Lorcan O′Mahony 20003866

Building a price prediction model for used cars in Ireland using a combination of machine learning and sentiment analysis

Introduction
Car companies and dealerships rely partially on subjective assessment and arbitrary figures when setting the prices of their car, this is irrational and leads to cars that are priced incorrectly.
Some cars endup overpriced some end up underpriced. This way of doing things is chaotic and inefficient. Fortunately machine learning could potentially lead to a superior way of setting prices.
Consumer also need to be ensure that they are buying at a fair price. Costomers would be more likely to buy if they know they are buying at a price that reflected the value of the cars instead of having to trust the car dealership.
Car dealerships and car companies need to be sure that they are selling their cars at a fair price so they do not sell their cars too cheaply or too expensively.
A machine learning prediction model solves this problem
it uses data to predict the price of cars based on various factors such as make, model, mileage, economic factors reviews, public sentiment year etc.

Literature review

During my review of the literature. I could not find any research that attempts to do exactly what I aim to do in my research project, but there has been lots of research published where people try to accomplish similar goals. In my review of this literature I have learned many things that will help me in my research.
The main lesson I learned from my review of the literature on using AI and machine learning to predict car prices is that there is not one approach that is consistently better than others, different machine learning methods were best for different people in different situations
For almost every piece of literature I read the most accurate model was different
The following are some examples of different people coming to different conclusions on what is the best way to predict car prices.
Kavitha et al. (2019) found that the decision tree classifier was the most effective method for predicting car price, they had an accuracy score of 96.5% while attempting to determine how much a car should cost based on various factors such as price, safety, and how luxurious or spacious the car is
Kalpana et al. (2022) found that the linear regression algorithm performed better than the random forest algorithm in their study
and for Shaprapawad, Borugadda, and Koshika (2023) the support vector classifier was the most effective method '' The support vector regressor is the optimal model based on the evaluation metrics such as R
Squared (R^2)of 95.27 %, Mean Absolute Error (MAE) of
0.142, Mean Squared Error (MSE) of 0.047, and Root Mean
Squared Error (RMSE) of 0.218 '
In a similar experiment Putra et al. (2023) had an accuracy of 72.13% when using the random forest method and 67.21% when using the decision tree method when predicting car prices
This illustrates the fact there is no perfect prediction method and the best method is dependant on many different variables.
When predicting the prices of used cars in India, Varshitha, Jahnavi, and Lakshmi (2022) used deep neural networks, linear regression, lasso regression, ridge regression and random forest.
Random forest was the most accurate
'Forest model with a Mean Absolute Error value of
1.0970472 and R2 error value of 0.772584 has given the less error
among all the other algorithms'

Varshitha, Jahnavi, and Lakshmi (2022)
Sharma et al.(2023) had an accuracy of 92.35%with the random forest classifier in a similar study by
Most of this research has one thing in common and that is the evaluation metric they used. Accuracy was the main evaluation metric used to evaluate all of these predictions. The main reason accuracy is chosen is simply because it is common and therefore is the default choice for many researchers. 'Accuracy is most common evaluation metric for classification problems' Kavitha et al. (2019)
The reason accuracy is chosen over other methods is that there is no significant difference between false positives or false negatives in this type of task. An overestimation or underestimation of the price is equally bad.

AlShared (2021) carried out a project similar to mine while creating a prediction model for used car prices in the United Arab Emirates. One of the most useful things I learned from reading this paper is the data collection tools that were used, AlShared scraped data from used car websites using a tool called parsehub

"ParseHub is a free and powerful web scraping tool. With our advanced web scraper, extracting data is as easy as clicking on the data you need." (Parsehub 2025)

Parsehub is a far quicker and user friendly way to scrape data from websites without having to write code.

In order to carry out data analytics and machine learning projects, data must be collected and this often requires web scraping, which is the process of creating software that collects data from websites

"Web scraping or web crawling refers to the procedure of automatic extraction of data from websites using software. It is a process that is particularly important in fields such as Business Intelligence in the modern age."
(Khder 2021)

The reason parsehub is so useful is because it implements techniques to evade measures by the website to detect your webscraper and block it.

If you are scraping too much data from a website the website will detect an abnormal number of requests coming from your ip address and block you

"Websites employ various anti-scraping techniques to protect their data. These techniques range from simple measures like changing the HTML structure

frequently to more advanced methods like CAPTCHAs, rate limiting, IP blocking, and honeypots." (Choudhary & Arora, 2021)

For example, Parsehub allows users to rotate proxies, which means it changes the IP address many times during the scraping process in order to prevent detection

this is possible with standard python webscraping libraries but it is very complicated and it is far quicker to use parsehub

Data collection

Car Data

The data was collected from donedeal.ie which is an Irish website where people buy and sell used cars

I scraped over 11,000 car listings from the website using parsehub

Each listing had 44 columns that consisted of different attributes of each car

such as “year”, “fuel type”, “Mileage” etc

Social Media Data

In this project I performed sentiment analysis on social media data in order to classify the sentiment of social media posts and combine this with the car data to create a more robust car price prediction model

The social media data was scraped from reddit using python and reddits API

A list of every car model in the car dataset was used in the python script to iterate through each car model and search reddit for the model and scrape the first 5 posts

After observing the posts, I learned that a large proportion of the posts were car related questions asking for reddit’s opinion on the car model

This type of post is not useful for sentiment analysis

in order to solve this problem I also scraped the top comment of each post the top comment was usually an answer to the question asked in the post,

this top comment usually contained sentiment

Abstract

Sentiment analysis and machine learning models were successfully used to create a prediction model that predicts the price of used cars in Ireland.

Social media data was analyzed with sentiment analysis techniques and combined with data from the used car website, donedeal.com to create a machine learning model.

The most effective machine learning model used was the randomforestRegressor that achieved an r squared score of 0.897

The sentiment analysis data added a slight improvement to the models ability to predict prices which suggest that sentiment analysis of social media data has the potential to improve price prediction models of used cars

Data collection

Donedeal Dataset

The Data was scraped from donedeal.ie. Donedeal is the biggest used car marketplace website in Ireland

It is used by hundreds of thousands of Irish people every year

Private sellers and car dealerships list their cars for sale on the website, when they do this they input information about the car such as the mileage, trim, car model, year etc.

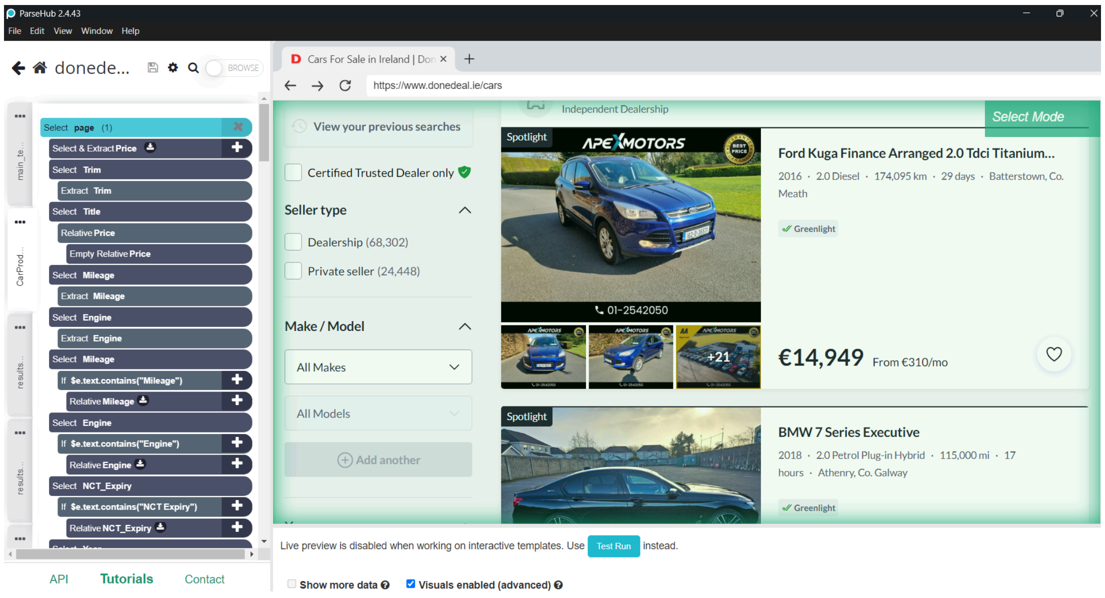
I scraped over 17,000 car listings from the website using a web-scraping tool called parsehub

I chose to use parsehub instead other web scraping tools like beautifulsoup and selenium because parsehub had built in proxy rotation and anti bot detection evasion methods.

This was necessary because I was scraping a large amount of data in a small amount of time from donedeal.ie and a basic python webscraper would have been detected and blocked by the website.

The final donedeal dataset consisted of 17192 rows and 44 columns

Below is an image of the parsehub web scraping tool


ParseHub 2.4.43
File Edit View Window Help
donede...
BROWSE
Cars For Sale in Ireland | Don
https://www.donedeal.ie/cars
Select page (1)
Select & Extract Price
Select Trim
Extract Trim
Select Title
Relative Price
Empty Relative Price
Select Mileage
Extract Mileage
Select Engine
Extract Engine
Select Mileage
If $e.text.contains("Mileage")
Relative Mileage
Select Engine
If $e.text.contains("Engine")
Relative Engine
Select NCT_Expiry
If $e.text.contains("NCT Expiry")
Relative NCT_Expiry
View your previous searches
Certified Trusted Dealer only
Seller type
Dealership (68,302)
Private seller (24,448)
Make / Model
All Makes
All Models
Add another
Independent Dealership
Select Mode
Spotlight
Ford Kuga Finance Arranged 2.0 Tdci Titanium...
2016 · 2.0 Diesel · 174,095 km · 29 days · Batterstown, Co. Meath
Greenlight
01-2542050
€14,949 From €310/mo
BMW 7 Series Executive
2018 · 2.0 Petrol Plug-in Hybrid · 115,000 mi · 17 hours · Athenry, Co. Galway
Live preview is disabled when working on interactive templates. Use Test Run instead.
Show more data
Visuals enabled (advanced)
API Tutorials Contact

Reddit Dataset

The goal here is to find the opinions of regular people on the car models to determine if they are generally positive or negative.

The theory behind this approach is that if a car has overwhelmingly positive sentiment in it's social media posts people like the car more and would be willing to pay more money for it therefore the price would be higher

And vice versa if the sentiment is bad people would be willing to pay less.

Therefore it is vital that the text used for sentiment analysis comes from real people giving their genuine opinion on the car types.

During my literature review I observed that most researchers doing similar projects used twitter for their data collection but after manually searching multiple car models on twitter I decided that it was insuff because most of the posts were car advertisements posted by the car companies themselves and the data that was not advertisements was just generally of lower quality than reddit for sentiment analysis. Car advertisements are biased and would almost always have positive sentiment.

Other websites I considered using included the following

- boards.ie. quora and facebook, I decided not to use these because of the anti scraping software that their sites use made it difficult for scraping large amounts of data.
- Youtube - I considered using the text transcript of youtube car reviews for my sentiment analysis, but the videos were too long. Sentiment analysis models are more effective on shorter texts than longer texts
- Vwvortex.com - This is a one of the largest car forums in the world but it's search engine is not very good and the posts just didn't seem as relevant for sentiment analysis as reddit posts. A lot of the posts contained news articles related to certain car models

Ultimately I decided to use reddit because of their user friendly free API, relevant posts and anti advertisement/spam policies

The quality of the posts on reddit were far higher and they were more relevant.

