Machine Learning Engineer
Nanodegree
Udacity

Capstone project
Mobile price prediction

Table of Contents

Definition

Domain

If you have a mobile company this project will make your life a lot easier. I know that when you try to sell your phones one of the most annoying things is what is the price tag that you should put in each smartphone. Nowadays with this huge numbers of smartphones with different specifications it must be really overwhelming. In this project I am going try to help you solve this problem by using different smart phones prices that we already know. I will try to help you in putting the right price tag on different smart phones to maximize your profits.

Problem statement

In this project I will use historical smart phones data like: sales, weight, resolution, internal memory etc.. to help us predict best price tags for new smart phones so we can maximize our profits.

Usage

- This kind of prediction will help companies to estimate the price of mobiles of an mobile phone based on its specifications.
- Beside that it will help Consumers to verify that they are paying best price according to the specs of the mobile.

Solution approach

Dataset

This dataset is the Mobile price prediction dataset from Kaggle the reason why I chose this dataset is it's a new one so the specifications of the smart phones are new in addition to that it's a regression dataset and this problem is a regression problem so it fits amazingly in our solution.

- Product_id: ID of each cellphone.
- Price: Price of each cellphone.
- Sale: Sales number of each smart phone.
- weight: Weight of each cellphone.
- resolution: Resolution of each cellphone.
- PPI: Phone Pixel Density.
- CPU core: type of CPU core in each cellphone.
- CPU freq: CPU Frequency in each cellphone.
- internal mem: Internal memory of each cellphone.
- ram: RAM of each cellphone.
- RearCam: Resolution of the primary camera of each phone in pixels.

- Front_Cam: Resolution of the front camera of each phone in pixels.

- battery: Capacity of the battery of each phone in mAh.

- thickness: Thickness of each phone in mm.

Platform

- This model here is a very light weight model and it's easy to train we can use it as a part of a website or a mobile application we can also use it as a desktop application.

- This model doesn't need a GPU to run as it's a machine learning model not a deep learning one and the dataset is relatively tiny so we don't need a powerful GPU to train this model your normal pc will train it easily.

Algorithm details

After a lot of experiences the elastic net model of Scikit learn is the best model for this problem so let me just explain to you all the models that we used in this project briefly.

1. Ridge Regression model: Ridge regression is a method to perform linear regression with fewer chances of a model getting into problems such as underfitting or overfitting.

   a) It is used highly for the treatment of multicollinearity in regression, it means when an independent variable is correlated in such a way that both resemble each other.

   b) It causes high variance among the independent variables, we can change the value of the independent variable but it will cause a loss of information.

2. Lasso Regression model: Lasso stands for Least Absolute Shrinkage Selector Operator.

   a) It works the same as ridge regression when it comes to assigning the penalty for coefficient.

b) It removes the coefficient and the variables with the help of this process and limits the bias through the below formula.

3. Elastic Net Regression model: Coefficient to the variables are considered to be information that must be relevant, however, ridge regression does not promise to remove all irrelevant coefficient which is one of its disadvantages over Elastic Net Regression(ENR).

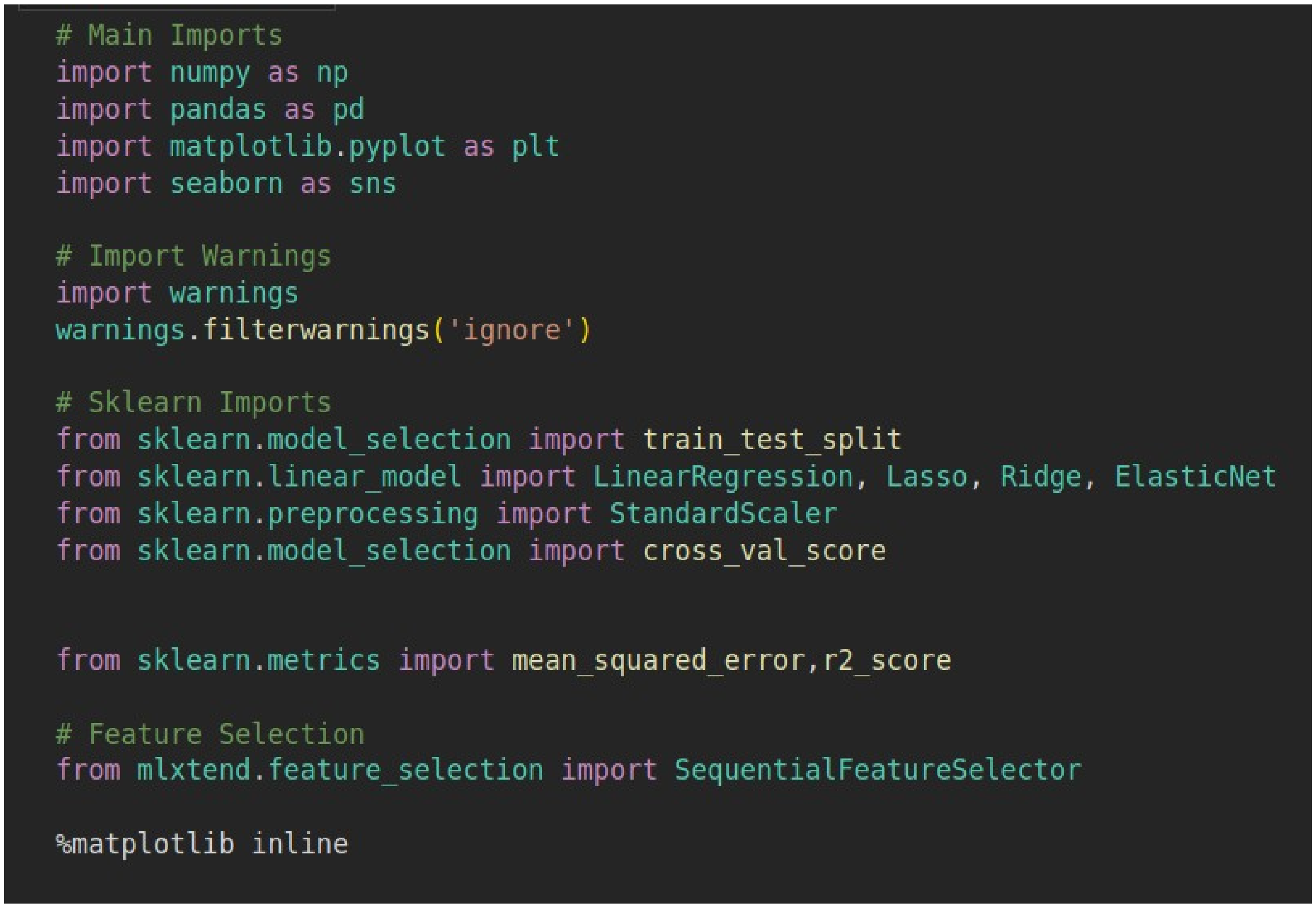
   a) It uses both Lasso as well as Ridge Regression regularization in order to remove all unnecessary coefficients but not the informative ones.

   b) ENR = Lasso Regression + Ridge Regression.

Cleaning

Importing the libraries:

This is the first step of the project in this step we import all the libraries that we need to import the dataset, clean it, transform it and off course visualize it and below here is a screenshot of all the imported libraries of our project.



```
# Main Imports
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Import Warnings
import warnings
warnings.filterwarnings('ignore')

# Sklearn Imports
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression, Lasso, Ridge, ElasticNet
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import cross_val_score


from sklearn.metrics import mean_squared_error,r2_score

# Feature Selection
from mlxtend.feature_selection import SequentialFeatureSelector

%matplotlib inline
```

Importing the dataset:

In this step we import our dataset from a CSV format in a text file into a data-frame format using pandas library this step is necessary to let us clean, transform, analyze, and visualize the dataset easily using python.

