data split into train (70%) and test (30%) sets
- Skew analysis done on numerical values and it seems there is some skewness in the dataset
- Skew limit>0.75
- **SQRT** transformation applied on the both train and test dataset in order to eliminate the skewness, but the target value (price) kept unchanged

Histogram before transformation



Histogram after transformation



	Train_Skew
age	1.861308
mileage	1.823722
engineSize	1.806635
price	1.143463

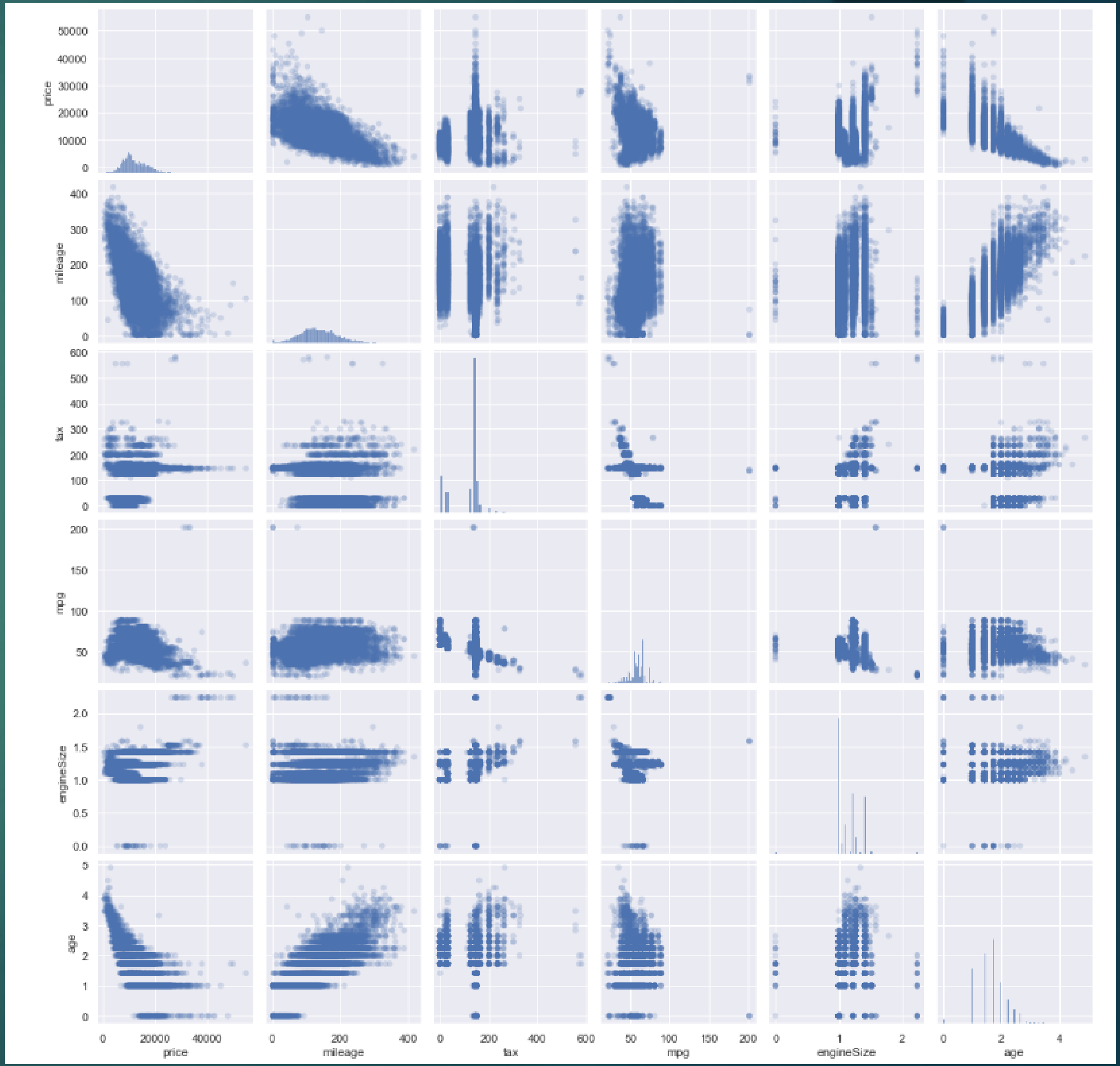
	Train_Skew
price	1.143463
mpg	0.716861
mileage	0.477056
age	0.319815
engineSize	0.233153
tax	-0.594362

