CAPSTONE PROJECT-3

Mobile Price Range Prediction

TEAM MEMBERS

SATYAM JYOTI SANKAR

KRUSHNAGOPAL BRAHMA

CONTENT

1) Defining problem statement
2) EDA and feature engineering
3) Feature Selection
4) Preparing dataset for modeling
5) Applying Model
6) Model Validation and Selection
7) Conclusion

Problem Statement

- The problem statement is to predict the price range of mobile phones based on the features available (price range indicating how high the price is). Here is the description of target classes:

- 0 - Low cost Phones
- 1 - Medium cost phones
- 2 - High cost phones
- 3 - Very High cost phones

- This will basically help companies to estimate price of mobiles to give tough competition to other mobile manufacturer. Also, it will be useful for consumers to verify that they are paying best price for a mobile.

DATA SUMMARY

➢ **Independent variables :**

- **Independent variables :**
- **Battery_power -** Total energy a battery can store in one time measured in mAh
- **Blue -** Has bluetooth or not
- **Clock_speed** - speed at which microprocessor executes instructions
- **Dual_sim -** Has dual sim support or not
- **Fc -** Front Camera mega pixels
- **Four_g -** Has 4G or not
- **Int_memory -** Internal Memory in Gigabytes
- **M_dep** - Mobile Depth in cm

DATA SUMMARY(cont..)

Mobile_wt - Weight of mobile phone

N_cores - Number of cores of processor

Pc - Primary Camera mega pixels

Px_height - Pixel Resolution Height

Px_width - Pixel Resolution Width

Ram - Random Access Memory in Mega Bytes

Sc_h - Screen Height of mobile in cm

Sc_w - Screen Width of mobile in cm

Talk_time - longest time that a single battery charge will last when you are

DATA SUMMARY(cont..)

Three_g - Has 3G or not

Touch_screen - Has touch screen or not

Wifi - Has wifi or not

Dependent variables :

Price_range - This is the target variable with value of

0(low cost),

1(medium cost),

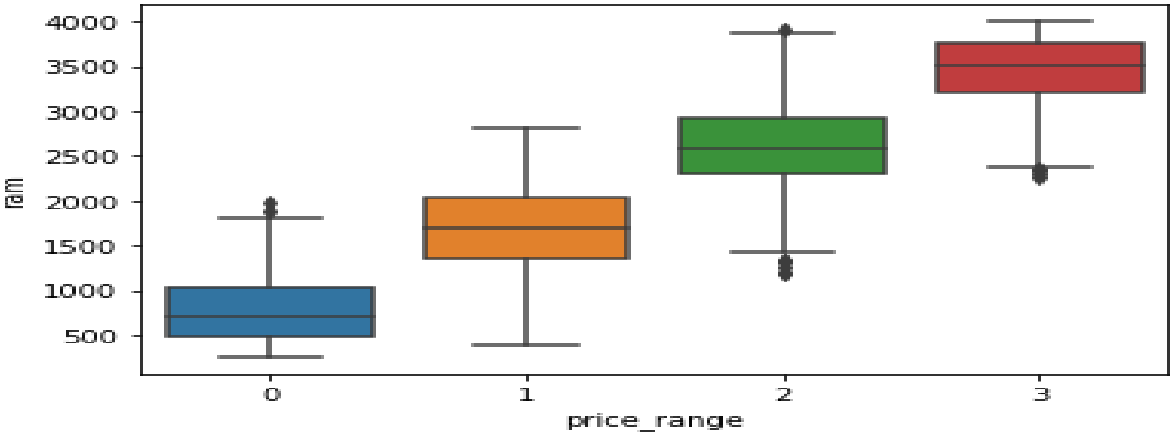
2(high cost)

and 3(very high cost).

EDA(contd...)

