Price Prediction for Mobile Phones

Machine learning Internship

Introduction:

Price is the most effective attribute of marketing and business. The very first question of costumer is about the price of items. All the costumers are first worried and thinks “If he would be able to purchase something with given specifications or not”. So, to estimate price at home is the basic purpose of the work. Artificial Intelligence-which makes machine capable to answer these questions intelligently- now a days is very vast in engineering field. Machine learning provides us best techniques for artificial intelligence like classification, regression, supervised learning and unsupervised learning and many more. We can use any of classifiers like Decision tree, Naïve Bayes and many more. Different type of feature selection algorithms is available to select only best features and minimize dataset. This will reduce computational complexity of the problem. As this is optimization problem so many optimization techniques are also used to reduce dimensionality of the dataset. Mobile now a days is one of the most selling and purchasing device. Every day new mobiles with new version and more features are launched. Hundreds and thousands of mobiles are sold and purchased on daily basis. So here the mobile price class prediction is a case study for the given type of problem i.e., finding optimal product.

Dataset details and Exploration:

- 1.Battery_power: Total energy a battery can store in one time measured in mAh
- 2. Blue: Has Bluetooth or not
- 3. dual sim: Has dual sim support or not
- 4. fc: Front Camera mega pixels
- 5. m_dep: Mobile Depth in cm
- 6. mobile_wt: Weight of mobile phone
- 7. n_cores: Number of cores of processor
- 8. pc: Primary Camera mega pixels
- 9. px_height: Pixel Resolution Height Tail
- 10. px_width: Pixel Resolution Width
- 11.RAM: Random Access Memory in Megabytes
- 12. sc_h: Screen Height of mobile in cm
- 13. sc_w: Screen Width of mobile in cm
- The output variable is the price range:
- 0: (low cost),
- 1: (medium cost),
- 2: (high cost), and
- 3: (very high cost).

1 : Data Exploration:

7]:

```
DS1 = pd.read_csv(r"C:\Users\Anuja Verma\Downloads\Processed_Flipdata - Processed_Flipdata.csv")
DS1
```

7]:

	Unnamed: 0	Memory	RAM	Battery_	AI Lens	Mobile Height	Model_APPLE iPhone 11	Model_APPLE iPhone 12	Model_APPLE iPhone 14 Plus	Model_Google Pixel 6a	Model_Google Pixel 7	Model_Google Pixel 7a	Model_I Kall Z19Pro	Model_I Kall Z19Pro Flash blue
0	0	64	4	6000	1	16.76	False	False	False	False	False	False	False	False
1	1	64	4	6000	1	16.76	False	False	False	False	False	False	False	False
2	2	128	8	5000	0	16.64	False	False	False	False	False	False	False	False
3	3	32	2	5000	0	16.56	False	False	False	False	False	False	False	False
4	4	128	8	5000	1	16.76	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
536	637	256	8	3900	0	15.49	False	False	False	False	False	False	False	False
537	638	32	2	3100	0	12.70	False	False	False	False	False	False	False	False
538	639	64	4	5000	0	16.76	False	False	False	False	False	False	False	False
539	641	128	8	5000	0	16.26	False	False	False	False	False	False	False	False
540	642	128	4	5000	0	16.66	False	False	False	False	False	False	False	False

541 rows × 776 columns

```
DS1.head()
```

	Unnamed: 0	Memory	RAM	Battery_	AI Lens	Mobile Height	Model_APPLE iPhone 11	Model_APPLE iPhone 12	Model_APPLE iPhone 14 Plus	Model_Google Pixel 6a	Model_Google Pixel 7	Model_Google Pixel 7a	Model_I Kall Z19Pro	Model_I Kall Z19Pro Flash blue
0	0	64	4	6000	1	16.76	False	False	False	False	False	False	False	False
1	1	64	4	6000	1	16.76	False	False	False	False	False	False	False	False
2	2	128	8	5000	0	16.64	False	False	False	False	False	False	False	False
3	3	32	2	5000	0	16.56	False	False	False	False	False	False	False	False
4	4	128	8	5000	1	16.76	False	False	False	False	False	False	False	False

```
columns = DS1.columns

columns
```

```
Index(['Unnamed: 0', 'Memory', 'RAM', 'Battery_', 'AI Lens', 'Mobile Height',
       'Model_APPLE iPhone 11', 'Model_APPLE iPhone 12',
       'Model_APPLE iPhone 14 Plus', 'Model_Google Pixel 6a',
       ...
       'Prize_9,104', 'Prize_9,290', 'Prize_9,349', 'Prize_9,387',
       'Prize_9,499', 'Prize_9,699', 'Prize_9,790', 'Prize_9,990',
       'Prize_9,999', 'Prize_920'],
      dtype='object', length=776)
```

```
      dtype='object', length=776)
```

```
DS1.describe().T
```