	Test_Skew
engineSize	2.102190
age	1.884772
mileage	1.843429
price	0.984616

	Test_Skew
price	0.984616
mileage	0.486235
mpg	0.345026
age	0.320344
engineSize	0.273678
tax	-0.482870

Data Exploration- Pair plot of the features

- As a next step, a pair plot was created of the SQRT transformed values to have an overview of the features and the target
- This plot shows that:
  - age has a linear relationship with price. It looks quite like polynomial.
  - mileage also has linear relationship with price.
  - age also has a linear relationship with mileage (the older the more miles). This is multicollinearity.

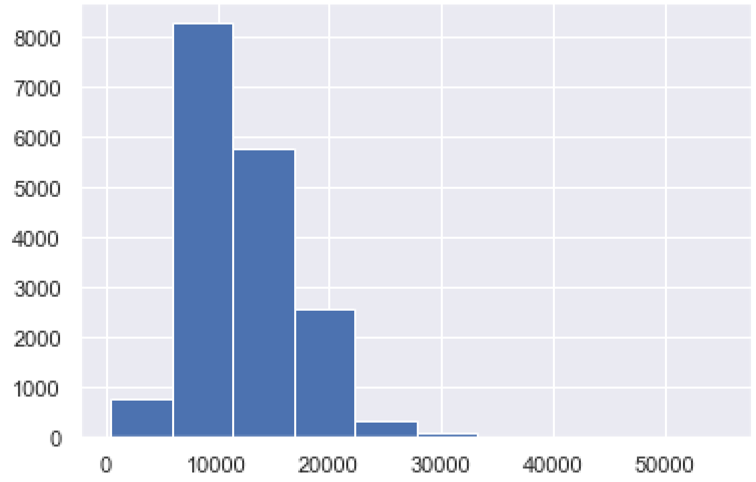


Data Exploration- Determining Normality of Target Variable

- Making our target variable normally distributed often will lead to better results
- If our target is not normally distributed, we can apply a transformation to it (log, square root, boxcox) and then fit our regression to predict the transformed values.
- How can we tell if our target is normally distributed? There are two ways:
  - Visually
  - Using a statistical test
- pvalue=0. so normal distribution. no need any transformation on target variable

```
# Visually
Y.hist()
```

```
<AxesSubplot:>
```