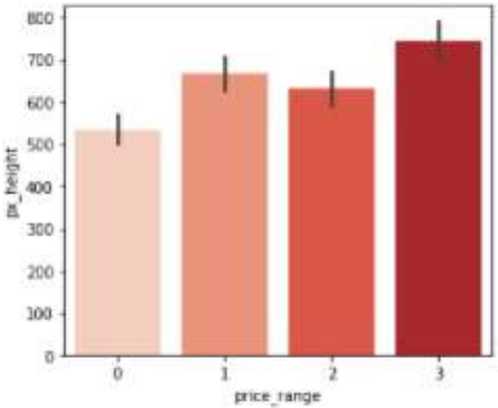
Relation Between Price Range & Ram

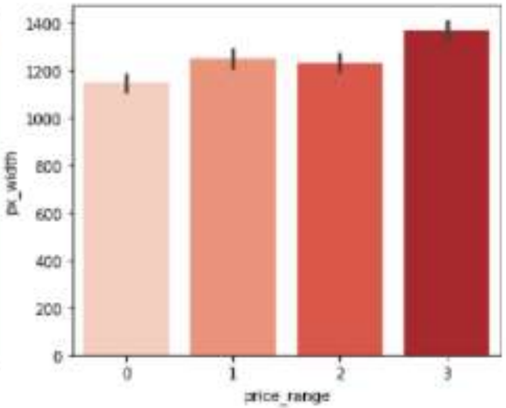
- This is a positive relationship, with increase in RAM, price too increases. There are 4 types of price range
- Type 1(low cost): RAM ranges between 216 to 1974 megabytes.
- Type 2(medium cost): RAM ranges between 387 to 2811 megabytes
- Type 3(high cost): RAM ranges between 1185 to 3916 megabytes
- Type 4(very high cost): RAM ranges between 2255 to 4000 megabytes

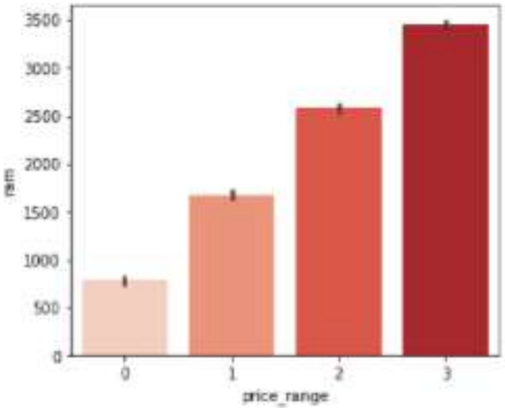


Relationship between the Price Range and Pixel Height/ Width /Ram

- Here we see that Ram of phone and price are highly corelated increase in ram increase in price
- From the above bar plot, we can see that the average pixel height and width are highest for the price range 3(very high cost).
- Low-cost phones have smaller average pixel width and pixel height.
- We can observe from this Bar plot that pixel height and pixel width are roughly equal in relevance when it comes to model development for prediction.



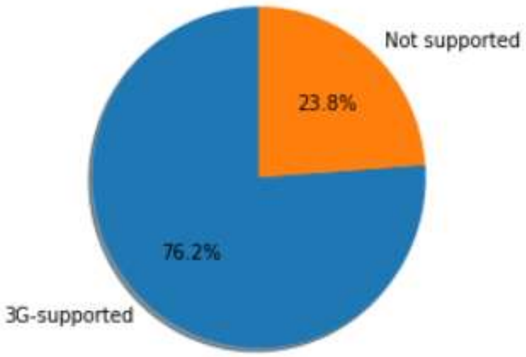


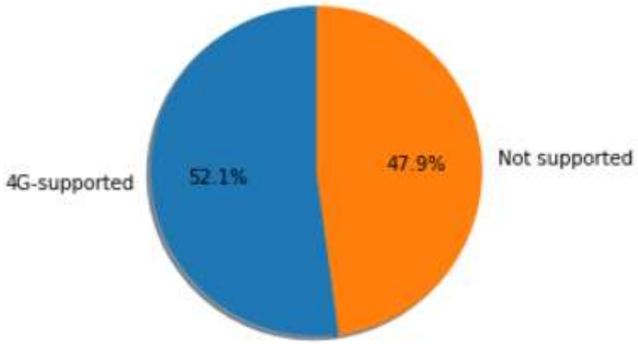


EDA(contd...)

3G-4G supported and Non-supported.

From the above fig, we can see that 3G-supported is 76.2% and 4G-supported is 52.1%.