	count	mean	std	min	25%	50%	75%	max
Unnamed: 0	541.0	289.711645	182.359185	0.0	135.00	273.00	434.00	642.00
Memory	541.0	110.550832	60.600694	16.0	64.00	128.00	128.00	256.00
RAM	541.0	5.397412	1.984923	2.0	4.00	6.00	8.00	8.00
Battery_	541.0	4871.587800	780.148862	800.0	5000.00	5000.00	5000.00	7000.00
AI Lens	541.0	0.062847	0.242911	0.0	0.00	0.00	0.00	1.00
Mobile Height	541.0	16.431201	2.523553	4.5	16.51	16.71	16.94	41.94

From the above description, I can see the minnimum value for columns 'sc_w' (screen width) and for 'px_h
life. We need handle these discrepencies in data and will have to replace these values

```
DS1.tail()
```

	Unnamed: 0	Memory	RAM	Battery_	AI Lens	Mobile Height	Model_APPLE iPhone 11	Model_APPLE iPhone 12	Model_APPLE iPhone 14 Plus	Model_Goog Pixel
536	637	256	8	3900	0	15.49	False	False	False	Fal
537	638	32	2	3100	0	12.70	False	False	False	Fal
538	639	64	4	5000	0	16.76	False	False	False	Fal
539	641	128	8	5000	0	16.26	False	False	False	Fal

3. Feature Extraction: ¶

A Statistical Analysis

To Find Mean, Media,& Mode per column

```
DS1.select_dtypes(include=['number']).mean()
```

```
Unnamed: 0        289.711645
Memory            110.550832
RAM                 5.397412
Battery_         4871.587800
AI Lens             0.062847
Mobile Height      16.431201
dtype: float64
```

```
DS1.select_dtypes(include=['number']).median()
```

```
Unnamed: 0        273.00
Memory            128.00
RAM                 6.00
Battery_         5000.00
AI Lens             0.00
Mobile Height      16.71
dtype: float64
```

```
DS1.select_dtypes(include=['number']).mode()
```

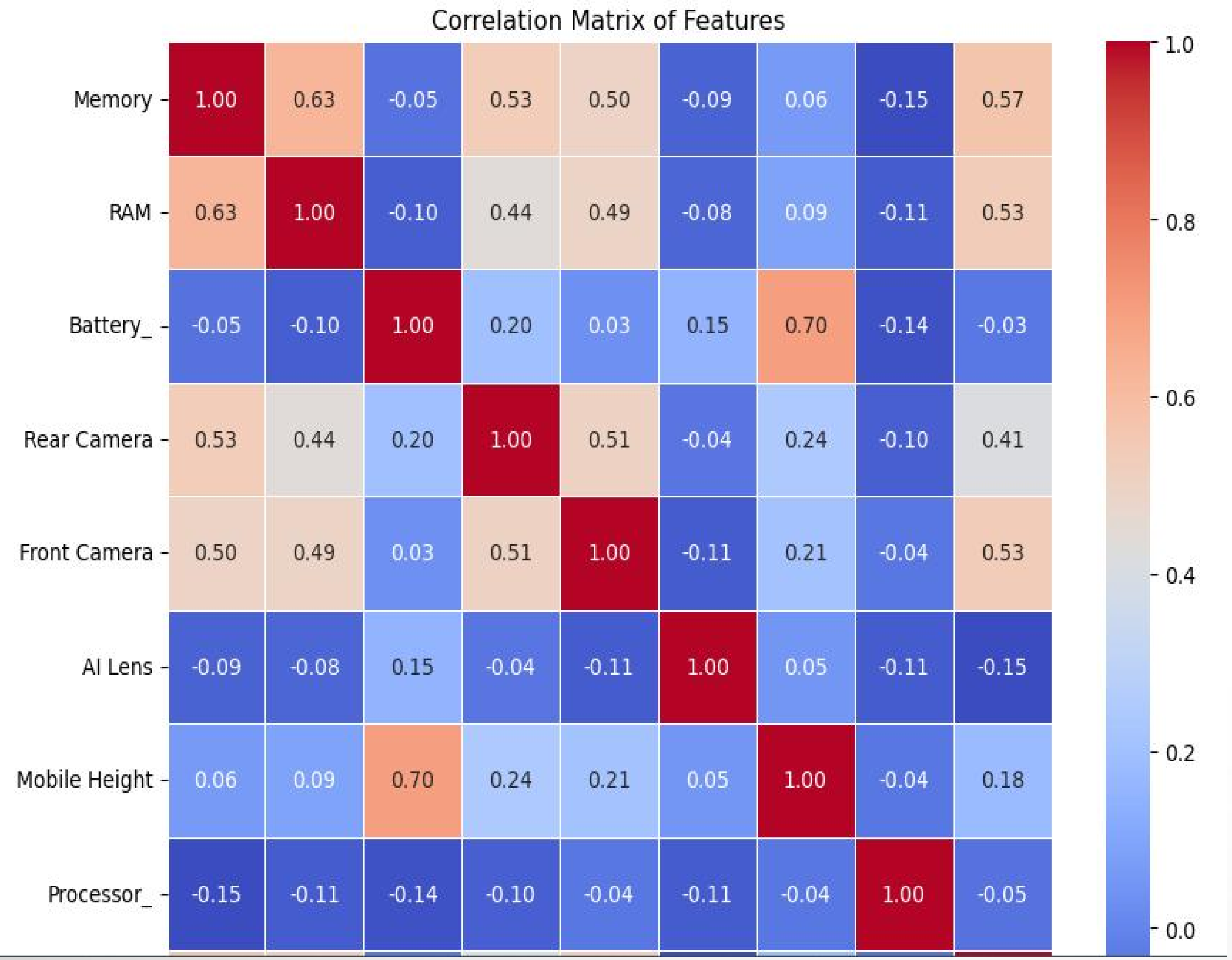
	Unnamed: 0	Memory	RAM	Battery_	AI Lens	Mobile Height
0	0	128.0	4.0	5000.0	0.0	16.76
1	1	NaN	NaN	NaN	NaN	NaN