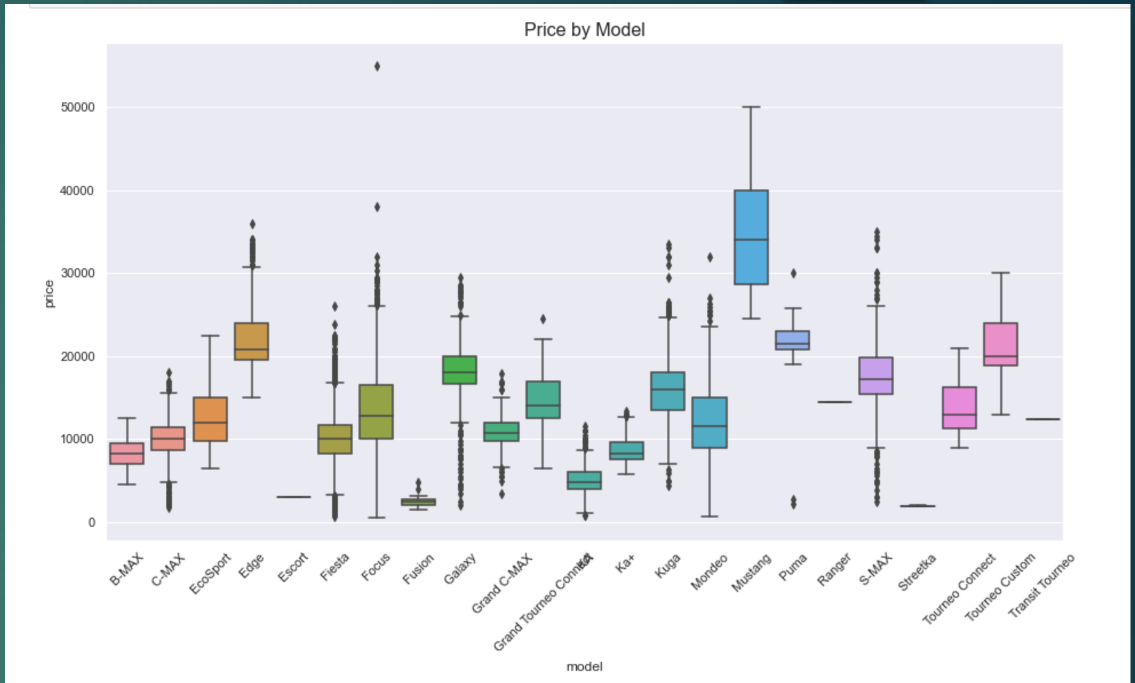


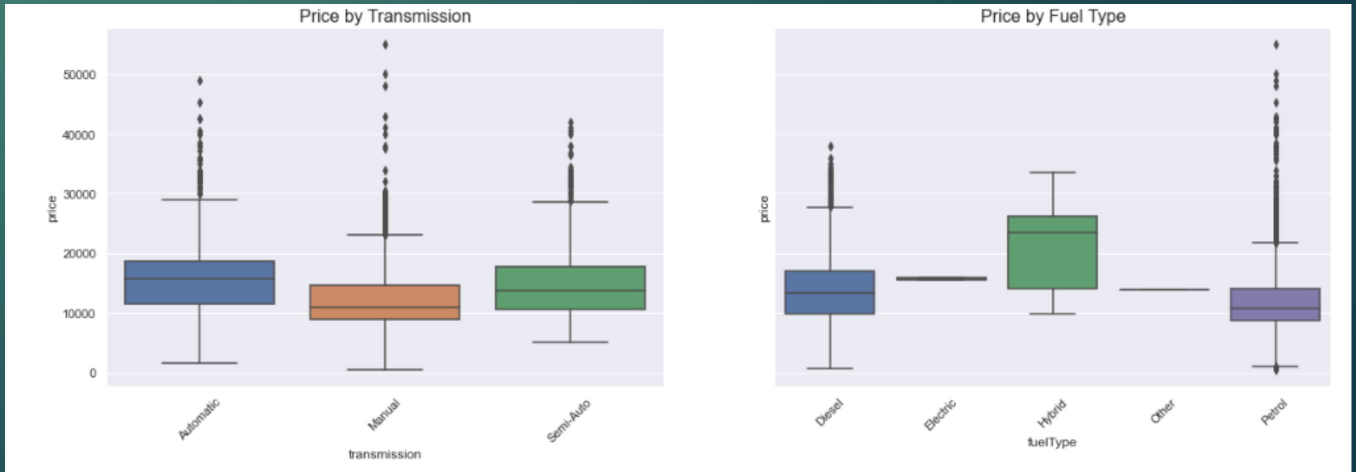

```
normaltest(Y.values)
```

```
NormaltestResult(statistic=3788.026421979386, pvalue=0.0)
```

Data Exploration- Box Plot

- Box plot of 3 categorical variables was created
- On average, car prices vary among models, transmission and fuel types
- Hybrid cars are most expensive ones compared to other fuel type cars
- On average, manual transmission cars are cheaper than automatic and semi-auto cars





Feature Engineering-Encoding and Scaling

- Feature engineering is applied in order to create model variations.
- Each model is evaluated based on its root mean square error and R2_score
- As mentioned in above slides, numerical features are transformed using SQRT transformation that have a skew value>0.75
- Firstly, plain Linear regression without any polynomial feature engineering was evaluated on 4 model variations:
  - Linear regsession without one-hot encoding and scaling
  - Linear regression without one-hot encoding, but with scaled version
  - Linear regression with one-hot encoding, but without scaling
  - Linear regression with one-hot encoding and with scaling version