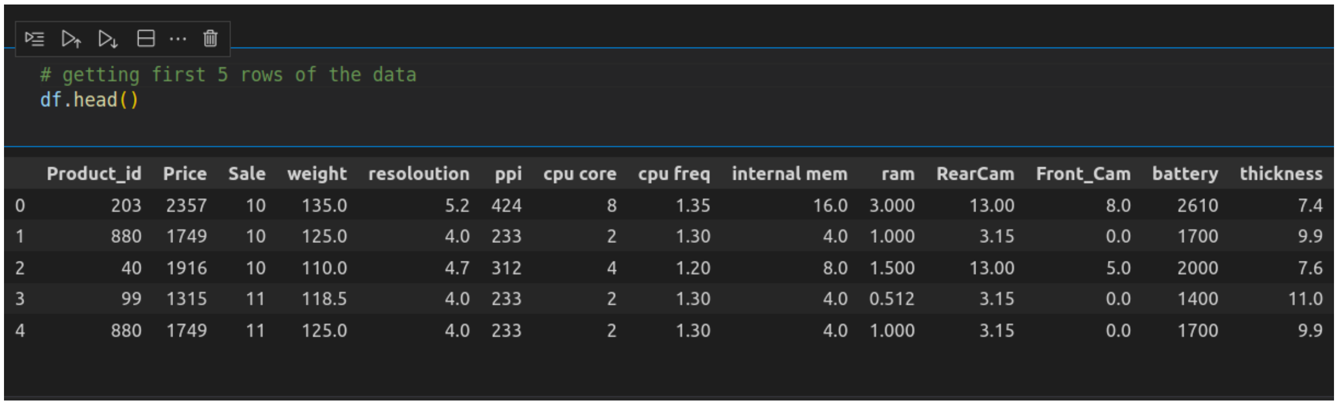
Understanding the dataset:

In this step we start understand our dataset more we use pandas to get useful insights about our dataset and this will help us in our decisions in the cleaning process and here below I will show you some screenshot of the useful information we get using pandas.

Here is the steps of how I am going to understand the dataset:

1. first 5 rows
2. Last 5 rows
3. basic information about the data
4. Full information about the data
5. number of null values in each column
6. Size of the data

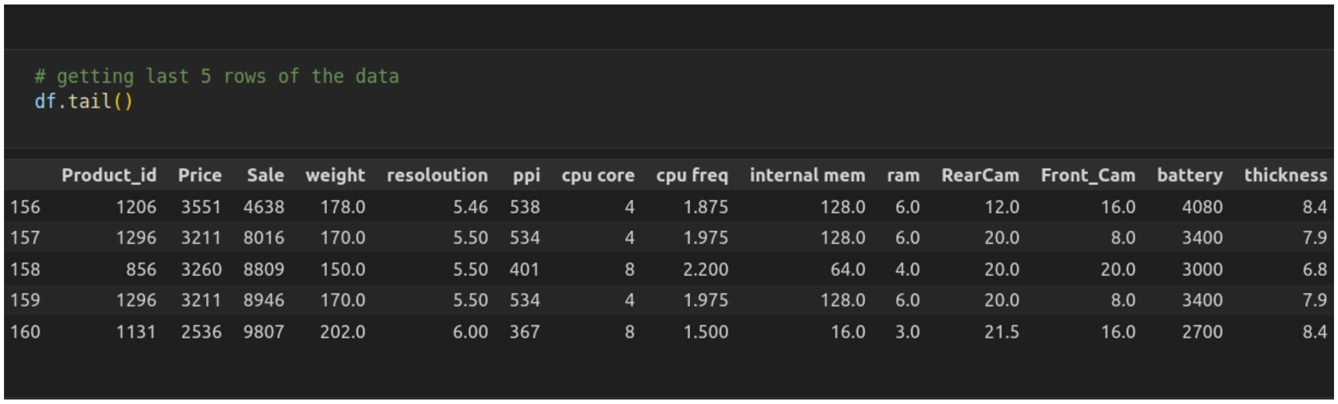
first 5 rows



```
# getting first 5 rows of the data
df.head()
```

	Product_id	Price	Sale	weight	resoloution	ppi	cpu core	cpu freq	internal mem	ram	RearCam	Front_Cam	battery	thickness
0	203	2357	10	135.0	5.2	424	8	1.35	16.0	3.000	13.00	8.0	2610	7.4
1	880	1749	10	125.0	4.0	233	2	1.30	4.0	1.000	3.15	0.0	1700	9.9
2	40	1916	10	110.0	4.7	312	4	1.20	8.0	1.500	13.00	5.0	2000	7.6
3	99	1315	11	118.5	4.0	233	2	1.30	4.0	0.512	3.15	0.0	1400	11.0
4	880	1749	11	125.0	4.0	233	2	1.30	4.0	1.000	3.15	0.0	1700	9.9

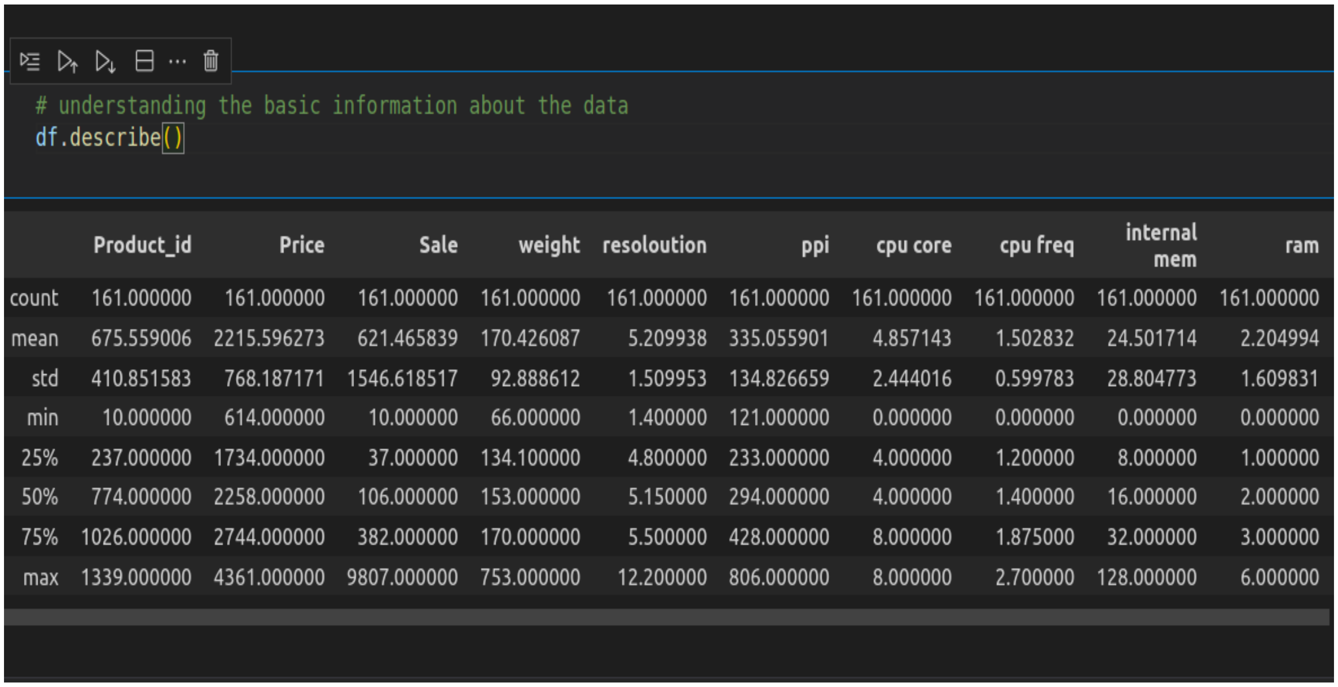
Last 5 rows



```
# getting last 5 rows of the data
df.tail()
```