Most of the posts were car reviews or people asking other users their opinions on cars, this was perfect for sentiment analysis.

The reddit dataset was collected using python and the reddit API

All unique models were collected from the donedeal dataset and put into a csv file

then a python script was used to iterate through the list

for each car model on the list the reddit API would search for the car model and extract the first reddit post and the top comment of the post.

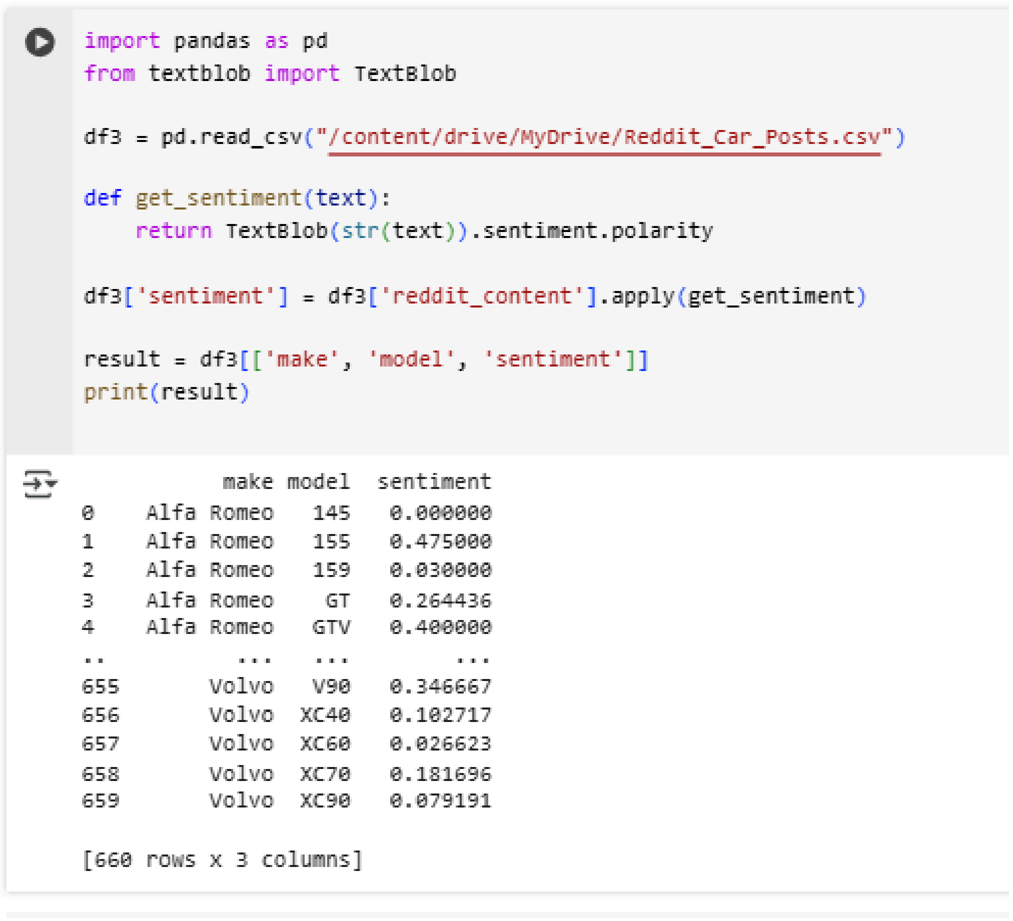
The reason the top comment was extracted along with each post because I noticed that many of the car related posts on reddit consisted of a person asking for opinions on the car model and the top comment would give their opinion. This was better for sentiment analysis than just using the post on their own

Sentiment Analysis

Textblob

Textblob is a lexicon based sentiment analysis tool

This was used to rate the sentiment of each post from 0 to 1, with zero being the worse and 1 being the best



```python
import pandas as pd
from textblob import TextBlob

df3 = pd.read_csv("/content/drive/MyDrive/Reddit_Car_Posts.csv")

def get_sentiment(text):
    return TextBlob(str(text)).sentiment.polarity

df3['sentiment'] = df3['reddit_content'].apply(get_sentiment)

result = df3[['make', 'model', 'sentiment']]
print(result)
```

```
           make model  sentiment
0    Alfa Romeo   145   0.000000
1    Alfa Romeo   155   0.475000
2    Alfa Romeo   159   0.030000
3    Alfa Romeo    GT   0.264436
4    Alfa Romeo   GTV   0.400000
..          ...   ...        ...
655       Volvo   V90   0.346667
656       Volvo  XC40   0.102717
657       Volvo  XC60   0.026623
658       Volvo  XC70   0.181696
659       Volvo  XC90   0.079191

[660 rows x 3 columns]
```

BERT Model using 1-2 scoring

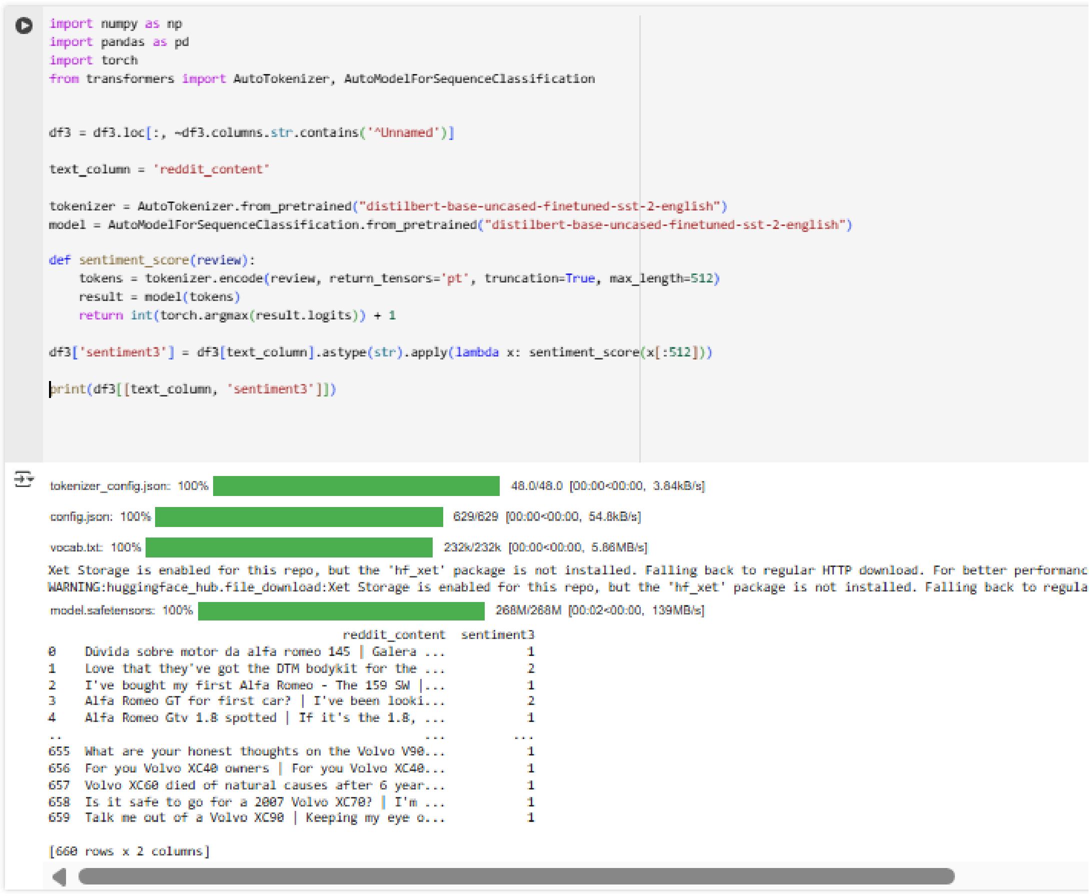
BERT models are more advanced forms of sentiment analysis tools than lexicon based tools such as text blob

The sentiment analysis model used here was “distilbert-base-uncased-finetuned-sst-2-english”

This is an english only sentiment analysis model that rates the sentiment of each post using a 1-2 scale

1 = Negative

2 = Positive



```python
import numpy as np
import pandas as pd
import torch
from transformers import AutoTokenizer, AutoModelForSequenceClassification


df3 = df3.loc[:, ~df3.columns.str.contains('^Unnamed')]

text_column = 'reddit_content'

tokenizer = AutoTokenizer.from_pretrained("distilbert-base-uncased-finetuned-sst-2-english")
model = AutoModelForSequenceClassification.from_pretrained("distilbert-base-uncased-finetuned-sst-2-english")

def sentiment_score(review):
    tokens = tokenizer.encode(review, return_tensors='pt', truncation=True, max_length=512)
    result = model(tokens)
    return int(torch.argmax(result.logits)) + 1

df3['sentiment3'] = df3[text_column].astype(str).apply(lambda x: sentiment_score(x[:512]))

print(df3[[text_column, 'sentiment3']])
```

```
tokenizer_config.json: 100%  48.0/48.0 [00:00<00:00, 3.84kB/s]
config.json: 100%  629/629 [00:00<00:00, 54.8kB/s]
vocab.txt: 100%  232k/232k [00:00<00:00, 5.86MB/s]
Xet Storage is enabled for this repo, but the 'hf_xet' package is not installed. Falling back to regular HTTP download. For better performanc
WARNING:huggingface_hub.file_download:Xet Storage is enabled for this repo, but the 'hf_xet' package is not installed. Falling back to regula
model.safetensors: 100%  268M/268M [00:02<00:00, 139MB/s]
                                         reddit_content  sentiment3
0    Dúvida sobre motor da alfa romeo 145 | Galera ...           1
1    Love that they've got the DTM bodykit for the ...           2
2    I've bought my first Alfa Romeo - The 159 SW |...           1
3    Alfa Romeo GT for first car? | I've been looki...           2
4    Alfa Romeo Gtv 1.8 spotted | If it's the 1.8, ...           1
..                                                 ...         ...
655  What are your honest thoughts on the Volvo V90...           1
656  For you Volvo XC40 owners | For you Volvo XC40...           1
657  Volvo XC60 died of natural causes after 6 year...           1
658  Is it safe to go for a 2007 Volvo XC70? | I'm ...           1
659  Talk me out of a Volvo XC90 | Keeping my eye o...           1

[660 rows x 2 columns]
```

Multilingual BERT Model using 1-5 scoring

This uses “nlptown/bert-base-multilingual-uncased-sentiment”