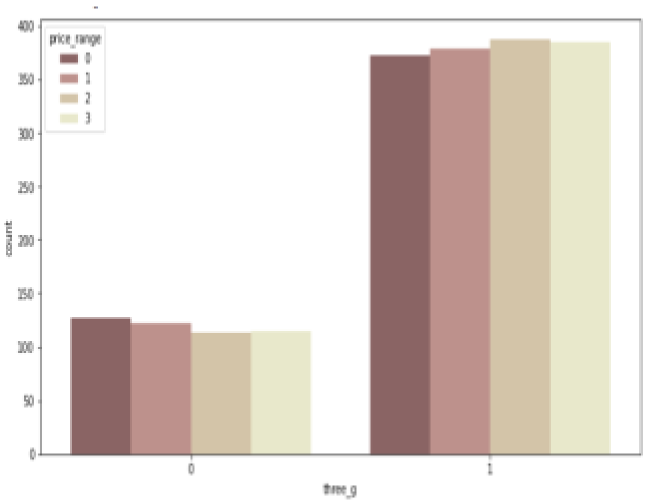


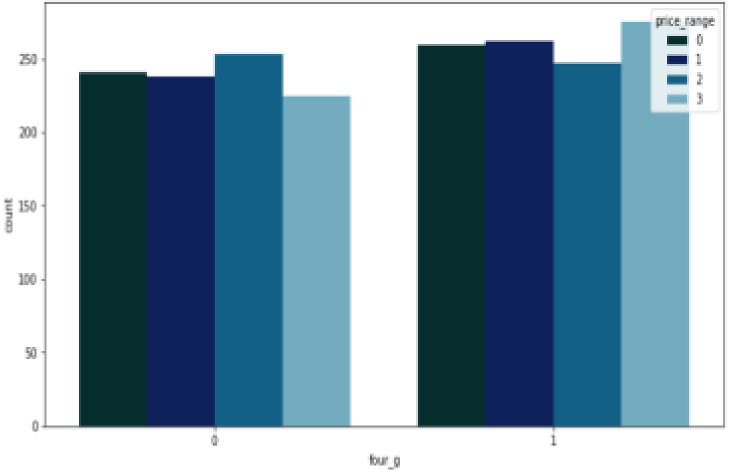


EDA(contd...)

Relation between Price Range & 3G/4G.

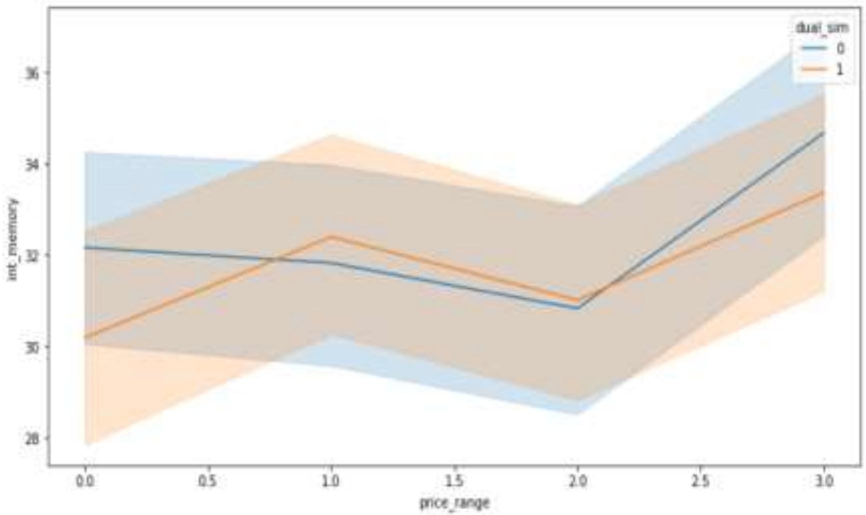
- Here , we see the price range also affecting the 3G and 4G .





Multivariate analysis - int_memory, mobile_wt

- There is drastic increase in internal memory for vey high prices.
- Also there is drastic Decrease in mobile weight for very high price



MULTIVATIATE ANALYSIS

- From the above correlation graph
- three_g and four_g are moderately correlated.
- px_width and px_height are moderately correlated. We will try to change them into a single variable.
- ram is highly correlated with our price range. May be one the most important factor in determining the price.