	Model	num_features	RMSE	R2_Score
0	LR_ohc no scaling	33	1663.270151	0.873157
0	LR_ohc scaling	33	1663.270151	0.873157
0	LR_no_ohc no scaling	5	2390.032955	0.738092
0	LR_no_ohc with scaling	5	2390.032955	0.738092

- It's seen that one-hot encoding clearly increases R2 score and decreases error values. From now, I will be using the encoded features for future analysis and modelling
- But scaling on plain vanilla linear regression has no effect. But it clearly effects ridge and Lasso regression results, which we will see later

Feature Engineering-Polynomial Features

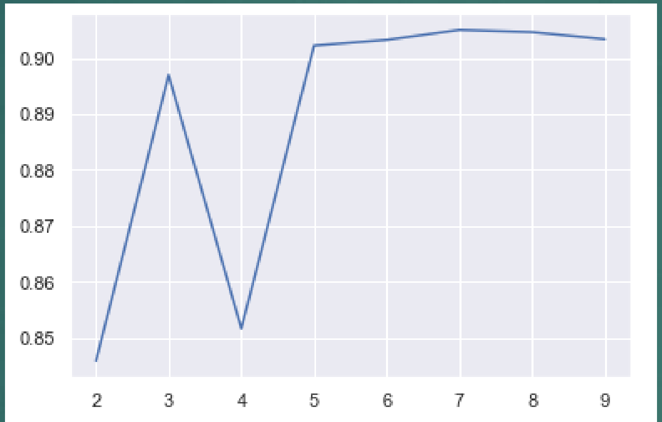
- Polynomial feature engineering carried out on the encoded version of the variables
- Here I created polynomialfeatures on the floats only, excluding the one hot encoded columns, and then combined the new polynomial features with one hot encoded columns to create new dataframe
- After that, I applied Vanilla Linear Regression on this new dataframe

Number of features	RMSE	R2_Score	PF Degree
33.0	1783.886599	0.830417	1.0
48.0	1546.725253	0.879034	2.0
83.0	1604.875571	0.872347	3.0
153.0	2340.364944	0.715962	4.0
279.0	7149.831697	0.024112	5.0
489.0	5672.504103	-0.153786	6.0

- It seems polynomial **'degree=2'** gives best **RMSE=1546.72** and **R2_score=0.879034**.
- It is better than previous **LR_ohc scaling** model, which gave **RMSE=1663.27** and **R2=0.873**

Cross-validation and Regularization-Defining Kfold splits

- So far, following pipeline created on the features:
  - One-hot encoding → SRTQ transformation → polynomial features → Vanilla Linear regression
- Data split carried out using KFold to define the best split number by using GridSearchCV




```
[0.8458331023702216,
 0.8970081003193395,
 0.8516323804828236,
 0.902309156796415,
 0.9033313951176059,
 0.9051166548580392,
 0.9047236676900003,
 0.9034623266185913]
```

- K=5 splits seems the optimum one. So, from now, I will be using k=5 in all regressions

Cross-validation and Regularization

- GridSearchCV cross-validation with k=5 folds used to fit the linear regression model on full data set, and then attempt to tune the hyperparameter to find a proper combination of alpha and polynomial degree for regularization
- Iterated over different polynomial degree (1, 2, 3) and alphas.
- Regularized models include Lasso, Ridge, and Elastic Net
- After finding the optimized hyperparameters with GridSearchCV, then same pipeline with tuned hyperparameters used in order to predict the results
- Each model is evaluated based on its average root mean squared error and R2_score
- All 4 models prediction results (RMSE and R2_score) seems pretty close to each other
- But, Vanilla Linear Regression is a little bit better than the others