This is a multilingual sentiment analysis model that rates the sentiment on a scale from 1-5

with 1 being the most negative and 5 being the most positive

This method of sentiment analysis has it’s pros and cons

pros

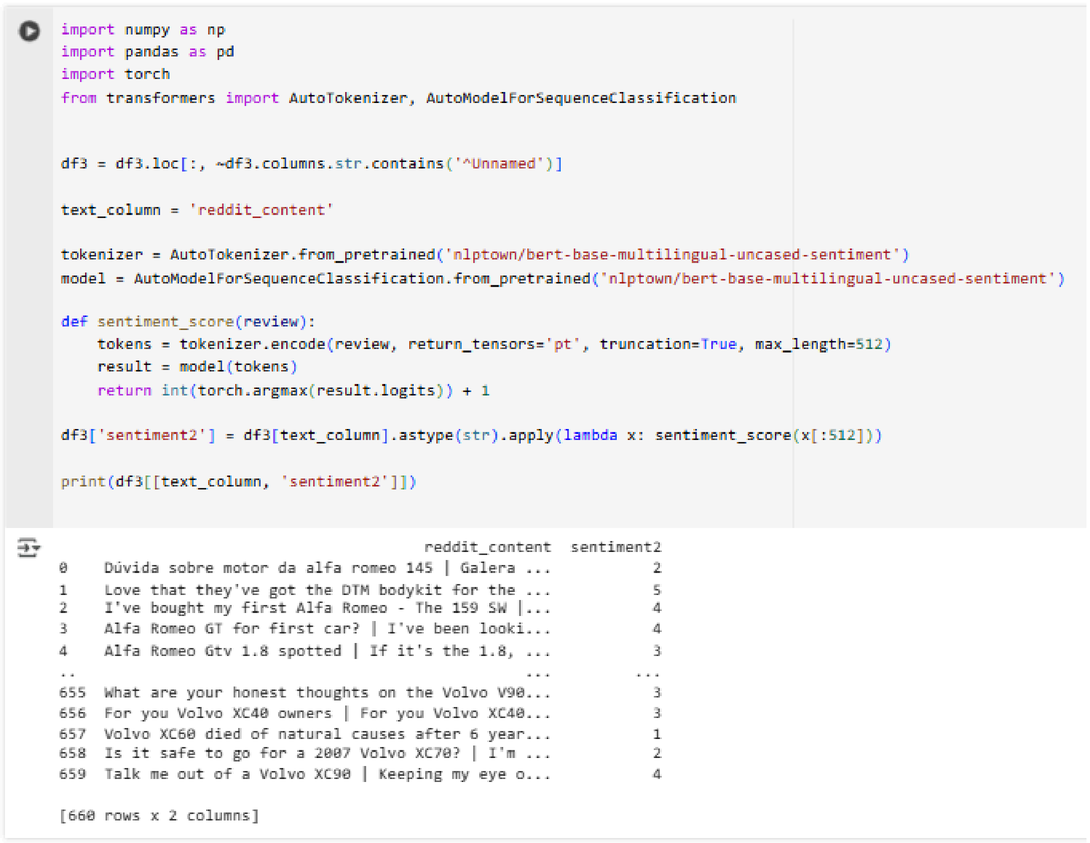
- It provides a more detailed and higher quality data, ie not only do you know if something is positive but you know just how positive

  it differentiates between cars that people have mildly positive/negative opinions on vs cars that people have extremely positive/negative opinions on.

- It is multilingual, many reddit posts are in different languages and this model is capable of rating the sentiment of these posts.

cons

- this has the potential to be a superior sentiment analysis model than just using the 1-2 scale but it also could be more influenced by irrelevant posts, ie if a text has 5 and it is irrelevant to the car model, it could give more misleading data than if a post is simply a 2



```python
import numpy as np
import pandas as pd
import torch
from transformers import AutoTokenizer, AutoModelForSequenceClassification


df3 = df3.loc[:, ~df3.columns.str.contains('^Unnamed')]

text_column = 'reddit_content'

tokenizer = AutoTokenizer.from_pretrained('nlptown/bert-base-multilingual-uncased-sentiment')
model = AutoModelForSequenceClassification.from_pretrained('nlptown/bert-base-multilingual-uncased-sentiment')

def sentiment_score(review):
    tokens = tokenizer.encode(review, return_tensors='pt', truncation=True, max_length=512)
    result = model(tokens)
    return int(torch.argmax(result.logits)) + 1

df3['sentiment2'] = df3[text_column].astype(str).apply(lambda x: sentiment_score(x[:512]))

print(df3[[text_column, 'sentiment2']])
```

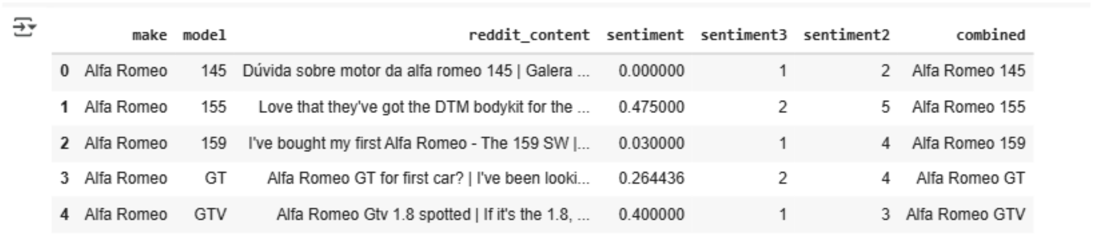
```
                                        reddit_content  sentiment2
0    Dúvida sobre motor da alfa romeo 145 | Galera ...           2
1    Love that they've got the DTM bodykit for the ...           5
2    I've bought my first Alfa Romeo - The 159 SW |...           4
3    Alfa Romeo GT for first car? | I've been looki...           4
4    Alfa Romeo Gtv 1.8 spotted | If it's the 1.8, ...           3
..                                                 ...         ...
655  What are your honest thoughts on the Volvo V90...           3
656  For you Volvo XC40 owners | For you Volvo XC40...           3
657  Volvo XC60 died of natural causes after 6 year...           1
658  Is it safe to go for a 2007 Volvo XC70? | I'm ...           2
659  Talk me out of a Volvo XC90 | Keeping my eye o...           4

[660 rows x 2 columns]
```

The output of each sentiment analysis tool were combined with the row of it's associated reddit post

Sentiment is the textblob sentiment analysis

sentiment 3 is the multilingual 1-5 BERT model

and sentiment 2 is the 1-2 BERT model

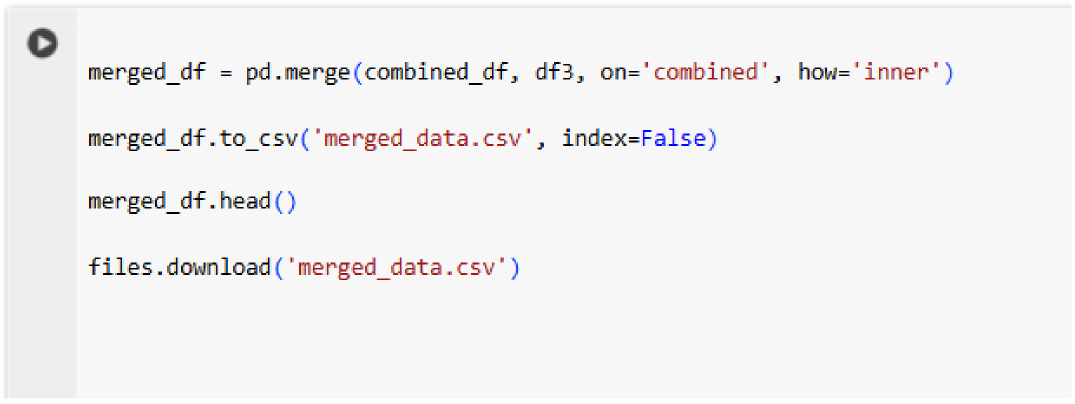


	make	model	reddit_content	sentiment	sentiment3	sentiment2	combined
0	Alfa Romeo	145	Dúvida sobre motor da alfa romeo 145 | Galera ...	0.000000	1	2	Alfa Romeo 145
1	Alfa Romeo	155	Love that they've got the DTM bodykit for the ...	0.475000	2	5	Alfa Romeo 155
2	Alfa Romeo	159	I've bought my first Alfa Romeo - The 159 SW |...	0.030000	1	4	Alfa Romeo 159
3	Alfa Romeo	GT	Alfa Romeo GT for first car? | I've been looki...	0.264436	2	4	Alfa Romeo GT
4	Alfa Romeo	GTV	Alfa Romeo Gtv 1.8 spotted | If it's the 1.8, ...	0.400000	1	3	Alfa Romeo GTV

Using python and pandas the reddit posts were combined with the donedeal dataset using an inner join where the car model of the reddit post dataset matched the car model in the donedeal dataset.

In the code below, merged_df is the donedeal dataset and df3 is the dataset that contains the reddit posts and the sentiment analysis scores of the three sentiment analysis models for each post.

The code below joins the df3 to the donedeal dataset where the car models both match.



```
merged_df = pd.merge(combined_df, df3, on='combined', how='inner')

merged_df.to_csv('merged_data.csv', index=False)

merged_df.head()

files.download('merged_data.csv')
```

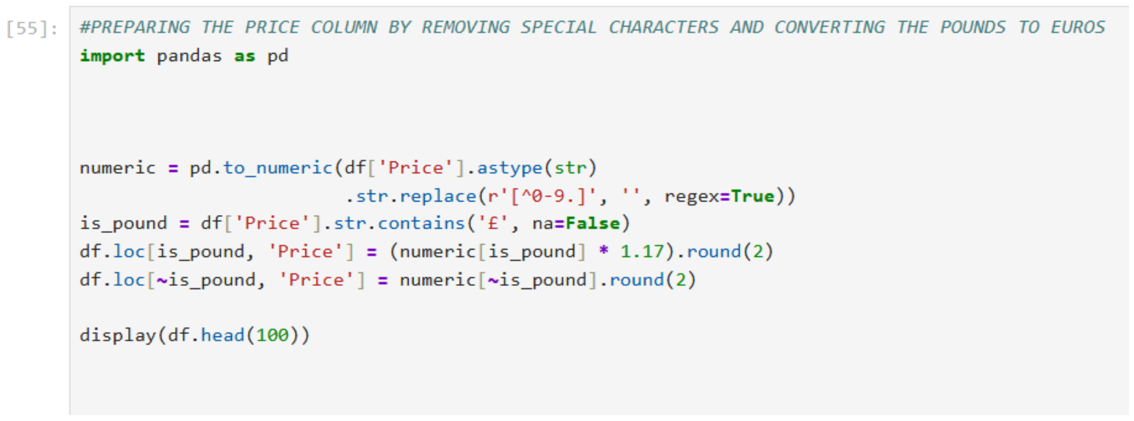
Data preparation

Data Cleaning

Data cleaning included the following steps

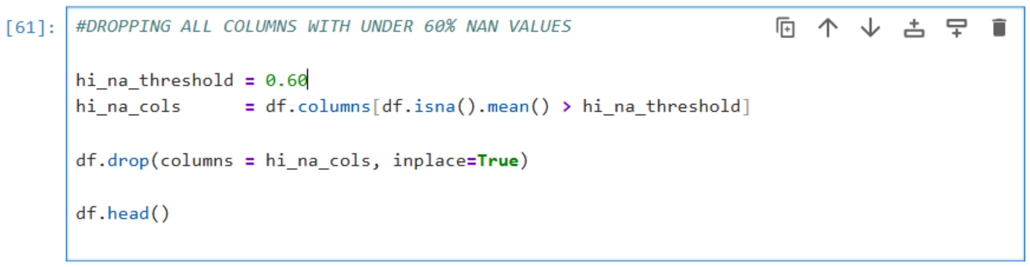
1. Cleaning the price column values - the price column had euro signs before every value, I used the following python code to remove everything from the code that does not contain numbers