Preparing dataset for modeling

Task : multiclass classification

Train set : (1340 , 17)

Test set : (660 , 17)

Response : 0-1-2-3

battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	n_cores	pc	px_height
842	0	2.2	0	1	0	7	0.6	188	2	2	20
1021	1	0.5	1	0	1	53	0.7	136	3	6	905
563	1	0.5	1	2	1	41	0.9	145	5	6	1263
615	1	2.5	0	0	0	10	0.8	131	6	9	1216
1821	1	1.2	0	13	1	44	0.6	141	2	14	1208
1859	0	0.5	1	3	0	22	0.7	164	1	7	1004
1821	0	1.7	0	4	1	10	0.8	139	8	10	381
1954	0	0.5	1	0	0	24	0.8	187	4	0	512
1445	1	0.5	0	0	0	53	0.7	174	7	14	386
509	1	0.6	1	2	1	9	0.1	93	5	15	1137
769	1	2.9	1	0	0	9	0.1	182	5	1	248
1520	1	2.2	0	5	1	33	0.5	177	8	18	151
1815	0	2.8	0	2	0	33	0.6	159	4	17	607

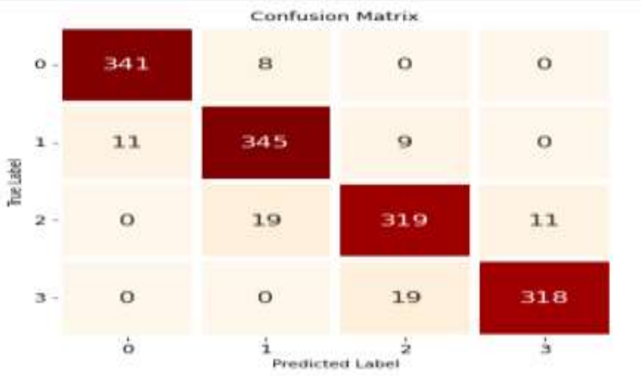
MODEL BUILDING

- KNEIGHBOUR CLASSIFIER
- RANDOM FOREST CLASSIFIER
- GRADIENT BOOSTING CLASSIFIER
- LOGISTIC REGRESSION
- XGB CLASSIFIERD
- DECISION TREE CLASSIFIER
- SUPPORT VECTOR MACHINE

Implementing KNeighbours Classifier contd.

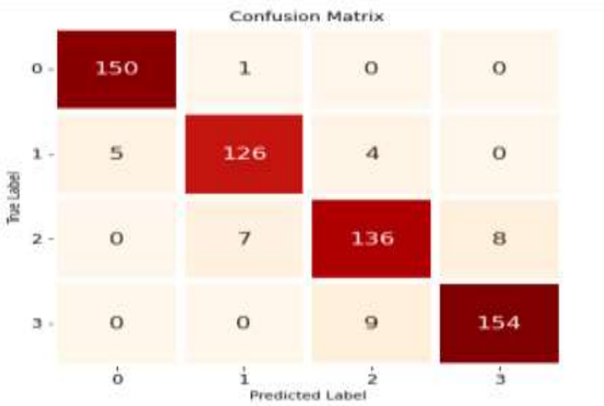
Train metrics

	precision	recall	f1-score	support
0	0.97	0.98	0.97	349
1	0.93	0.95	0.94	365
2	0.92	0.91	0.92	349
3	0.97	0.94	0.95	337
accuracy			0.94	1400
macro avg	0.95	0.94	0.95	1400
weighted avg	0.95	0.94	0.95	1400



Test metrics

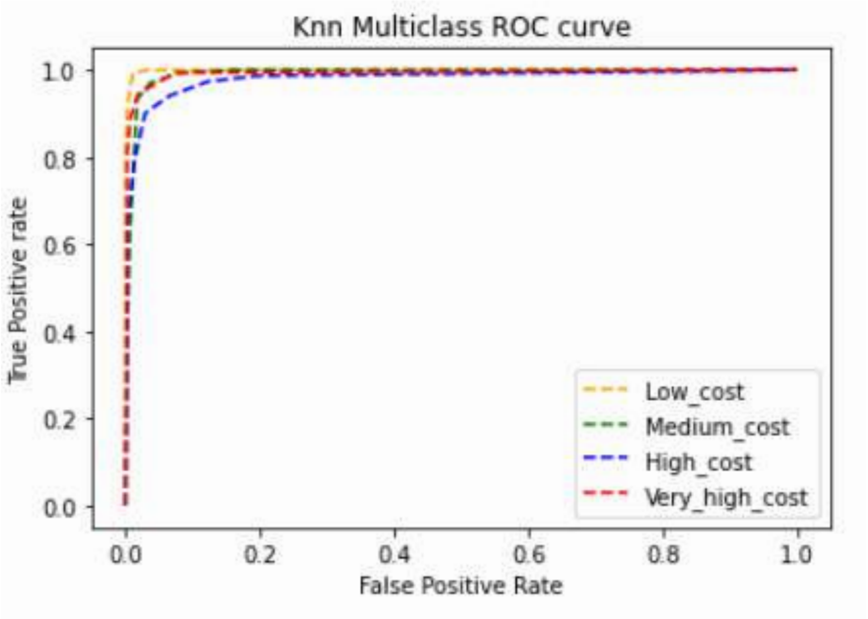
	precision	recall	f1-score	support
0	0.97	0.99	0.98	151
1	0.94	0.93	0.94	135
2	0.91	0.90	0.91	151
3	0.95	0.94	0.95	163
accuracy			0.94	600
macro avg	0.94	0.94	0.94	600
weighted avg	0.94	0.94	0.94	600