	Product_id	Price	Sale	weight	resoloution	ppi	cpu core	cpu freq	internal mem	ram	RearCam	Front_Cam	battery	thickness
156	1206	3551	4638	178.0	5.46	538	4	1.875	128.0	6.0	12.0	16.0	4080	8.4
157	1296	3211	8016	170.0	5.50	534	4	1.975	128.0	6.0	20.0	8.0	3400	7.9
158	856	3260	8809	150.0	5.50	401	8	2.200	64.0	4.0	20.0	20.0	3000	6.8
159	1296	3211	8946	170.0	5.50	534	4	1.975	128.0	6.0	20.0	8.0	3400	7.9
160	1131	2536	9807	202.0	6.00	367	8	1.500	16.0	3.0	21.5	16.0	2700	8.4

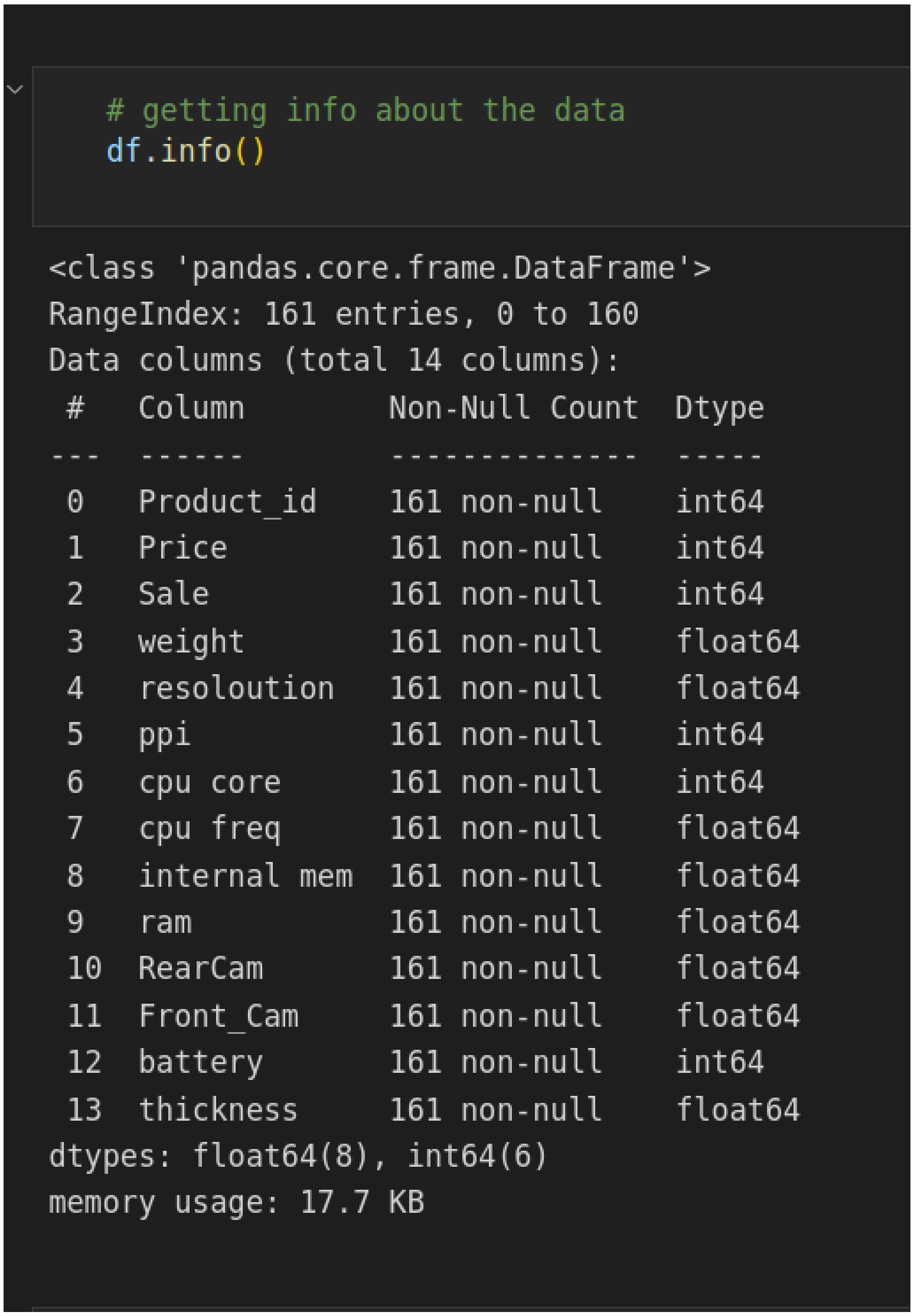
Basic information about the data



```python
# understanding the basic information about the data
df.describe()
```

	Product_id	Price	Sale	weight	resoloution	ppi	cpu core	cpu freq	internal mem	ram
count	161.000000	161.000000	161.000000	161.000000	161.000000	161.000000	161.000000	161.000000	161.000000	161.000000
mean	675.559006	2215.596273	621.465839	170.426087	5.209938	335.055901	4.857143	1.502832	24.501714	2.204994
std	410.851583	768.187171	1546.618517	92.888612	1.509953	134.826659	2.444016	0.599783	28.804773	1.609831
min	10.000000	614.000000	10.000000	66.000000	1.400000	121.000000	0.000000	0.000000	0.000000	0.000000
25%	237.000000	1734.000000	37.000000	134.100000	4.800000	233.000000	4.000000	1.200000	8.000000	1.000000
50%	774.000000	2258.000000	106.000000	153.000000	5.150000	294.000000	4.000000	1.400000	16.000000	2.000000
75%	1026.000000	2744.000000	382.000000	170.000000	5.500000	428.000000	8.000000	1.875000	32.000000	3.000000
max	1339.000000	4361.000000	9807.000000	753.000000	12.200000	806.000000	8.000000	2.700000	128.000000	6.000000