Some of the price values were in pounds, so I used python to multiply the values with the pound sign in them by the exchange rate of 1.17 in order to convert them into euros



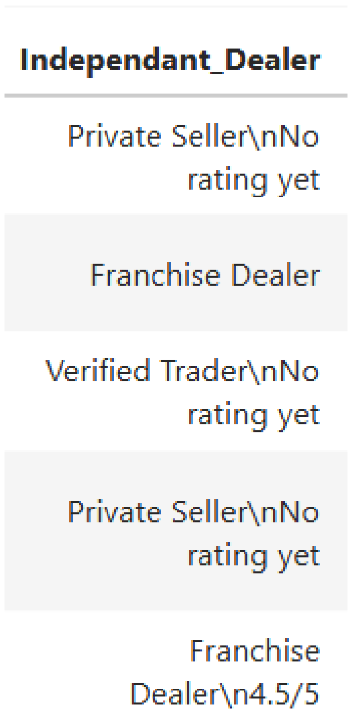
```
[55]: #PREPARING THE PRICE COLUMN BY REMOVING SPECIAL CHARACTERS AND CONVERTING THE POUNDS TO EUROS
import pandas as pd



numeric = pd.to_numeric(df['Price'].astype(str)
                        .str.replace(r'[^0-9.]', '', regex=True))
is_pound = df['Price'].str.contains('£', na=False)
df.loc[is_pound, 'Price'] = (numeric[is_pound] * 1.17).round(2)
df.loc[~is_pound, 'Price'] = numeric[~is_pound].round(2)

display(df.head(100))
```

2. Columns that had over 60% empty values were dropped, because columns with too much null values actually decrease the quality of machine learning models.

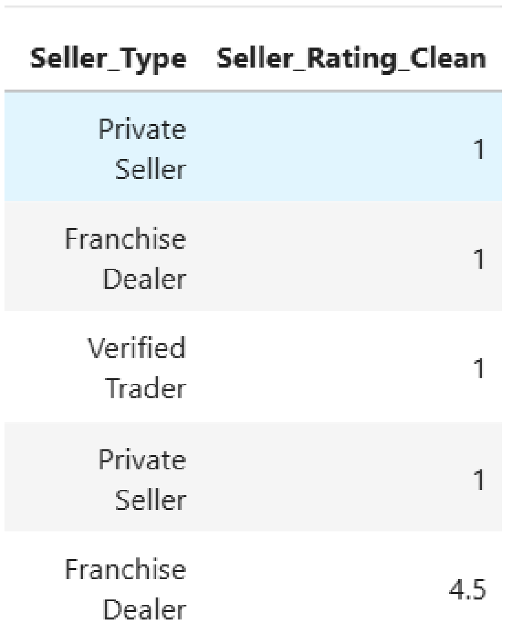


```
[61]: #DROPPING ALL COLUMNS WITH UNDER 60% NAN VALUES

hi_na_threshold = 0.60
hi_na_cols      = df.columns[df.isna().mean() > hi_na_threshold]

df.drop(columns = hi_na_cols, inplace=True)

df.head()
```

3. Some columns were scraped incorrectly by the web scraping software, and had to be separated using python such as the seller rating and seller type categories



Independant_Dealer
Private Seller\nNo rating yet
Franchise Dealer
Verified Trader\nNo rating yet
Private Seller\nNo rating yet
Franchise Dealer\n4.5/5



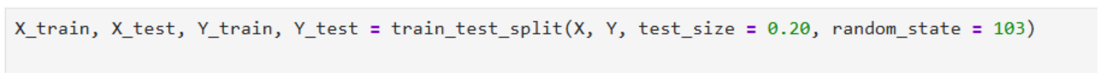
Seller_Type	Seller_Rating_Clean
Private Seller	1
Franchise Dealer	1
Verified Trader	1
Private Seller	1
Franchise Dealer	4.5

Splitting the Data

The data is split into a training set and a testing set

I used an 80/20 split meaning 80% of the dataset is used to train the models

and 20% of the data is used to test them



```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.20, random_state = 103)
```

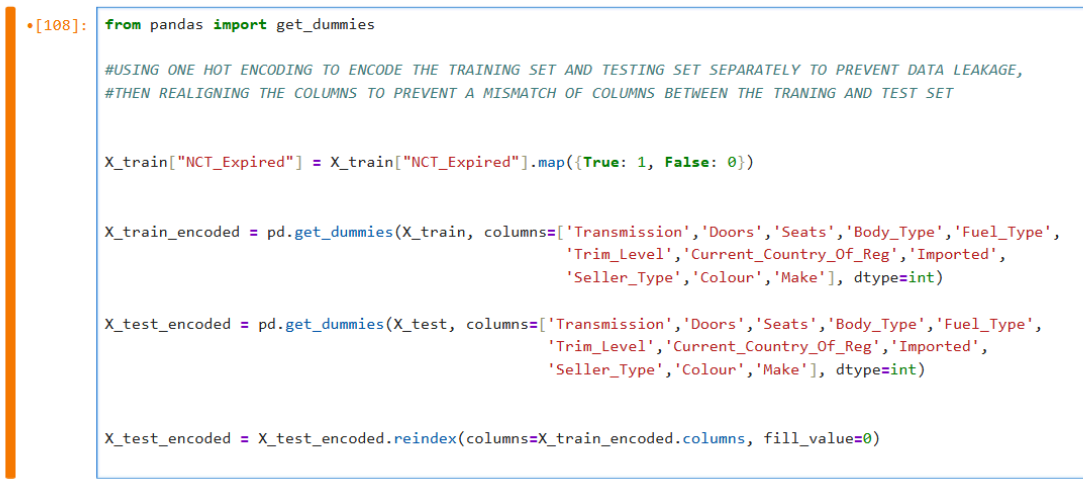
Encoding

Encoding is the process of transforming categorical or text data into numerical data

Machine learning models do not understand text they only understand numbers so all data must be converted into numbers. This was done in the following ways

The data was encoded using one hot encoding and target encoding

One hot encoding encodes the data by converting every unique value in a column into it's own column then adding the number 1 in every row where that value is present and a zero in every row where it is not present



```
•[108]: from pandas import get_dummies

        #USING ONE HOT ENCODING TO ENCODE THE TRAINING SET AND TESTING SET SEPARATELY TO PREVENT DATA LEAKAGE,
        #THEN REALIGNING THE COLUMNS TO PREVENT A MISMATCH OF COLUMNS BETWEEN THE TRANING AND TEST SET


        X_train["NCT_Expired"] = X_train["NCT_Expired"].map({True: 1, False: 0})


        X_train_encoded = pd.get_dummies(X_train, columns=['Transmission','Doors','Seats','Body_Type','Fuel_Type',
                                                           'Trim_Level','Current_Country_Of_Reg','Imported',
                                                           'Seller_Type','Colour','Make'], dtype=int)

        X_test_encoded = pd.get_dummies(X_test, columns=['Transmission','Doors','Seats','Body_Type','Fuel_Type',
                                                         'Trim_Level','Current_Country_Of_Reg','Imported',
                                                         'Seller_Type','Colour','Make'], dtype=int)


        X_test_encoded = X_test_encoded.reindex(columns=X_train_encoded.columns, fill_value=0)
```

The training set and testing set are separated before encoding to prevent data leakage

Re-indexing

After one hot encoding is done the reindex function is used to ensure that the training set and the testing set have the same columns

if there is more columns in the training set than the testing set this will cause problems for the machine learning models

The new columns are given a default value of zero

Target encoding

Target encoding is another encoding method that converts categorical values into numerical

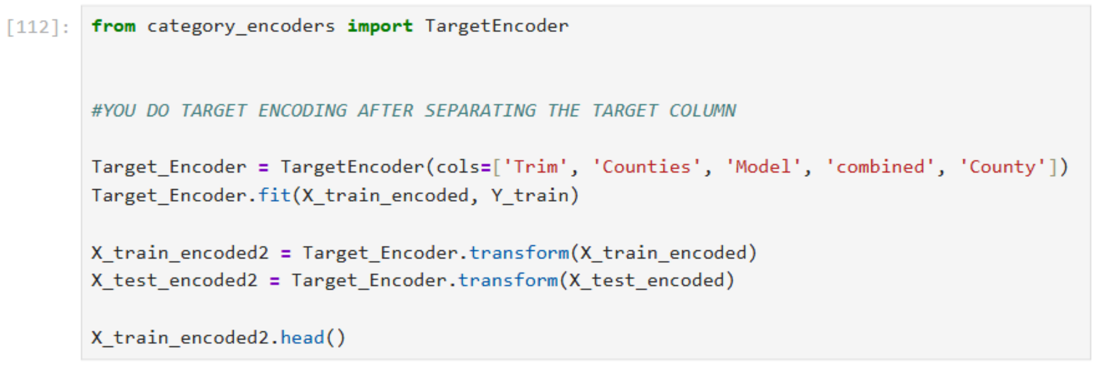
The numbers given to each value is dependent on it's relation to the target variable

For each unique value, the target encoding function adds up every price with that unique value and finds the average price of rows with that value and assigns the average number to it.

The main benefit of target encoding is that it doesn’t introduce new columns to the dataset, this helps keep the dimensionality low.

I used target encoding on categorical variables that had a very high number of categories.

The reason I did this is that there are some features with over 100 unique variables and if I used one hot encoding on these features it would lead to far too much dimensionality in the data which leads to subpar machine learning model performance



```
[112]: from category_encoders import TargetEncoder


       #YOU DO TARGET ENCODING AFTER SEPARATING THE TARGET COLUMN

       Target_Encoder = TargetEncoder(cols=['Trim', 'Counties', 'Model', 'combined', 'County'])
       Target_Encoder.fit(X_train_encoded, Y_train)

       X_train_encoded2 = Target_Encoder.transform(X_train_encoded)
       X_test_encoded2 = Target_Encoder.transform(X_test_encoded)

       X_train_encoded2.head()
```

Standardization

Standardization transforms each feature so that it has a mean of zero and a standard deviation of 1

This makes it easier for the machine learning models to form predictions

because all of the features will be on a similar scale and the model will not be unevenly effected by features with large numbers.

some features have large numbers in the thousands and some are less than 10 standardization reduces every feature so it is close to 1 so each feature has similar influence.