Visualizations

```
# Drop unnecessary columns
DS1_clean = DS1.drop(columns=['Unnamed: 0','Model', 'Colour'])

# Convert 'Prize' to numeric by removing commas and casting to int
DS1_clean['Prize'] = DS1_clean['Prize'].astype(str).str.replace(',', '').astype(int)

# Convert 'Rear Camera' and 'Front Camera' from string (e.g., '13MP') to int
DS1_clean['Rear Camera'] = DS1_clean['Rear Camera'].astype(str).str.replace('MP', '').astype(int)
DS1_clean['Front Camera'] = DS1_clean['Front Camera'].astype(str).str.replace('MP', '').astype(int)

# Encode 'Processor_' using label encoding (for simplicity)
from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()
DS1_clean['Processor_'] = le.fit_transform(DS1_clean['Processor_'])

# Preview cleaned data
DS1_clean.head()
```

	Memory	RAM	Battery_	Rear Camera	Front Camera	AI Lens	Mobile Height	Processor_	Prize
0	64	4	6000	13	5	1	16.76	113	7299
1	64	4	6000	13	5	1	16.76	113	7299
2	128	8	5000	50	16	0	16.64	75	11999
3	32	2	5000	8	5	0	16.56	56	5649
4	128	8	5000	50	5	1	16.76	14	8999

```python
correlation_matrix = DS1_clean.corr(numeric_only=True)

# Visualize correlation matrix with a heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f", linewidths=0.5)
plt.title("Correlation Matrix of Features")
plt.show()

# Sort features by correlation with 'Prize'
price_corr = correlation_matrix['Prize'].drop('Prize').sort_values(ascending=False)
price_corr
```



62]:
```
Memory           0.566660
Front Camera     0.532321
RAM              0.532024
Rear Camera      0.410367
Mobile Height    0.176009
Battery_        -0.034297
Processor_      -0.050244
AI Lens         -0.153691
Name: Prize, dtype: float64
```

4. Model Building:

Split the dataset

Use scikit-learn's train_test_split to divide dataset into training and testing sets.