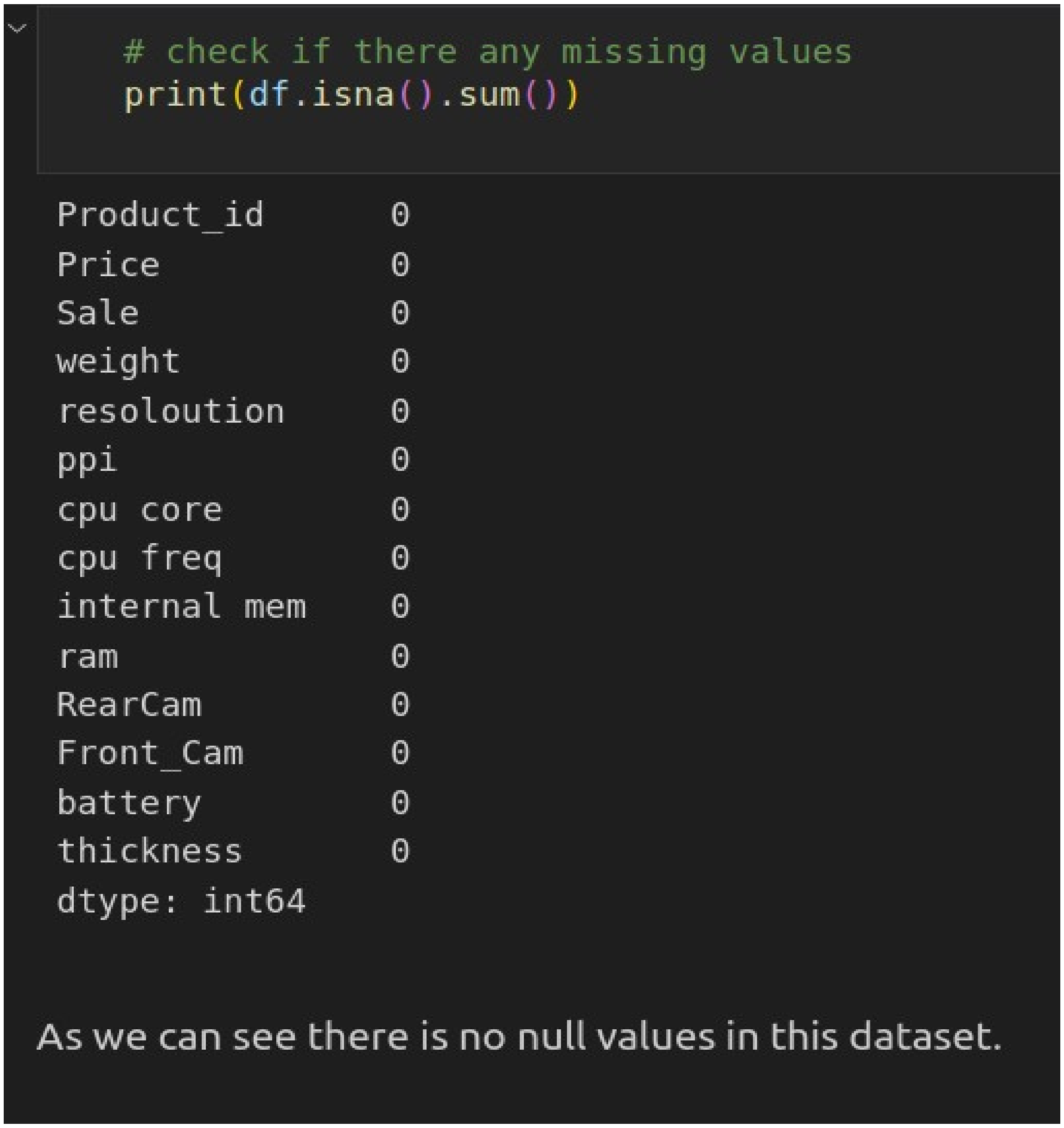
Full information about the data



```
# getting info about the data
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 161 entries, 0 to 160
Data columns (total 14 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   Product_id    161 non-null    int64  
 1   Price         161 non-null    int64  
 2   Sale          161 non-null    int64  
 3   weight        161 non-null    float64
 4   resoloution   161 non-null    float64
 5   ppi           161 non-null    int64  
 6   cpu core      161 non-null    int64  
 7   cpu freq      161 non-null    float64
 8   internal mem  161 non-null    float64
 9   ram           161 non-null    float64
 10  RearCam       161 non-null    float64
 11  Front_Cam     161 non-null    float64
 12  battery       161 non-null    int64  
 13  thickness     161 non-null    float64
dtypes: float64(8), int64(6)
memory usage: 17.7 KB
```

Number of null values in each column



```python
# check if there any missing values
print(df.isna().sum())
```

```
Product_id      0
Price           0
Sale            0
weight          0
resoloution     0
ppi             0
cpu core        0
cpu freq        0
internal mem    0
ram             0
RearCam         0
Front_Cam       0
battery         0
thickness       0
dtype: int64
```

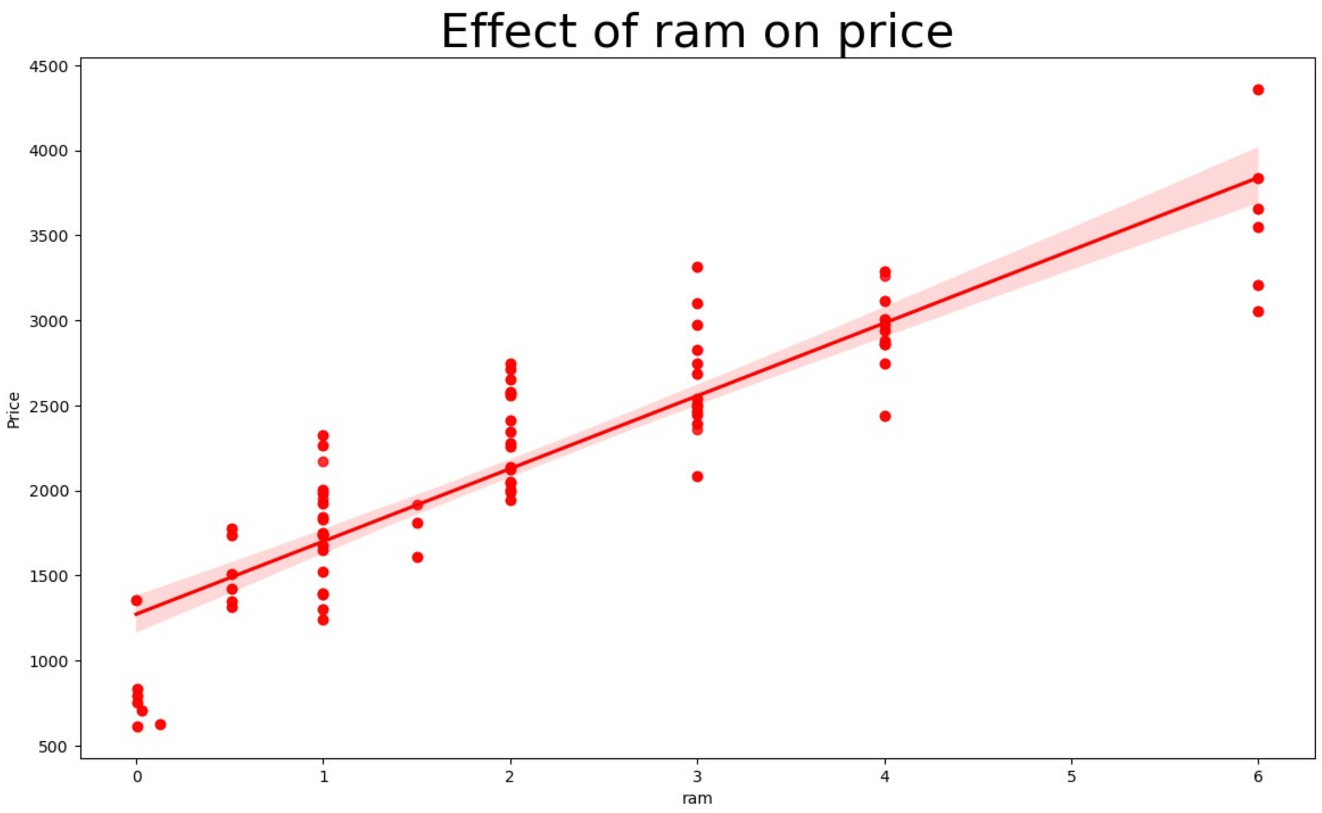
As we can see there is no null values in this dataset.