Scikit learn Standard scalar function was used to standardize the data

first the scalar is fit to the training data using scalar.fit

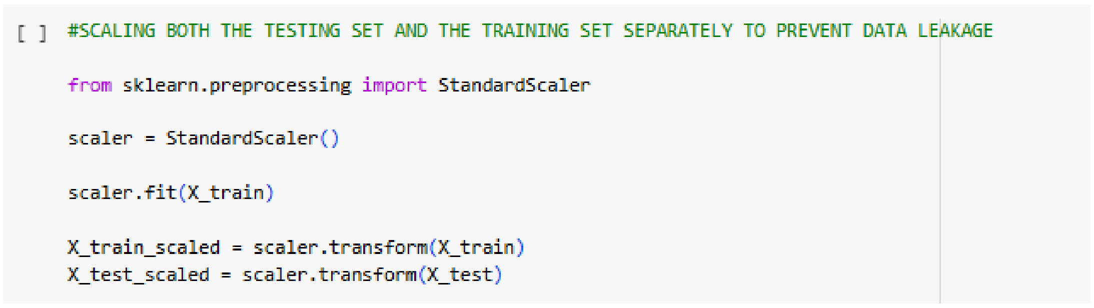
This computes the changes that need to be made to each feature to standardize them (reduce them to a mean of 0 and standard deviation of 1)

then it is transformed using scalar.transform

this applies the changes to the actual data

The training set and testing set were scaled separately in order to prevent data leakage

if the data was split before standardization the training data would be influenced by the testing data which would lead to data leakage



```
[ ]  #SCALING BOTH THE TESTING SET AND THE TRAINING SET SEPARATELY TO PREVENT DATA LEAKAGE

     from sklearn.preprocessing import StandardScaler

     scaler = StandardScaler()

     scaler.fit(X_train)

     X_train_scaled = scaler.transform(X_train)
     X_test_scaled = scaler.transform(X_test)
```

Machine Learning Model Creation

Random Forest Regressor

“Random forests are a combination of tree predictors such that each tree depends on the values of a

random vector sampled independently and with the same distribution for all trees in the forest" (BREIMAN 2001)

A random forest is a combination of decision trees that creates a superior prediction to using one prediction tree alone.

For my random forest regressor I used sklearns Randomforesttregressor function

I tested a variety of different hyperparameters and found that the optimal hyperparameters were 800 n estimators, a max depth of "None", cv of 8 and max features of square root

- **N estimators** - this hyperparameter allows you to choose the number of decision trees used in the random forest regressor, 800 decision trees are used here
- Max depth - max depth is the number of splits for each node in the decision tree, here the max depth is none, this means there is no specified limit for the decision trees. they simply stop splitting when it can no longer split the data i.e when the nodes are pure
- cv - This is the cross validation hyperparameter, this controls the number of different subsets of the data that is used for cross validation.

  In this case the data is split into 8 different subsets for cross validation

  Cross validation is a process where the data is split into different subsets

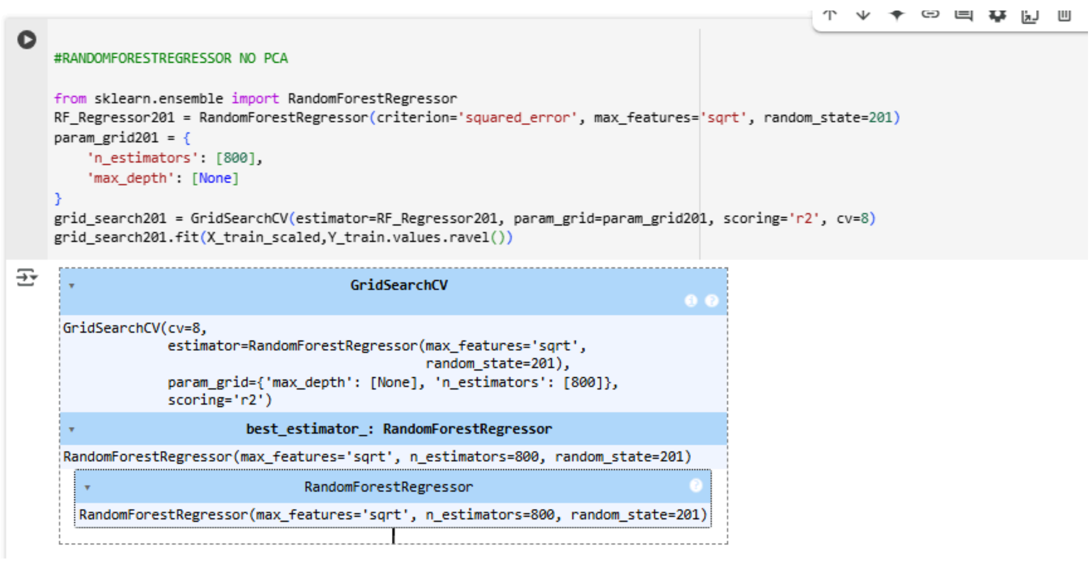
  each subset acts as the testing set and the other subsets act as the training sets.

  This is done for each subset until every subset of the data has been used as the testing set.

  This creates a robust model that is far very familiarized with the data and the relationships between the features.
- Max features - This refers to the number of features that are considered by the model when each node is split

  max features sqrt means that the number of features considered for each split is the square root of the number of features in the dataset,

  There are 152 columns in this dataset so that would mean 12 features are considered for each split



```
#RANDOMFORESTREGRESSOR NO PCA

from sklearn.ensemble import RandomForestRegressor
RF_Regressor201 = RandomForestRegressor(criterion='squared_error', max_features='sqrt', random_state=201)
param_grid201 = {
    'n_estimators': [800],
    'max_depth': [None]
}
grid_search201 = GridSearchCV(estimator=RF_Regressor201, param_grid=param_grid201, scoring='r2', cv=8)
grid_search201.fit(X_train_scaled,Y_train.values.ravel())
```

```
GridSearchCV

GridSearchCV(cv=8,
             estimator=RandomForestRegressor(max_features='sqrt',
                                             random_state=201),
             param_grid={'max_depth': [None], 'n_estimators': [800]},
             scoring='r2')

best_estimator_: RandomForestRegressor
RandomForestRegressor(max_features='sqrt', n_estimators=800, random_state=201)

RandomForestRegressor
RandomForestRegressor(max_features='sqrt', n_estimators=800, random_state=201)
```

Support Vector Regressor

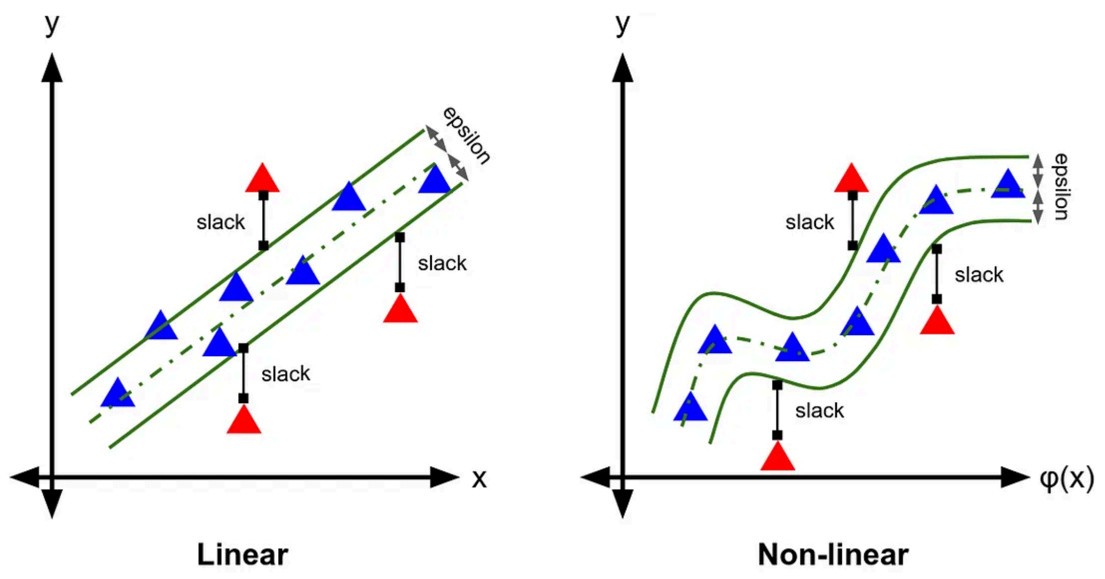
“As in classification, support vector regression (SVR) is characterized by the use of kernels, sparse solution, and VC control of the margin and the number of support vectors. Although less popular than SVM, SVR has been proven to be an effective tool in real-value function estimation” (Awad and Khanna, 2015).

Support vector regression is a machine learning method that predicts numerical data by creating a line or curve that aims to go through the most data points possible

this line is called a hyperplane, the surrounding data points are called support vectors

The line cannot perfectly go through all of the data points because they are too spread out

instead, it attempts to capture the maximum number of data points in an area around the hyperplane known as the epsilon tube.

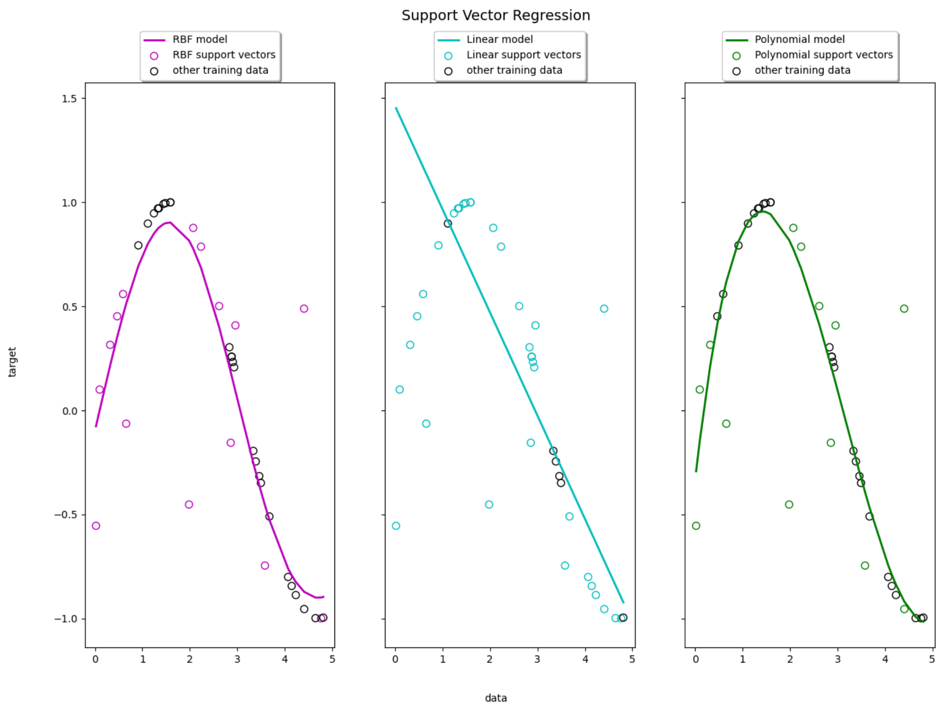


The optimal hyperparameters for this car price prediction model is as following

Kernel - RBf - The kernel refers to the function used to increase the dimensionality of the hyperplane, support vectors are very effective because they work in different dimensions. of data

if there is no good line that goes through data points effectively,changing the dimension often solves this problem. The Kernel trick is the process of increasing the number of dimensions until you can find a hyperplane that effectively captures the pattern of the data.