Implementing KNeighbours Classifier

TPR = TP/(TP+FN)

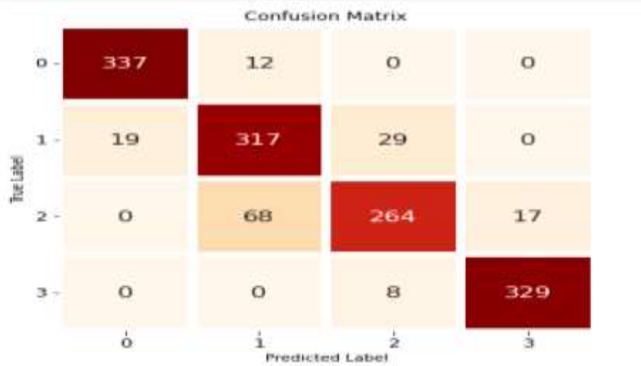
FPR = FP/(FP+TN)



Implementing Random Forest Classifier

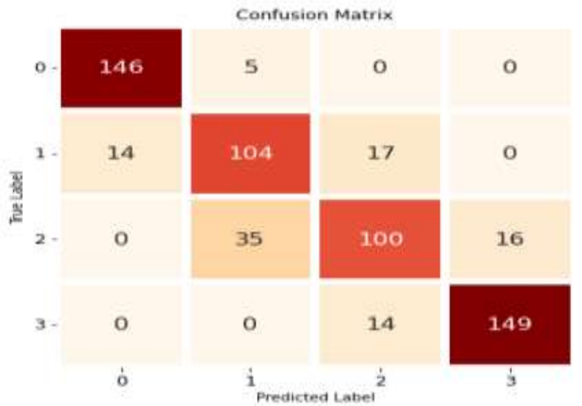
Train metrics

	precision	recall	f1-score	support
0	0.95	0.97	0.96	349
1	0.80	0.87	0.83	365
2	0.88	0.76	0.81	349
3	0.95	0.98	0.96	337
accuracy			0.89	1400
macro avg	0.89	0.89	0.89	1400
weighted avg	0.89	0.89	0.89	1400



Test metrics

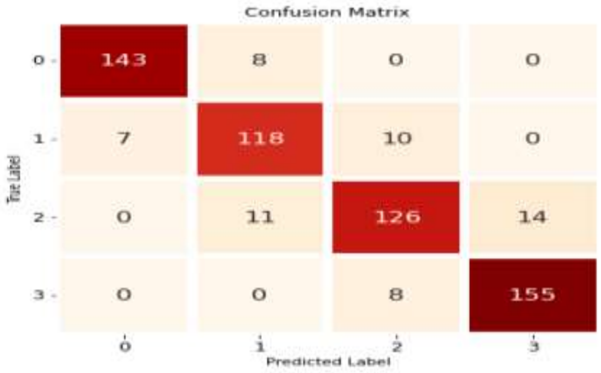
	precision	recall	f1-score	support
0	0.91	0.97	0.94	151
1	0.72	0.77	0.75	135
2	0.76	0.66	0.71	151
3	0.90	0.91	0.91	163
accuracy			0.83	600
macro avg	0.83	0.83	0.83	600
weighted avg	0.83	0.83	0.83	600