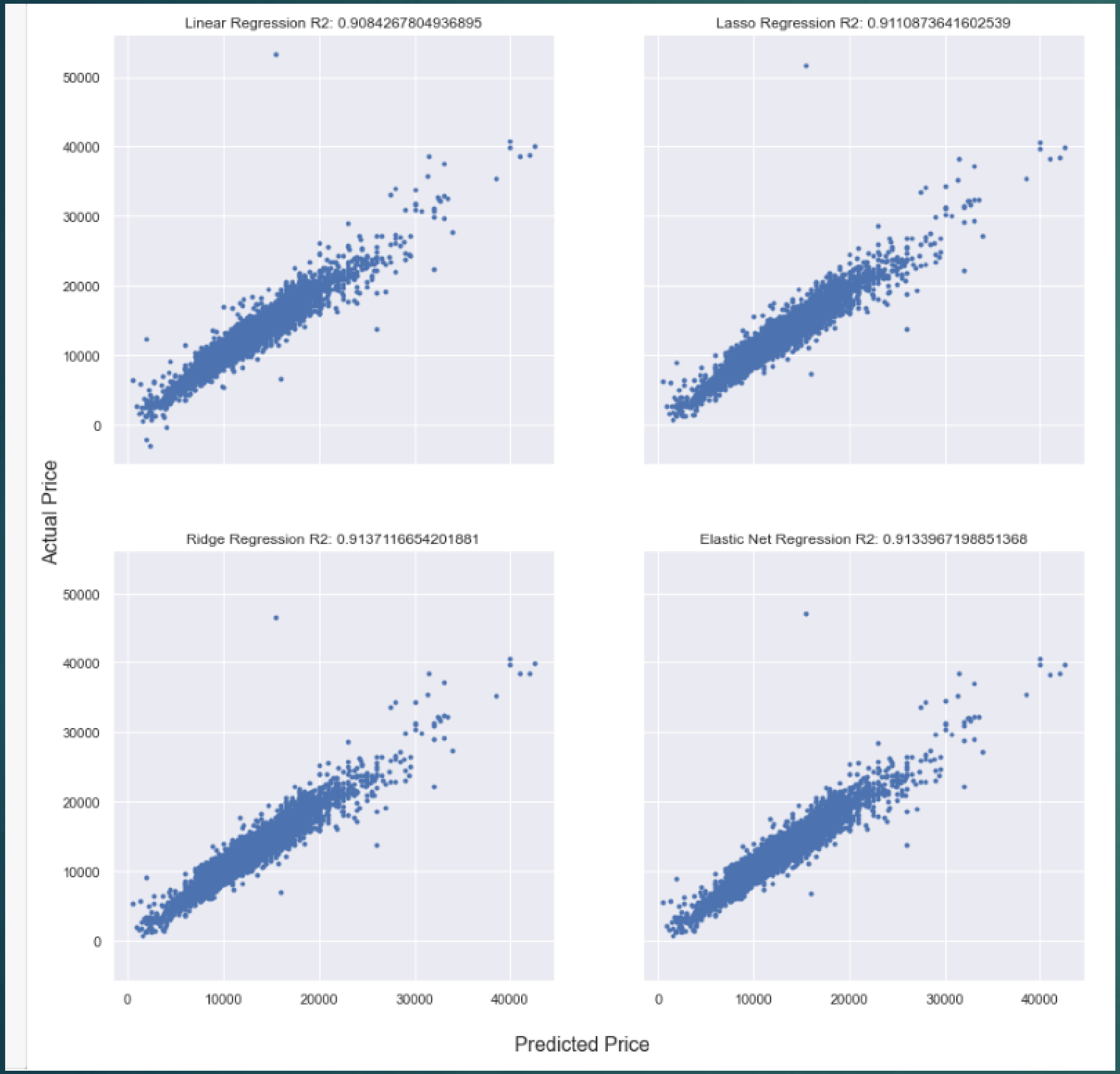
Model	RMSE	R2_Score
Vanilla LinearRegresson	1346.431386	0.919179
Lasso LinearRegresson	1357.517804	0.917843
Ridge LinearRegresson	1355.975876	0.918030
ElasticNet LinearRegresson	1358.787571	0.917689