I used the RBF kernel



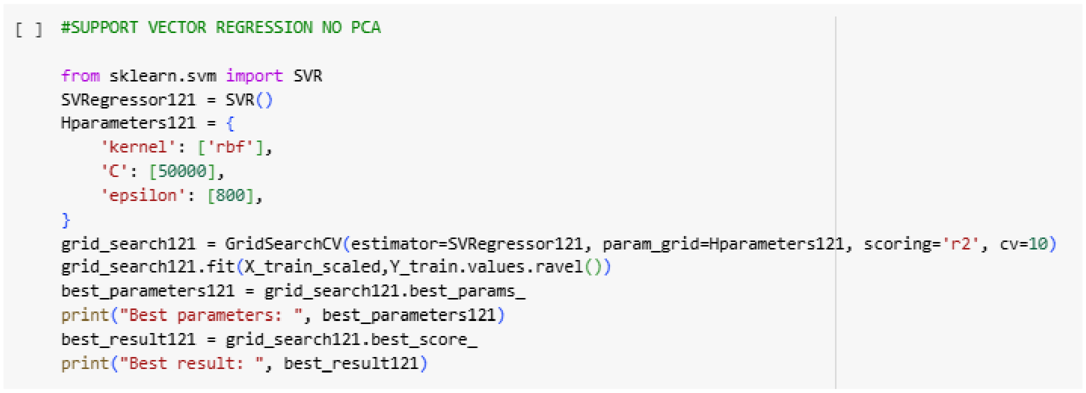
(scikit-learn.org 2025)

C- 50,000 - This hyperparameter controls how strong the penalty is for data points that are outside the epsilon tube. I used a C of 50,000 which is a strong penalty

Epsilon -This hyperparameter specifies the size of the epsilon tube ie how far data points have to be from the hyperplane before they are penalized.(**Scikit-learn**, 2024.) I used an epsilon of 800

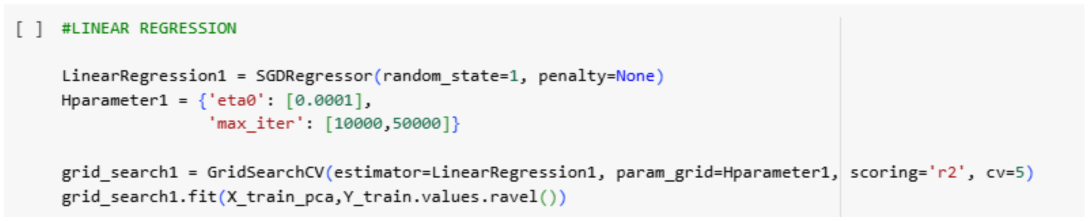
“Epsilon in the epsilon-SVR model. It specifies the epsilon-tube within which no penalty is associated in the training loss function with points predicted within a distance epsilon from the actual value. Must be non-negative.”

Cv = 10 - Cross validation of 10



```
[ ] #SUPPORT VECTOR REGRESSION NO PCA

    from sklearn.svm import SVR
    SVRegressor121 = SVR()
    Hparameters121 = {
        'kernel': ['rbf'],
        'C': [50000],
        'epsilon': [800],
    }
    grid_search121 = GridSearchCV(estimator=SVRegressor121, param_grid=Hparameters121, scoring='r2', cv=10)
    grid_search121.fit(X_train_scaled,Y_train.values.ravel())
    best_parameters121 = grid_search121.best_params_
    print("Best parameters: ", best_parameters121)
    best_result121 = grid_search121.best_score_
    print("Best result: ", best_result121)
```

Linear Regression



```
[ ] #LINEAR REGRESSION

    LinearRegression1 = SGDRegressor(random_state=1, penalty=None)
    Hparameter1 = {'eta0': [0.0001],
                   'max_iter': [10000,50000]}

    grid_search1 = GridSearchCV(estimator=LinearRegression1, param_grid=Hparameter1, scoring='r2', cv=5)
    grid_search1.fit(X_train_pca,Y_train.values.ravel())
```

"LinearRegression fits a linear model with coefficients w = (w1, ..., wp) to minimize the residual sum of squares between the observed targets in the dataset, and the targets predicted by the linear approximation." (https://scikit-learn.org/ 2025)

Linear regression is a machine learning function that aims to create a line that goes through the middle of the maximum number of data points in order to capture the pattern of the data

The hyper parameters used here include the following

- eta0 - This is the learning rate of the regressor

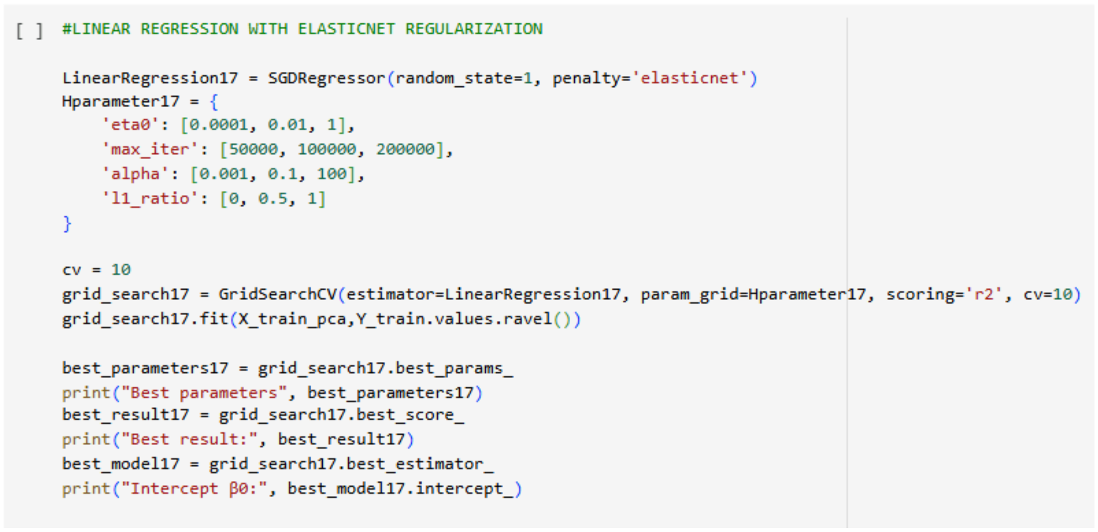
"The initial learning rate for the 'constant', 'invscaling' or 'adaptive' schedules. The default value is 0.01." (scikit-learn 2025)

I used a eta0 of 0.0001 which is far lower than normal

- Max_iter - This is the maximum iterations hyperparameter, it allows you to specify the number of iterations the machine learning model does.

  This refers to the number of times the machine learning model passes over the training data and makes and adjustment

L inear Regression with Elastic Net Regression



```
[ ] #LINEAR REGRESSION WITH ELASTICNET REGULARIZATION

    LinearRegression17 = SGDRegressor(random_state=1, penalty='elasticnet')
    Hparameter17 = {
        'eta0': [0.0001, 0.01, 1],
        'max_iter': [50000, 100000, 200000],
        'alpha': [0.001, 0.1, 100],
        'l1_ratio': [0, 0.5, 1]
    }

    cv = 10
    grid_search17 = GridSearchCV(estimator=LinearRegression17, param_grid=Hparameter17, scoring='r2', cv=10)
    grid_search17.fit(X_train_pca,Y_train.values.ravel())

    best_parameters17 = grid_search17.best_params_
    print("Best parameters", best_parameters17)
    best_result17 = grid_search17.best_score_
    print("Best result:", best_result17)
    best_model17 = grid_search17.best_estimator_
    print("Intercept β0:", best_model17.intercept_)
```

Elastic net regression uses a combination of the ridge and lasso regression methods. These regularization methods aim to reduce the influence of features that do not improve the predictions of the model. This reduces noise and allow the machine learning model to use more resources focusing on the features that are most effective for creating accurate predictions

This is done by adding penalties to the features to reduce their influence

Lasso regression

Lasso regression lowers adds a penalty to the features and decreases some to zero

Ridge regression

Ridge regression is similar to lasso regression, features have their coefficients reduced but never have their coefficients decreased to zero

alpha - this hyperparameter allows you to control how much penalty is applied to the coefficients that are irrelevant. I used an alpha of 0.1

l1 ratio - This hyperparameter allows you to control the ratio of ridge and lasso regression an l1 ratio of 0 would be full ridge regression and an l1 ratio of 1 would be full lasso regression.

I used an l1 ratio of 0.5 which is means 50% ridge regression was used and 50% lasso regression was used.

Results

R squared

The main evaluation metric used for this prediction model is r squared

R squared is a measure of how accurate predictions are in regression models

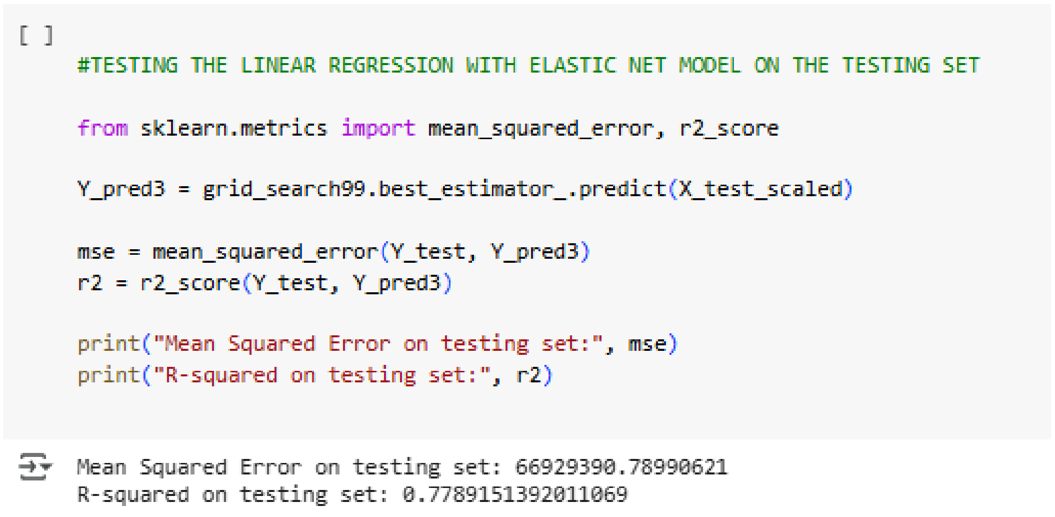
The best possible score is 1,

A score of zero isn’t necessarily the worst possible prediction it just means the model simply predicts the expected value of y

R squared measures how much of the variance is explained by the machine learning model, the higher the r squared, the better the predictions

For this project I used Scikit learns r2_score function to calculate the r squared