Implementing GradientBoostingClassifier

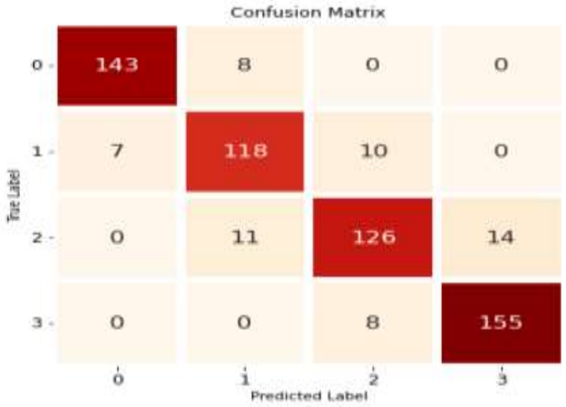
Train metrics

	precision	recall	f1-score	support
0	0.95	0.95	0.95	151
1	0.86	0.87	0.87	135
2	0.88	0.83	0.85	151
3	0.92	0.95	0.93	163
accuracy			0.90	600
macro avg	0.90	0.90	0.90	600
weighted avg	0.90	0.90	0.90	600



Test metrics

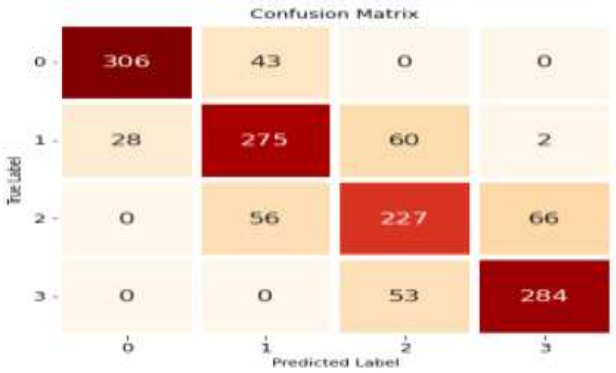
	precision	recall	f1-score	support
0	0.95	0.95	0.95	151
1	0.86	0.87	0.87	135
2	0.88	0.83	0.85	151
3	0.92	0.95	0.93	163
accuracy			0.90	600
macro avg	0.90	0.90	0.90	600
weighted avg	0.90	0.90	0.90	600



Implementing Logistic regression

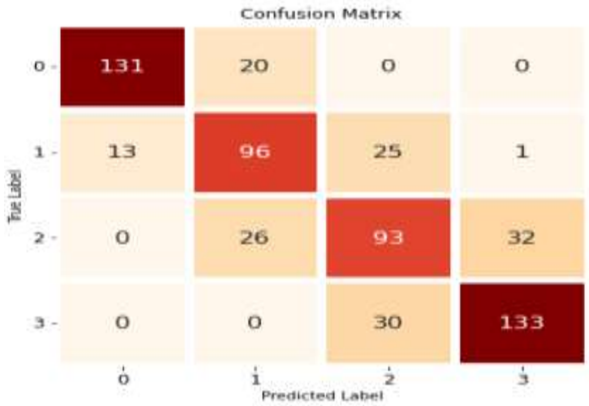
Train metrics

	precision	recall	f1-score	support
0	0.92	0.88	0.90	349
1	0.74	0.75	0.74	365
2	0.67	0.65	0.66	349
3	0.81	0.84	0.82	337
accuracy			0.78	1400
macro avg	0.78	0.78	0.78	1400
weighted avg	0.78	0.78	0.78	1400



Test metrics

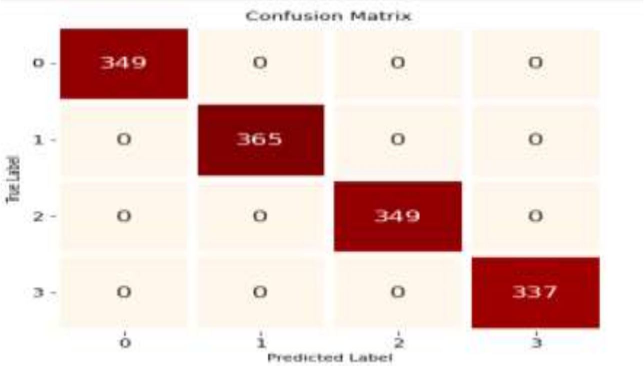
	precision	recall	f1-score	support
0	0.91	0.87	0.89	151
1	0.68	0.71	0.69	135
2	0.63	0.62	0.62	151
3	0.80	0.82	0.81	163
accuracy			0.76	600
macro avg	0.75	0.75	0.75	600
weighted avg	0.76	0.76	0.76	600