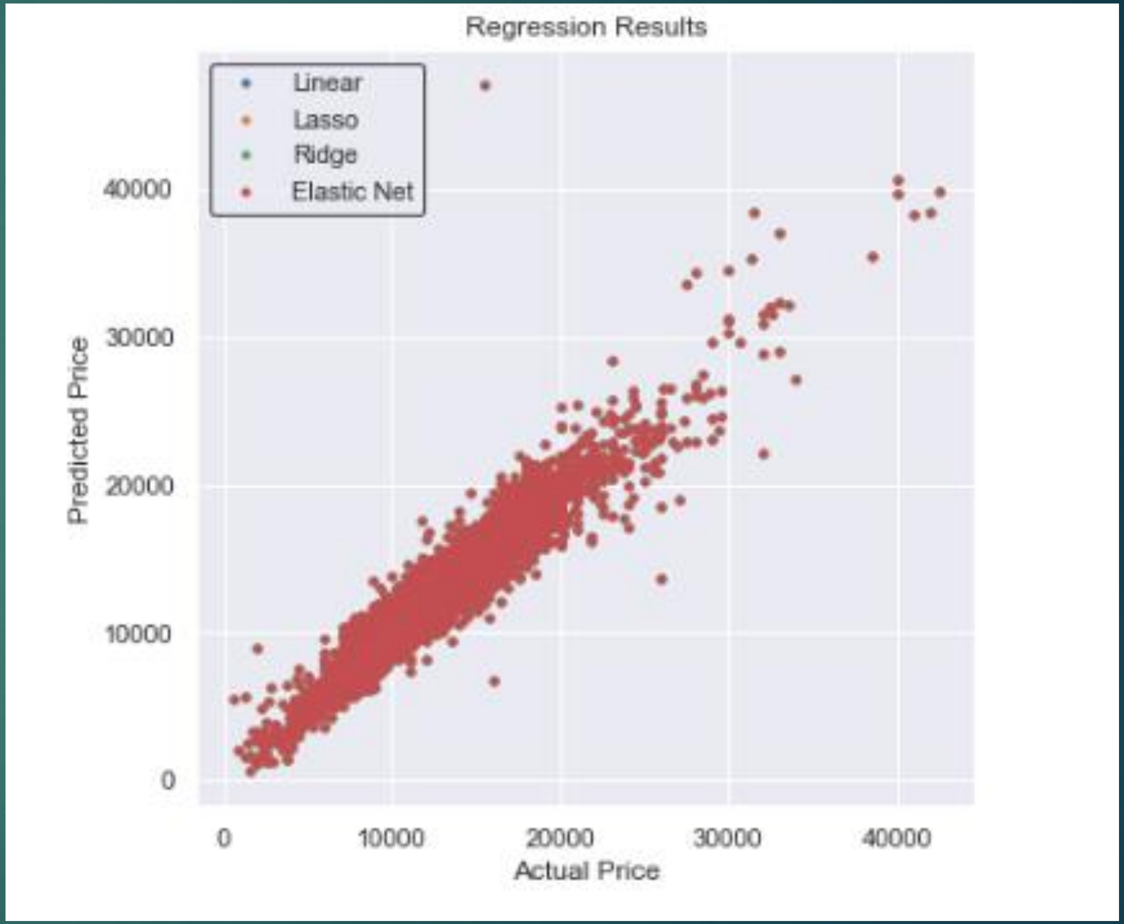
Cross-validation and Regularization-Prediction on Unseen Data

- Four models fit on the train set and then predicted on the unseen test set and calculated the R2 score for each model.
  - Linear regression with 2nd degree polynomial features
  - Lasso regression with 2nd degree polynomial features and alpha = 0.85
  - Ridge regression with 2nd degree polynomial features and alpha = 12.32
  - Elastic Net regression with 2nd degree polynomial features and alpha = 0.01 and l1_ratio=0.9
- Ridge Regression has the best prediction on the test set. All these models can explain the target around 90% - 91%

Model	RMSE	R2_Score
Elastic Net Regresson	1374.348015	0.913397
Ridge LinearRegresson	1371.846729	0.913712
Lasso LinearRegresson	1392.551583	0.911087
Vanilla LinearRegresson	1413.233066	0.908427