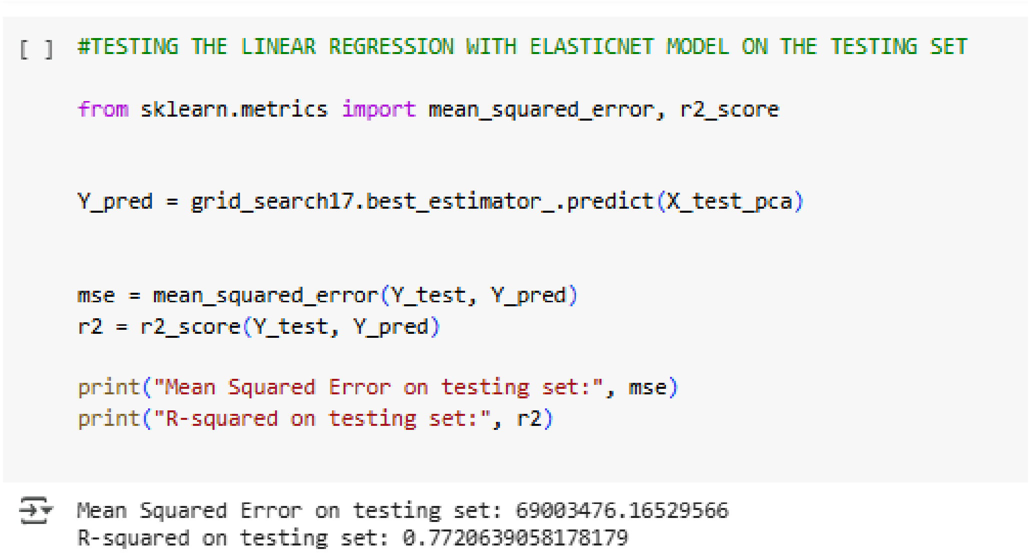
Linear Regressor scored 0.762 on the training set and 0.778 on the testing set



```
[ ]
    #TESTING THE LINEAR REGRESSION WITH ELASTIC NET MODEL ON THE TESTING SET

    from sklearn.metrics import mean_squared_error, r2_score

    Y_pred3 = grid_search99.best_estimator_.predict(X_test_scaled)

    mse = mean_squared_error(Y_test, Y_pred3)
    r2 = r2_score(Y_test, Y_pred3)

    print("Mean Squared Error on testing set:", mse)
    print("R-squared on testing set:", r2)
```

```
Mean Squared Error on testing set: 66929390.78990621
R-squared on testing set: 0.7789151392011069
```

Linear Regressor with elastic net regression scored 0.766 on the training set and 0.772 on the testing set

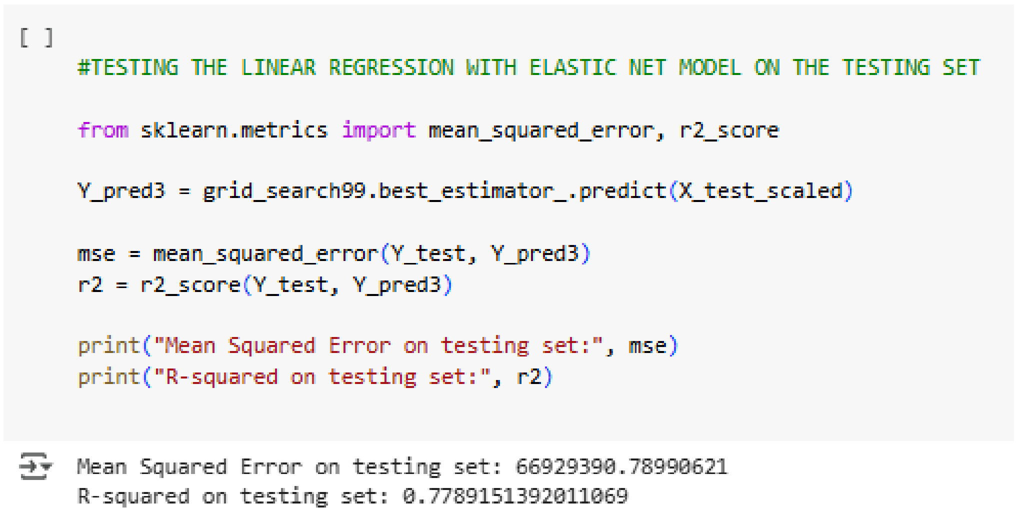


```
[ ] #TESTING THE LINEAR REGRESSION WITH ELASTICNET MODEL ON THE TESTING SET

    from sklearn.metrics import mean_squared_error, r2_score


    Y_pred = grid_search17.best_estimator_.predict(X_test_pca)


    mse = mean_squared_error(Y_test, Y_pred)
    r2 = r2_score(Y_test, Y_pred)

    print("Mean Squared Error on testing set:", mse)
    print("R-squared on testing set:", r2)
```

```
Mean Squared Error on testing set: 69003476.16529566
R-squared on testing set: 0.7720639058178179
```

Linear Regressor

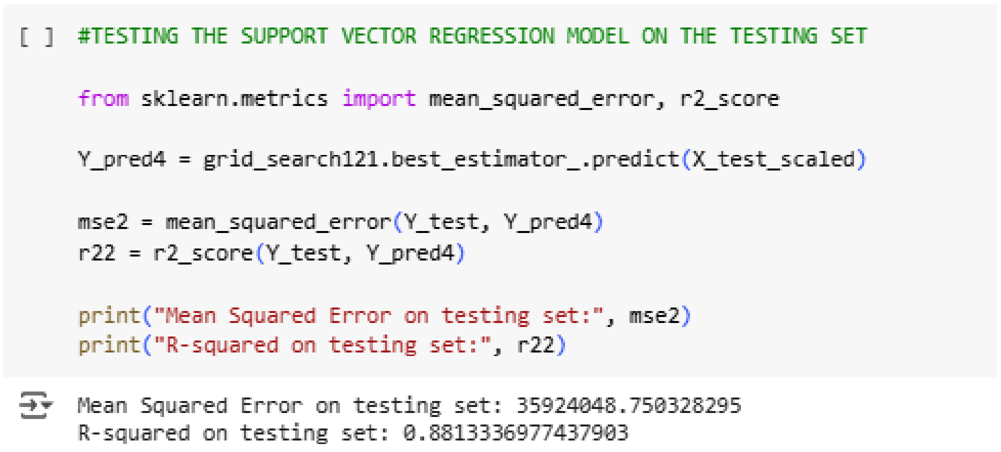
The linear regressor scored 0.762 on the training set and 0.778 on the testing set



```
[ ]
    #TESTING THE LINEAR REGRESSION WITH ELASTIC NET MODEL ON THE TESTING SET

    from sklearn.metrics import mean_squared_error, r2_score

    Y_pred3 = grid_search99.best_estimator_.predict(X_test_scaled)

    mse = mean_squared_error(Y_test, Y_pred3)
    r2 = r2_score(Y_test, Y_pred3)

    print("Mean Squared Error on testing set:", mse)
    print("R-squared on testing set:", r2)
```

```
Mean Squared Error on testing set: 66929390.78990621
R-squared on testing set: 0.7789151392011069
```

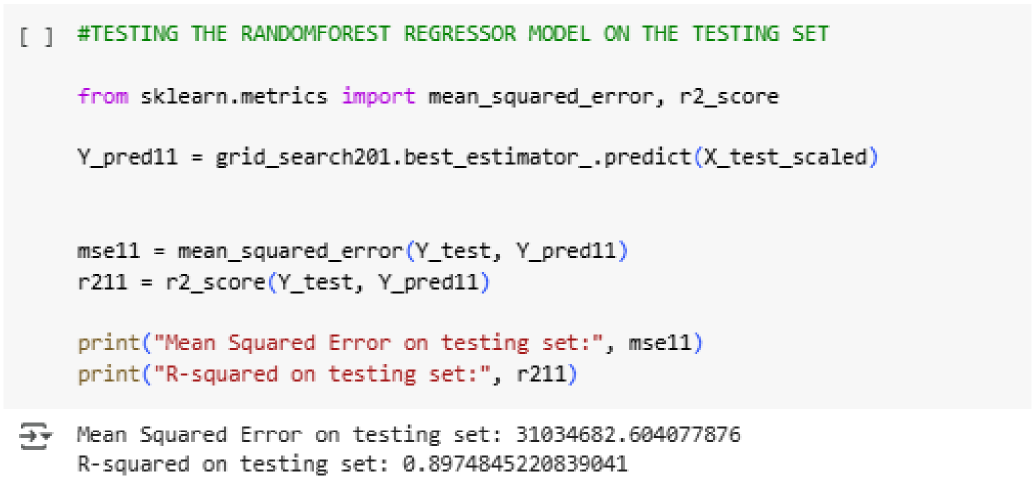
The Support Vector Regressor scored 0.866 on the training set and 0.881 on the testing set



```
[ ] #TESTING THE SUPPORT VECTOR REGRESSION MODEL ON THE TESTING SET

    from sklearn.metrics import mean_squared_error, r2_score

    Y_pred4 = grid_search121.best_estimator_.predict(X_test_scaled)

    mse2 = mean_squared_error(Y_test, Y_pred4)
    r22 = r2_score(Y_test, Y_pred4)

    print("Mean Squared Error on testing set:", mse2)
    print("R-squared on testing set:", r22)
```

```
Mean Squared Error on testing set: 35924048.750328295
R-squared on testing set: 0.8813336977437903
```

The random forest regressor scored 0.896 on the training set and 0.897 on the testing set



```
[ ] #TESTING THE RANDOMFOREST REGRESSOR MODEL ON THE TESTING SET

    from sklearn.metrics import mean_squared_error, r2_score

    Y_pred11 = grid_search201.best_estimator_.predict(X_test_scaled)


    mse11 = mean_squared_error(Y_test, Y_pred11)
    r211 = r2_score(Y_test, Y_pred11)

    print("Mean Squared Error on testing set:", mse11)
    print("R-squared on testing set:", r211)
```

```
Mean Squared Error on testing set: 31034682.604077876
R-squared on testing set: 0.8974845220839041
```

All of the models performed slightly better on the testing set than the training set, this means there is no overfitting and maybe slight underfitting taking place.

The best machine learning model for predicting car prices was the random forest regressor with an r squared value of 0.894

Random forest was likely the best machine learning model because of the complex non linear relationships of the features in the data. The fact that the linear regressors scored much lower supports this theory because linear regressors are less effective at handling non linear relationships between features.

Another reason random forest was so effective was likely because random forest is great at handling high dimensional data

The dataset had over 150 features after encoding which is very high.