Size of the data

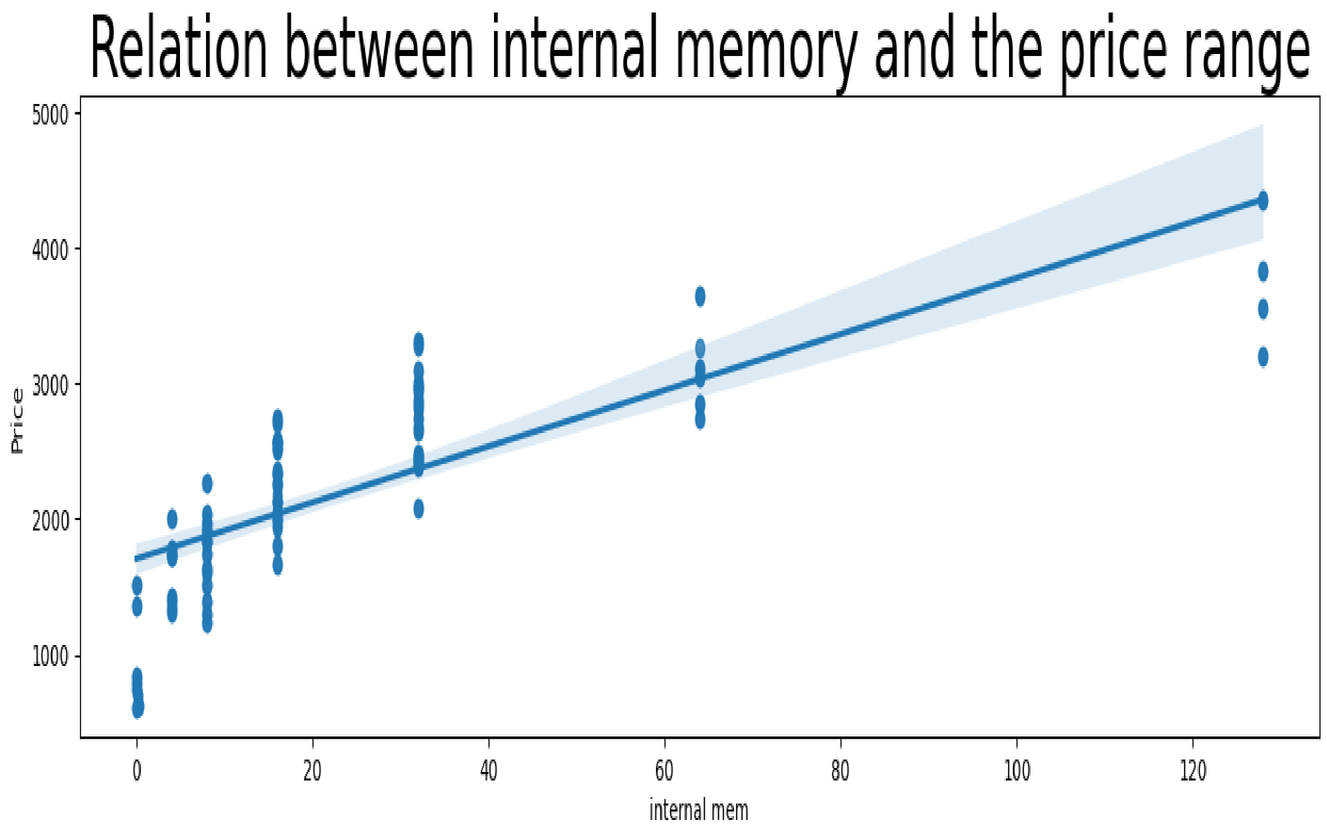
- 161 rows
- 14 columns

Analysis and visualization

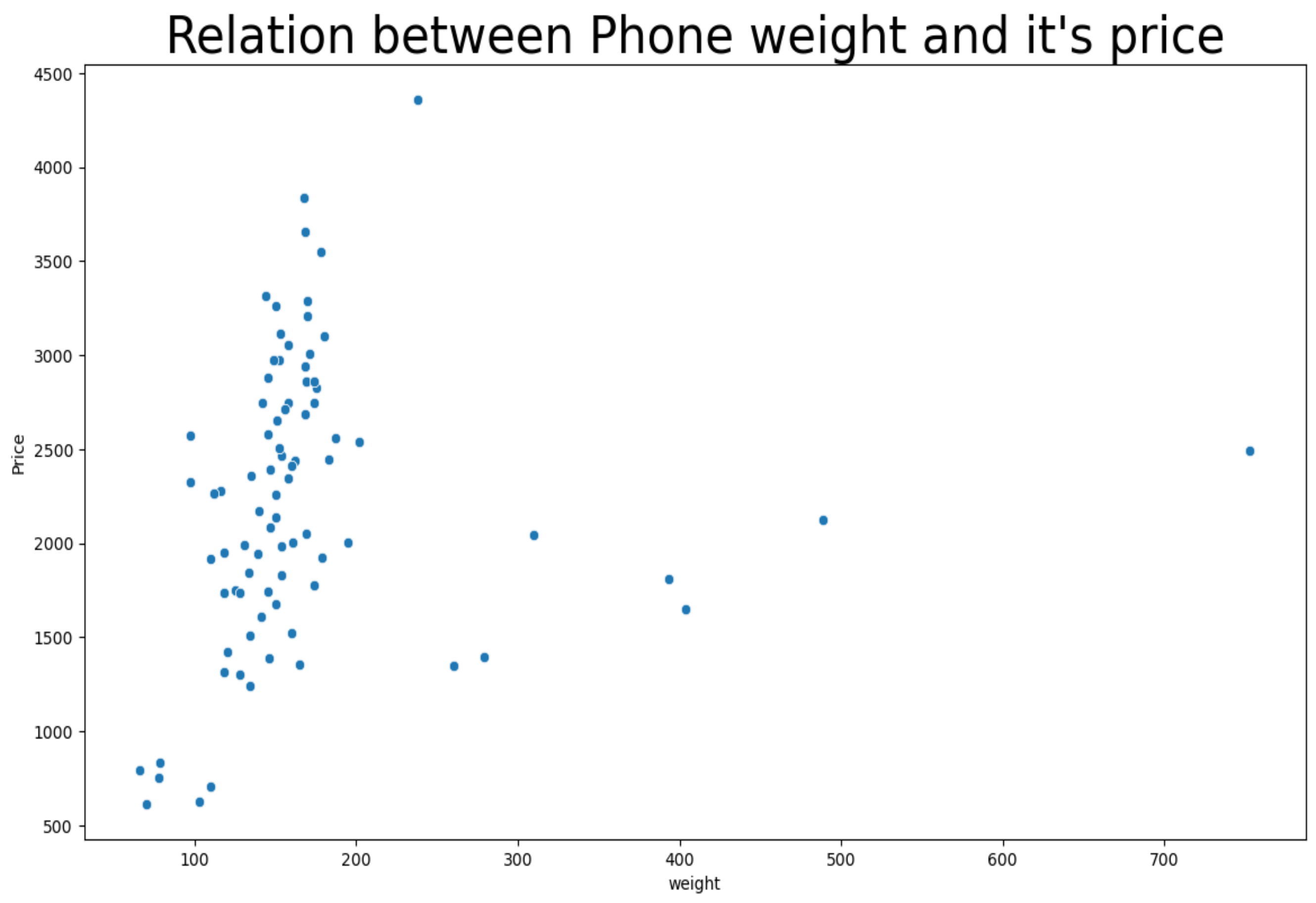
How does ram affect the price ?