70]:
```python
X = DS1_clean.drop(columns=['Prize'])  # Features
y = DS1_clean['Prize']  # Target

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Train Random Forest Regressor

Random Forest is an ensemble learning method that builds multiple decision trees and merges their resul

74]:
```python
model = RandomForestRegressor(random_state=42)
model.fit(X_train, y_train)
```

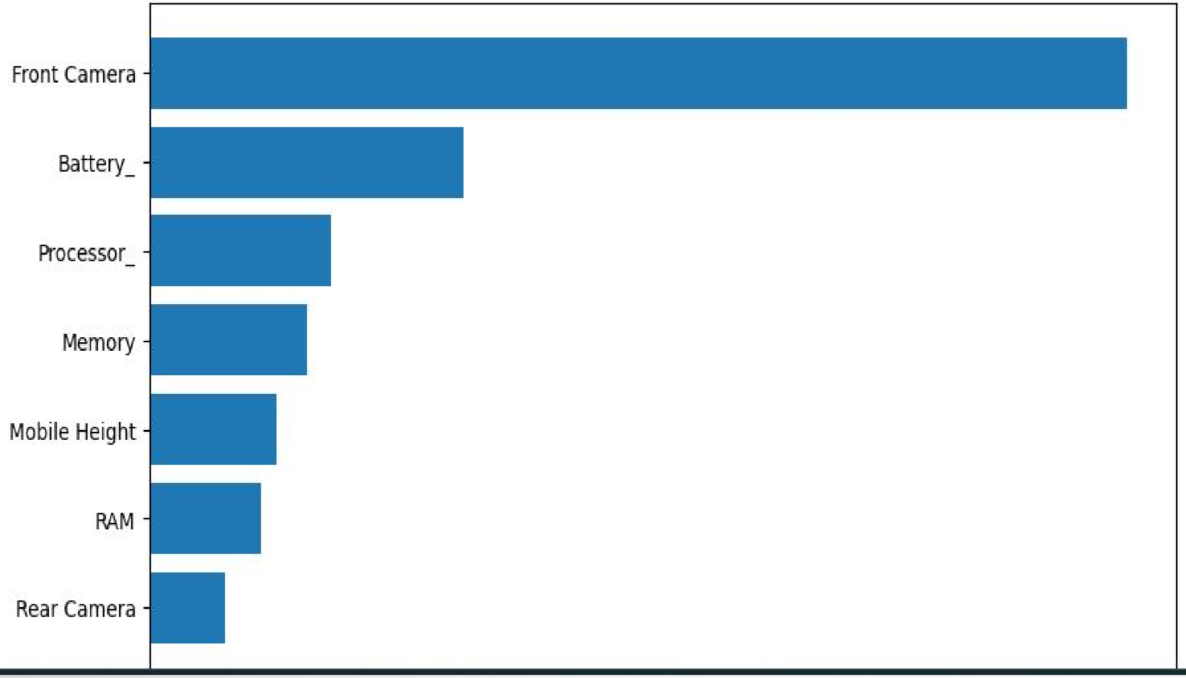
74]:
```
RandomForestRegressor
RandomForestRegressor(random_state=42)
```

Evaluate model

Feature Importance

Determine which features most influence the price prediction.

```python
# Get feature importances
importances = model.feature_importances_
feature_names = X.columns

# Create a DataFrame for visualization
feature_importances = pd.DataFrame({'Feature': feature_names, 'Importance': importances})
feature_importances = feature_importances.sort_values(by='Importance', ascending=False)

# Plot feature importances
plt.figure(figsize=(10, 6))
plt.barh(feature_importances['Feature'], feature_importances['Importance'])
plt.xlabel('Importance')
plt.title('Feature Importances')
plt.gca().invert_yaxis()
plt.show()
```



6. Feature Importance Analysis

Analyze and visualize feature importances

```python
# Create a DataFrame of features and their importances
feature_importances = pd.DataFrame({
    'Feature': X.columns,
    'Importance': importances
}).sort_values(by='Importance', ascending=False)

# Display top features
print(feature_importances.head(10))

# Plot
plt.figure(figsize=(10, 6))
plt.barh(feature_importances['Feature'][:10], feature_importances['Importance'][:10])
plt.gca().invert_yaxis()
plt.xlabel('Importance')
plt.title('Top 10 Feature Importances')
plt.tight_layout()
plt.show()
```

```
              Feature  Importance
0          Unnamed: 0    0.086810
2                 RAM    0.058594
1              Memory    0.045300
5       Mobile Height    0.037217
3            Battery_    0.020531
478   Rear Camera_50MP    0.013562
486  Front Camera_16MP    0.010677
492   Front Camera_5MP    0.009379
494   Front Camera_8MP    0.009153
480   Rear Camera_64MP    0.007924
```

Top 10 Feature Importa

CONCLUSION :

This work can be concluded with the comparable results of both Feature selection algorithms and classifier. This combination has achieved maximum accuracy and selected minimum but most appropriate features. It is important to note that in Forward selection by adding irrelevant or redundant features to the data set decreases the efficiency of both classifiers. While in backward selection if we remove any important feature from the data set, its efficiency decreases. The main reason of low

OUTCOMES OF THE WORK:

- Cost prediction is the very important factor of marketing and business. To predict the
- cost same procedure can be performed for all types of products for example Cars,
- Foods, Medicine, Laptops etc.
- Best marketing strategy is to find optimal product (with minimum cost and maximum
- specifications). So products can be compared in terms of their specifications, cost,
- manufacturing company etc.
- By specifying economic range a good product can be suggested to a costumer.

FUTURE WORK EXTENSION :

- More sophisticated artificial intelligence techniques can be used to maximized the accuracy and predict the accurate price of the products.

- Software or Mobile app can be developed that will predict the market price of any new launched product.

- To achieve maximum accuracy and predict more accurate, more and more instances should be added to the data set. And selecting more appropriate features can also increase the accuracy. So data set should be large and more appropriate features should be selected to achieve higher accuracy

Thank You