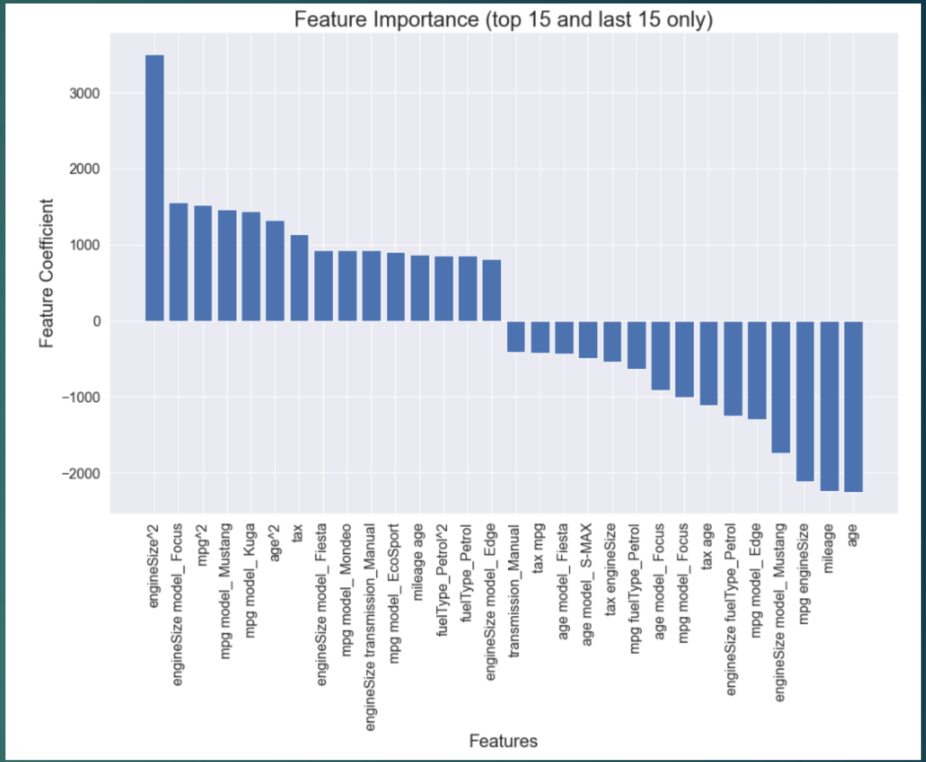
Scatter plots (true vs predicted price) and R2 scores on the unseen data.





Feature Importance

- Ridge regression eliminated totally 322 rows
- The main drivers of this model are:
  - Engine Size
  - Mpg
  - Age or model year
  - tax



406	model_ Grand Tourneo Connect model_ KA	0.0
409	model_ Grand Tourneo Connect model_ Mondeo	0.0
408	model_ Grand Tourneo Connect model_ Kuga	0.0
334	model_ Focus model_ Tourneo Custom	0.0
407	model_ Grand Tourneo Connect model_ Ka+	0.0
...	...	...
263	model_ Edge model_ Transit Tourneo	0.0
285	model_ Escort model_ Streetka	0.0
284	model_ Escort model_ S-MAX	0.0
282	model_ Escort model_ Puma	0.0
283	model_ Escort model_ Ranger	0.0

322 rows × 2 columns

Conclusion

- It's seen that one-hot encoding clearly increases R2 score and decreases error values
- But scaling on plain vanilla linear regression has no effect. But it clearly effects ridge and Lasso regression results
- Polynomial feature engineering with degree=2 works best for this data set
- All 4 models prediction results (RMSE and R2_score) seems pretty close to each other
- But, Ridge Regression performs a little bit better than the others
- All these models can explain the target around 90% - 91%
- Engine size, mpg, age and tax are the main drivers of the model
- In this work, only Linear Regression analysis was used. It would be better to try other methods as well like, classification methods
- My Jupyter Notebook can be found here:

https://github.com/NARIMANPASHA/Supervised_Learning-Regression-on-UK-USed-Car-Data-Set.git