Effect of ram on price
Price
ram
4500
4000
3500
3000
2500
2000
1500
1000
500
0
1
2
3
4
5
6

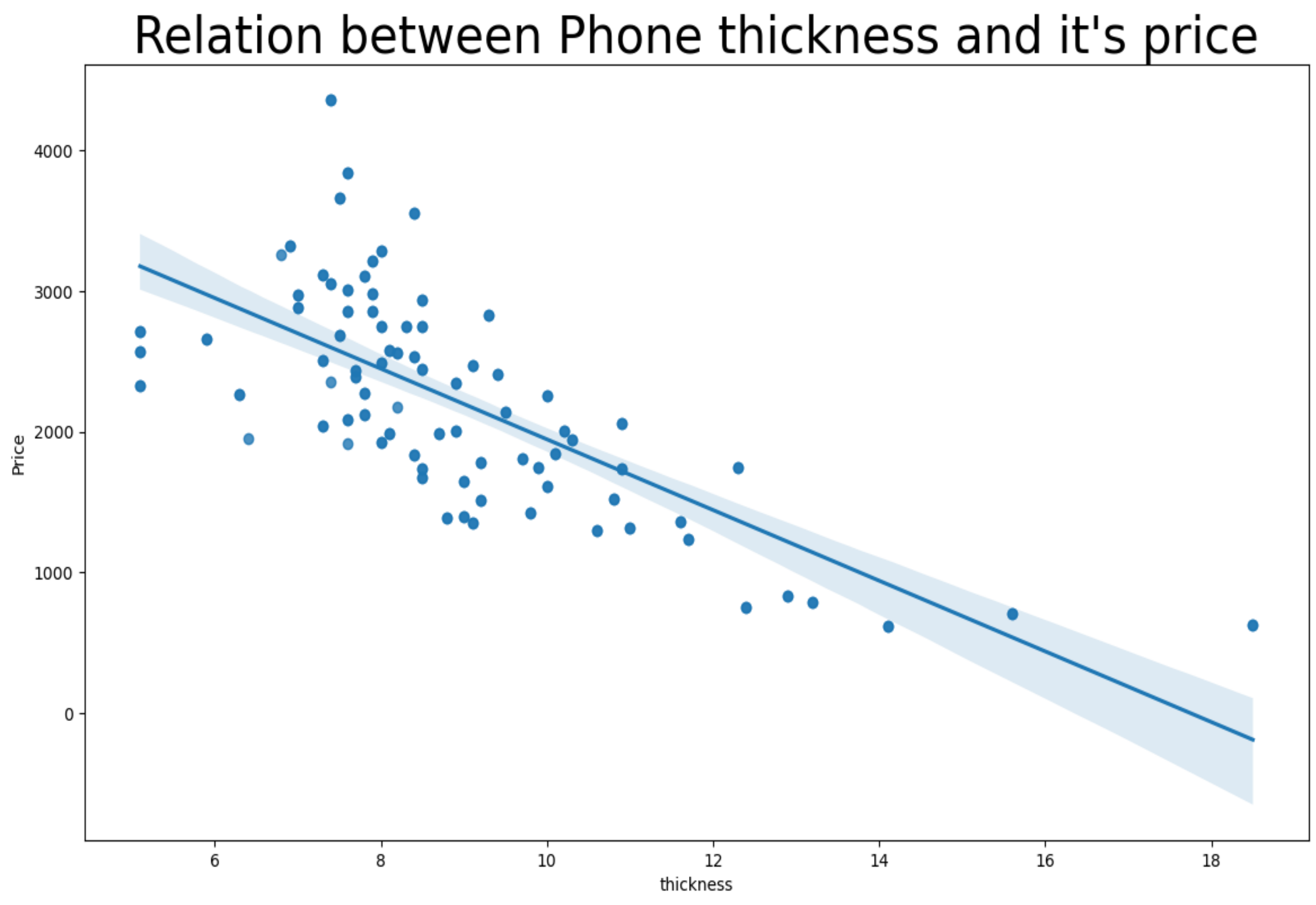
What is the relation between Internal Memory and Price Range ?



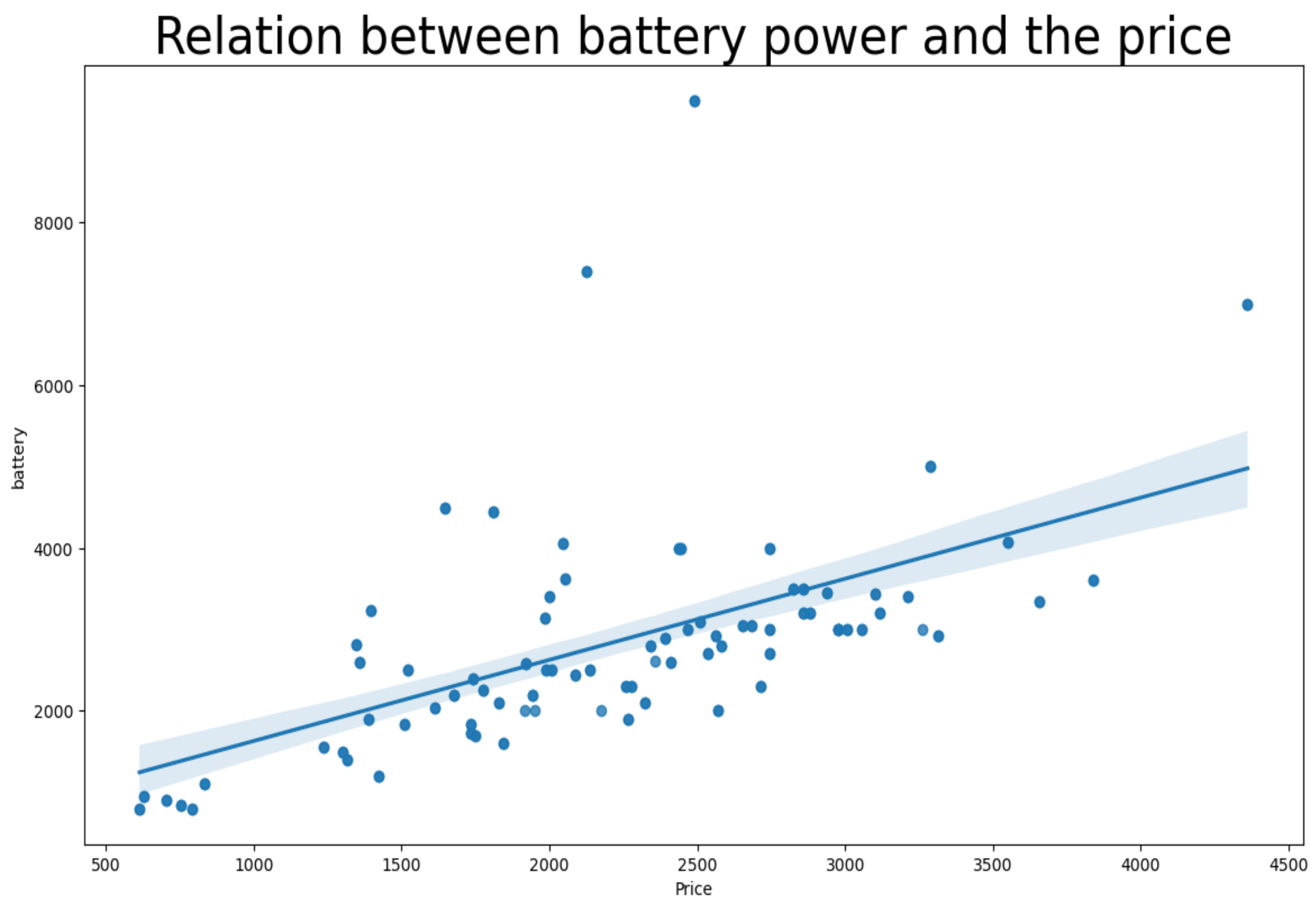
Wat is the relation between Phone weight and it's price ?