In conclusion an r squared of 0.89 is high and shows that used car prices can be accurately predicted using machine learning

The sentiment analysis feature added a slight increase in the models ability to predict prices but it was relatively insignificant compared to other features

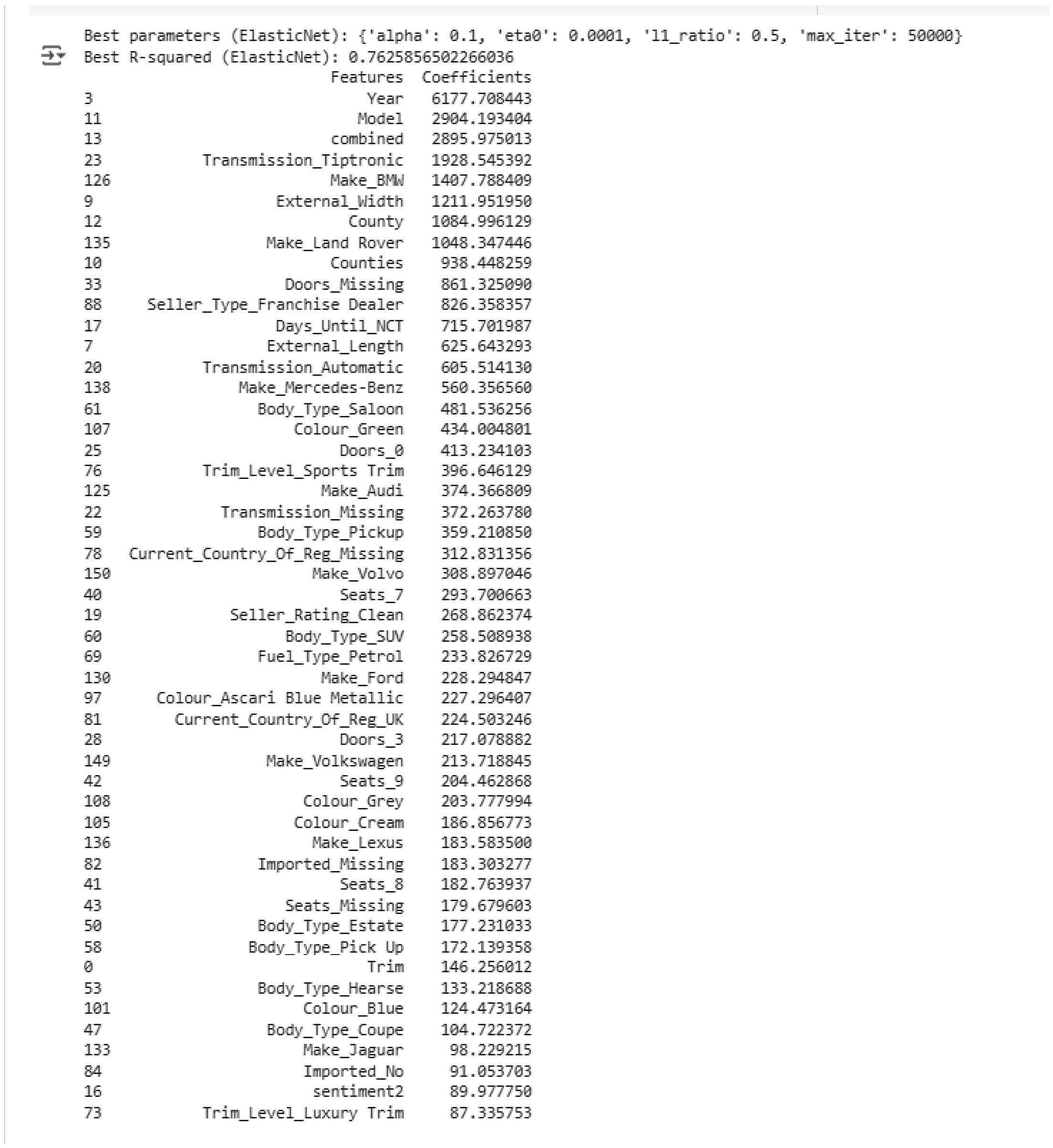
Below is a list of the top 50 features that were most effective for predicting the car prices.

Sentiment 2 is the 49th most important features out of over 150 features

Sentiment 2 was the multilingual BERT sentiment analyzer that rated sentiment on a scale from 1-5

it performed far better than the other two sentiment analysis techniques

this suggests that using a BERT model is likely better than using textblob and rating sentiment from 1-5 is likely better than simply rating them from



```
Best parameters (ElasticNet): {'alpha': 0.1, 'eta0': 0.0001, 'l1_ratio': 0.5, 'max_iter': 50000}
Best R-squared (ElasticNet): 0.7625856502266036
                          Features  Coefficients
3                             Year   6177.708443
11                           Model   2904.193404
13                        combined   2895.975013
23          Transmission_Tiptronic   1928.545392
126                       Make_BMW   1407.788409
9                   External_Width   1211.951950
12                          County   1084.996129
135                Make_Land Rover   1048.347446
10                        Counties    938.448259
33                   Doors_Missing    861.325090
88    Seller_Type_Franchise Dealer    826.358357
17                  Days_Until_NCT    715.701987
7                  External_Length    625.643293
20          Transmission_Automatic    605.514130
138             Make_Mercedes-Benz    560.356560
61                Body_Type_Saloon    481.536256
107                   Colour_Green    434.004801
25                         Doors_0    413.234103
76          Trim_Level_Sports Trim    396.646129
125                      Make_Audi    374.366809
22            Transmission_Missing    372.263780
59                Body_Type_Pickup    359.210850
78   Current_Country_Of_Reg_Missing    312.831356
150                     Make_Volvo    308.897046
40                         Seats_7    293.700663
19             Seller_Rating_Clean    268.862374
60                   Body_Type_SUV    258.508938
69                Fuel_Type_Petrol    233.826729
130                      Make_Ford    228.294847
97      Colour_Ascari Blue Metallic    227.296407
81        Current_Country_Of_Reg_UK    224.503246
28                         Doors_3    217.078882
149                Make_Volkswagen    213.718845
42                         Seats_9    204.462868
108                    Colour_Grey    203.777994
105                   Colour_Cream    186.856773
136                     Make_Lexus    183.583500
82                Imported_Missing    183.303277
41                         Seats_8    182.763937
43                   Seats_Missing    179.679603
50                Body_Type_Estate    177.231033
58               Body_Type_Pick Up    172.139358
0                             Trim    146.256012
53                Body_Type_Hearse    133.218688
101                    Colour_Blue    124.473164
47                 Body_Type_Coupe    104.722372
133                    Make_Jaguar     98.229215
84                     Imported_No     91.053703
16                      sentiment2     89.977750
73          Trim_Level_Luxury Trim     87.335753
```

BIBLIOGRAPHY

scikit-learn developers, 2025.
sklearn.linear_model.ElasticNet. [online] scikit-learn. Available at: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.ElasticNet.html [Accessed 11 May 2025].

Scikit-learn. (n.d.). *1.4.2. r2_score*. [online] Available at: https://scikit-learn.org/0.17/modules/generated/sklearn.metrics.r2_score.html [Accessed 11 May 2025].
scikit-learn.com (2025)
sklearn.linear_model.SGDRegressor — scikit-learn 1.6.1 documentation. Available at: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.SGDRegressor.html (Accessed: 11 May 2025).

https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.SGDRegressor.html
Scikit-learn developers, no date.
sklearn.linear_model.LinearRegression. [online] Scikit-learn. Available at: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html [Accessed 10 May 2025].

Scikit-learn developers, no date.
SVM regression. [online] Scikit-learn. Available at: https://scikit-learn.org/stable/auto_examples/svm/plot_svm_regression.html [Accessed 10 May 2025].

Scikit-learn, 2024. *sklearn.svm.SVR — scikit-learn 1.4.2 documentation*. [online] Available at: https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVR.html [Accessed 10 May 2025].

Awad, M. and Khanna, R., 2015. *Support Vector Regression*. In: *Efficient Learning Machines: Theories, Concepts, and Applications for Engineers and System Designers*. Apress. Available at: https://link.springer.com/chapter/10.1007/978-1-4302-5990-9_4 [Accessed 10 May 2025].

Beny Maulana Achsan (10/12/2019). *Support Vector Machine Regression*. [online] Medium. Available at: https://medium.com/it-paragon/support-vector-machine-regression-cf65348b6345 [Accessed 10 May 2025].

Scikit-learn developers, 2024.
sklearn.ensemble.RandomForestRegressor. [online] Scikit-learn. Available at:
https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html
[Accessed 10 May 2025].

nicknochnack (2021)
Sentiment.ipynb. GitHub. Available at:
https://github.com/nicknochnack/BERTSentiment/blob/main/Sentiment.ipynb
(Accessed: 11 May 2025).

> NeuralNine (2024) Simple Sentiment Text Analysis in Python. YouTube video, 10 March. Available at: https://www.youtube.com/watch?v=tXuvh5_Xyrw (Accessed: 11 May 2025).

> Renotte, N. (2023) Sentiment Analysis with BERT Neural Network and Python. YouTube video, 10 March. Available at: https://www.youtube.com/watch?v=szczpgOEdXs (Accessed: 11 May 2025).YouTube

Mulla, R. (2023)

Python Sentiment Analysis Project with NLTK and Huggingface

. YouTube video, 10 March. Available at:

https://www.youtube.com/watch?v=QpzMWQvxXWk

(Accessed: 11 May 2025).

Robikscube (2022)
Sentiment Analysis Python 🤗 [YouTube Tutorial]. Kaggle. Available at: https://www.kaggle.com/code/robikscube/sentiment-analysis-python-youtube-tutorial (Accessed: 11 May 2025).

Parsehub (2025) *Free web scraping - the most powerful web scraper*, *ParseHub*. Available at: https://www.parsehub.com/ (Accessed: 10 April 2025).

Moaiad Ahmad Khder, "Web Scraping or Web Crawling: State of Art, Techniques, Approaches and Application". Int. J. Advance Soft Compu. Appl, 2021, 13(3).

Source: Choudhary, A., & Arora, S. (2021). Web Scraping Techniques and Applications: A Review. International Journal of Intelligent Systems and Applications in Engineering, 9(1), 88–98.

Breiman, L., 2001. Random forests. *Machine Learning*, 45(1), pp.5–32. Available at: https://link.springer.com/article/10.1023/A:1010933404324 [Accessed 10 May 2025].

Shaprapawad, S., Borugadda, P. and Koshika, N., 2023. Car Price Prediction: An Application of Machine Learning.
Proceedings of the International Conference on
Inventive Computation Technologies (ICICT 2023)
. Available at:
https://www.ieee.org (Accessed 23 June 2024)
4o

Kavitha, V., Sai, P. D., Revathi, S., & Reddy, P. V. K. (2019). Car Buying Decision using Machine Learning Algorithms.
International Journal of Research in
Advanced Science and Engineering Technology, 6 6, 18041809

Kalpana, G., Durga, A.K., Reddy, T.A. and Karuna, G., 2022. Predicting the Price of

Pre-Owned Cars Using Machine Learning and Data Science.
International Journal
for Research in Applied Science & Engineering Technology (IJRASET)
, 10 (VII),
pp.1468-1476. Available at:

https://doi.org/10.22214/ijraset.2022.45469
(23 June 2024)(Accessed)

Sharma, K.M.L., Kumar, P., Budur, M., Reddy, S.P. and Narayan, M., 2023. Anticipation of Car Price Using Machine Learning Approach. International Journal for Research in Applied Science & Engineering Technology (IJRASET), 11(V), pp.3585-3589. Available at:

https://doi.org/10.22214/ijraset.2023.51817 (Accessed
23 June 2024).

Shaprapawad, S., Borugadda, P. and Koshika, N., 2023. Car Price Prediction: An Application of Machine Learning. Proceedings of the International Conference on Inventive Computation Technologies (ICICT 2023) Available at:

https://www.ieee.org (Accessed 23 June 2024).

Putra, P.H., Azanuddin, B., Purba, B. and Dalimunthe, Y.A., 2023. Random forest and decision tree algorithms for car price prediction.
Jurnal Matematika Dan Ilmu Pengetahuan Alam LLDikti Wilayah 1 (JUMPA), 3 (2, pp.81-89. Available at:

https://lldikti1.ristekdikti.go.id/jurnal/index.php/jumpa
(Accessed 23 June 2024).

Varshitha, J., Jahnavi, K. and Lakshmi, C., 2022. Prediction Of Used Car Prices Using Artificial Neural Networks And Machine Learning. 2022 International Conference on Computer Communication and Informatics (ICCCI), Jan. 25-27, Coimbatore, India. Available at:

https://www.ieee.org (Accessed 23 June 2024).