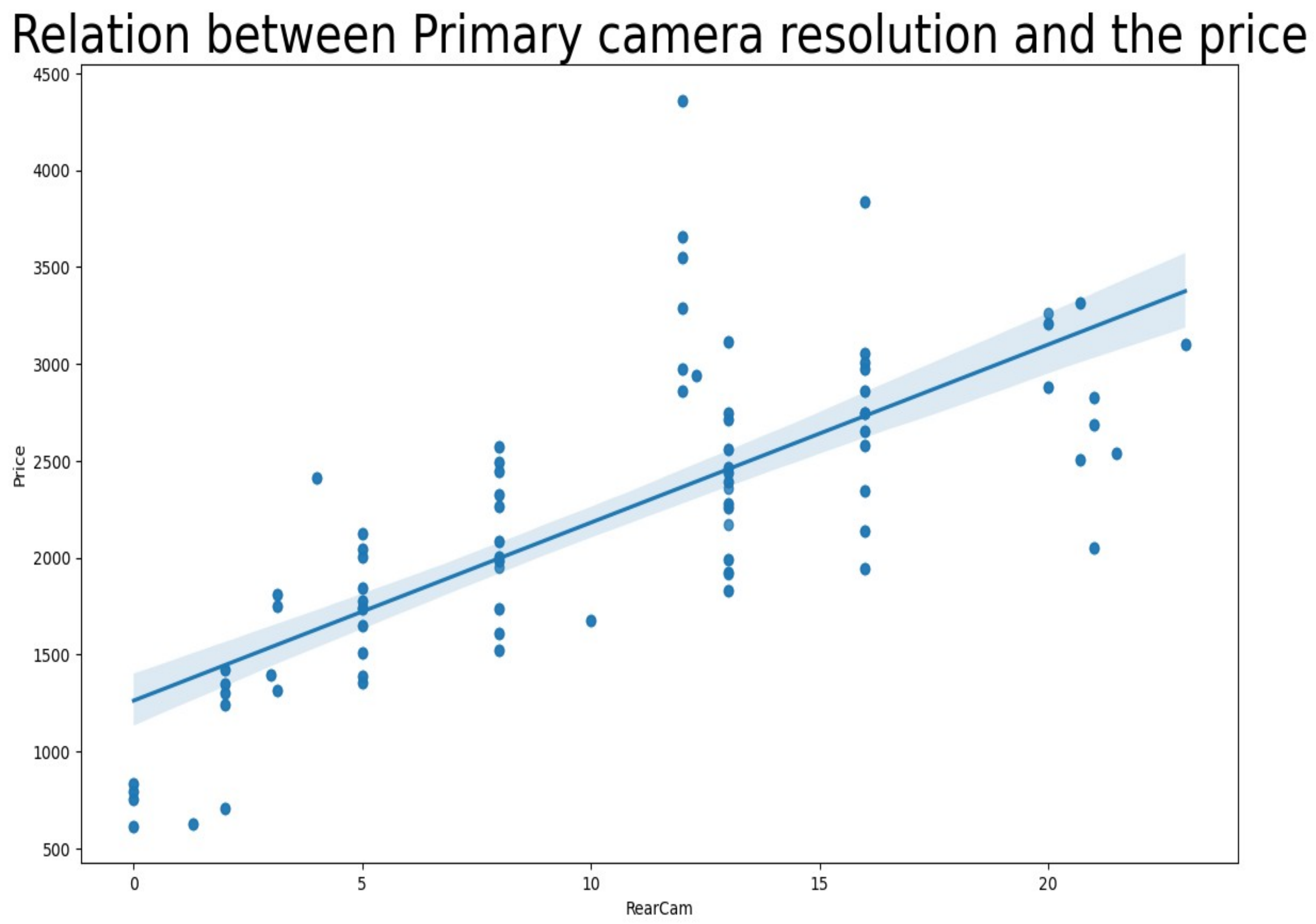
What is the relation between Phone thickness and it's price ?



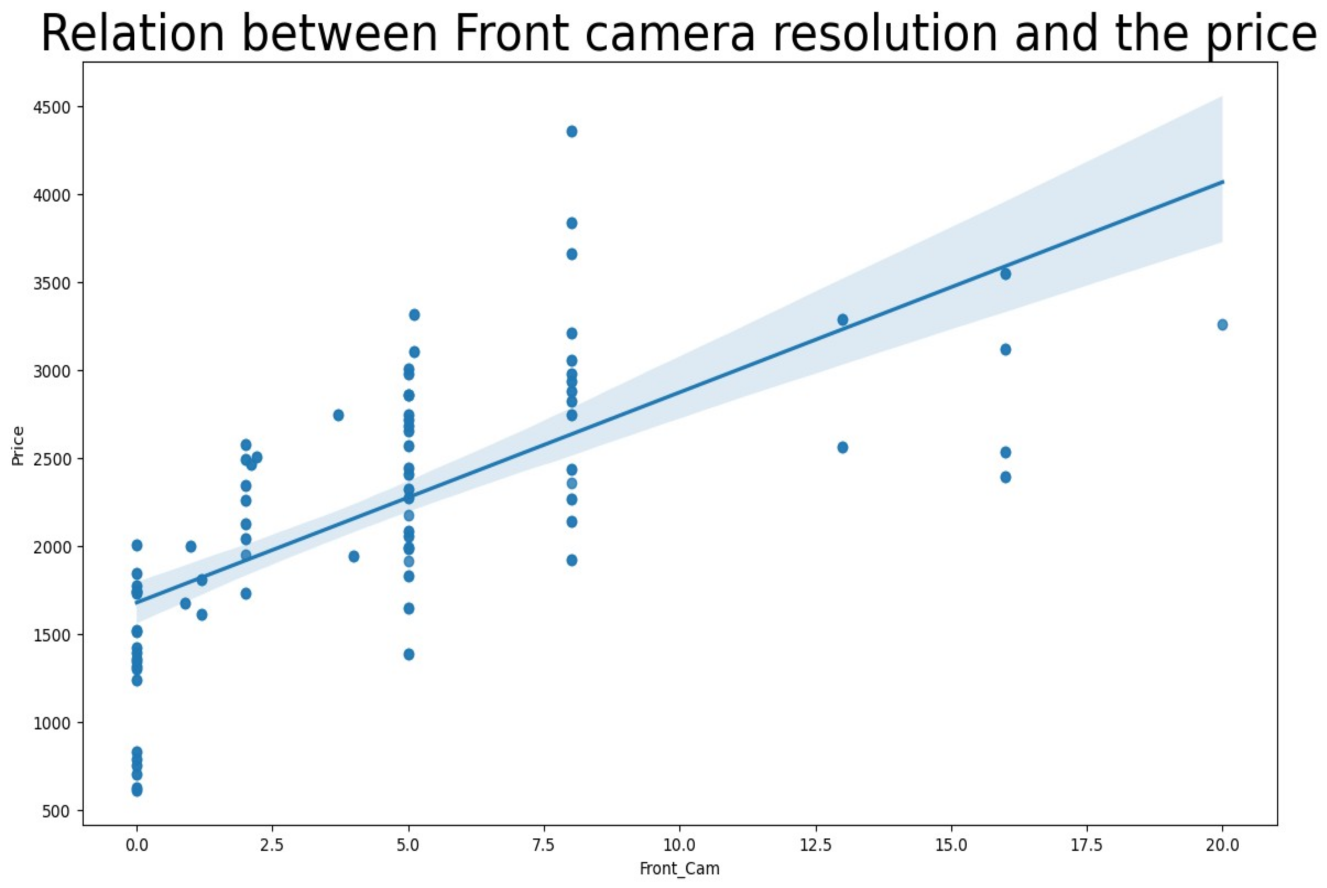
What is the relation between Battery power and Price Range ?



What is the relation between primary camera resolution and the phone price ?



What is the relation between Front camera resolution and the phone price ?



Data preprocessing

In this step I start prepare my dataset for the machine learning phase in this steps:

- First step I divide my dataset into two variables features and target.
- second step I split my data in train and test datasets.
- Third step I normalize the columns

Evaluation metrics

In this project I have used 2 different metrics which are:

- Root mean squared error: this metric the smaller the number we have the better the model the reason why I chose this metric is because it is a regression metric and this problem here is a regression problem.

- r2_score: this metric is easier to understand for people outside of machine learning as this metric is a percentage and the bigger the percentage the better the model the reason why I chose this metric is to let other people outside of machine learning understand which model is better easily.

Benchmark model

The benchmark model for this project is the Scikit learn linear regression model where after cleaning and normalizing the dataset I was able to achieve:

- 0.901 r2_score
- 218.7 root mean squared error.

Model improvement

After trying the normal model on all the columns I decided to improve my model by doing the following:

1. Feature selection
2. Changing the architecture of the model

Feature selection

In this step I have used mlxtend sequential feature selector to tell me which features have the most impact on the price of each phone and these columns are:

- weight
- resolution
- PPI
- CPU core
- CPU frequency
- internal memory
- ram
- front camera pixels
- battery
- thickness

Final model

My final model will be the elastic net regression model as by experiment it outperforms all the other models and the reason this model is very effective and I have chosen this model is it uses both Lasso as well as Ridge Regression regularization in order to remove all unnecessary coefficients but not the informative ones.

The scores of the final model:

- 0.9034 r2_score

- 216.71 Root mean squared error

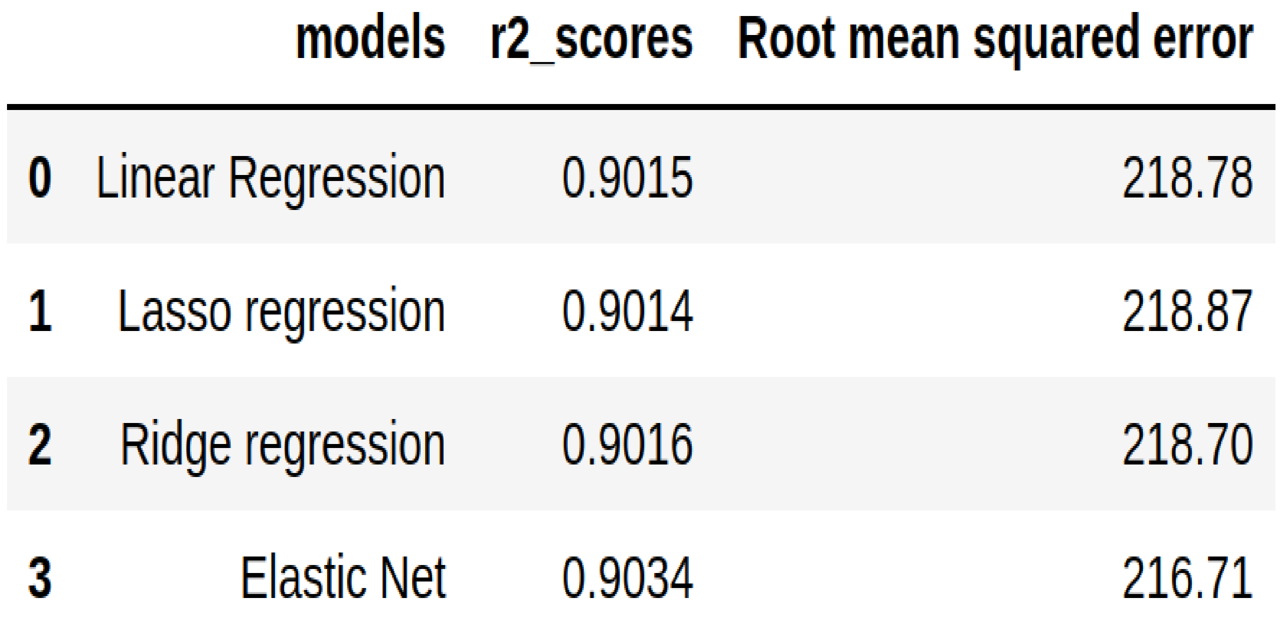
Comparison between benchmark model and final model.

Benchmark model performance:

- 0.9015 on the r2_score metric
- 218.78 root mean squared error

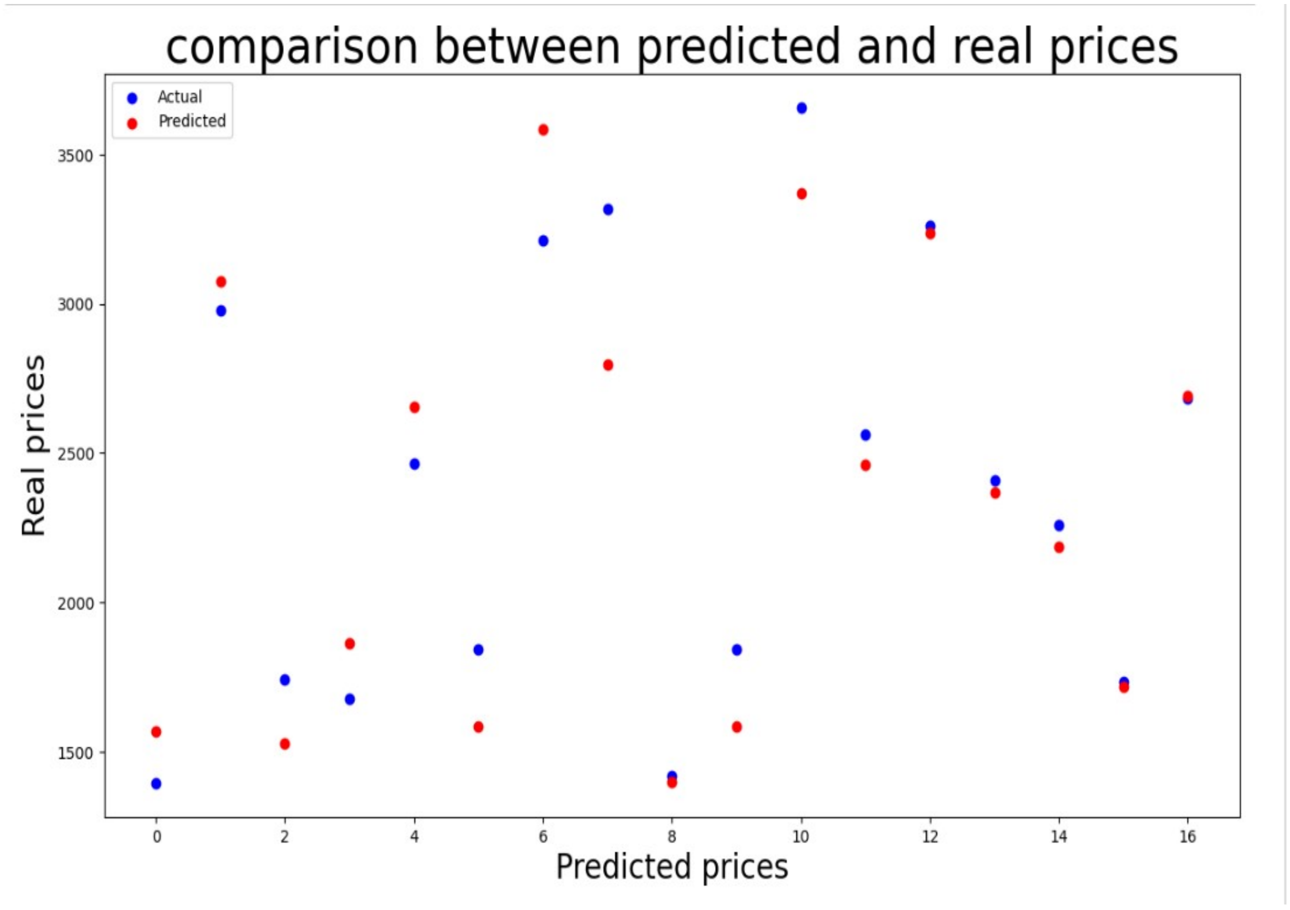
Final model performance:

- 0.0.9034 on the r2_score metric
- 216.71 root mean squared error



	models	r2_scores	Root mean squared error
0	Linear Regression	0.9015	218.78
1	Lasso regression	0.9014	218.87
2	Ridge regression	0.9016	218.70
3	Elastic Net	0.9034	216.71

Comparison between real prices and predicted prices using my final model.


comparison between predicted and real prices
Actual
Predicted
Real prices
Predicted prices