Implementing XGBClassifier

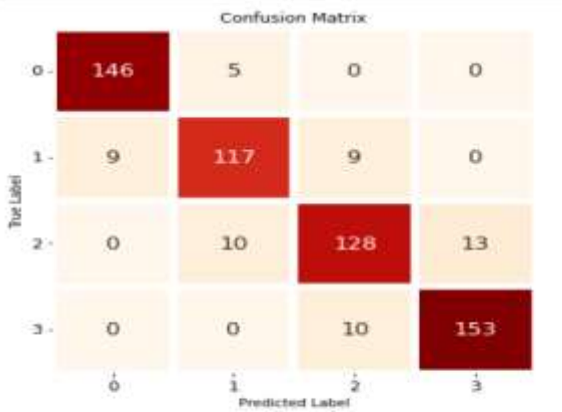
Train metrics

	precision	recall	f1-score	support
0	1.00	1.00	1.00	349
1	1.00	1.00	1.00	365
2	1.00	1.00	1.00	349
3	1.00	1.00	1.00	337
accuracy			1.00	1400
macro avg	1.00	1.00	1.00	1400
weighted avg	1.00	1.00	1.00	1400



Test metrics

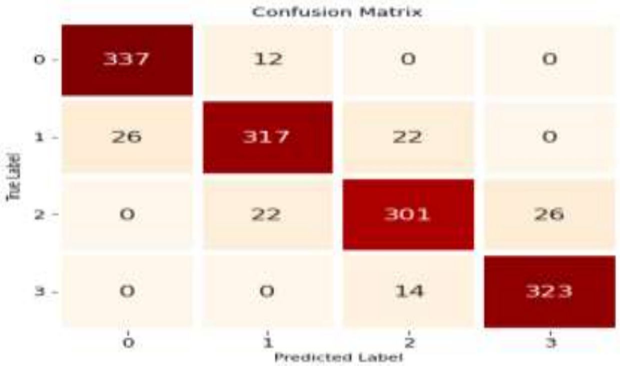
	precision	recall	f1-score	support
0	0.94	0.97	0.95	151
1	0.89	0.87	0.88	135
2	0.87	0.85	0.86	151
3	0.92	0.94	0.93	163
accuracy			0.91	600
macro avg	0.91	0.90	0.90	600
weighted avg	0.91	0.91	0.91	600



Implementing Decision Tree Classifier

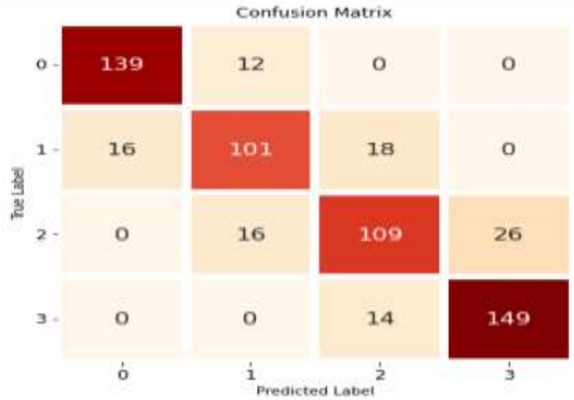
Train metrics

	precision	recall	f1-score	support
0	0.93	0.97	0.95	349
1	0.90	0.87	0.89	365
2	0.89	0.86	0.88	349
3	0.93	0.96	0.94	337
accuracy			0.91	1400
macro avg	0.91	0.91	0.91	1400
weighted avg	0.91	0.91	0.91	1400



Test metrics

	precision	recall	f1-score	support
0	0.90	0.92	0.91	151
1	0.78	0.75	0.77	135
2	0.77	0.72	0.75	151
3	0.85	0.91	0.88	163
accuracy			0.83	600
macro avg	0.83	0.83	0.83	600
weighted avg	0.83	0.83	0.83	600



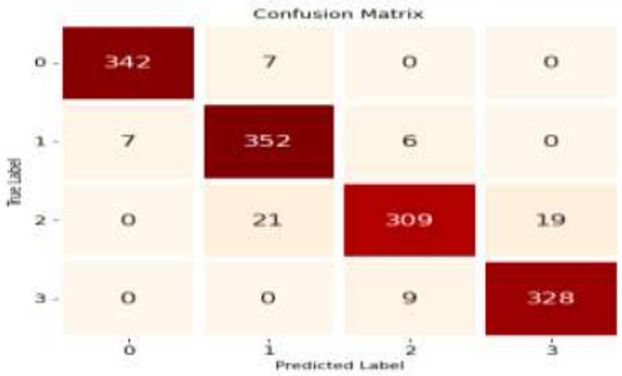
Implementing Support Vector Machine

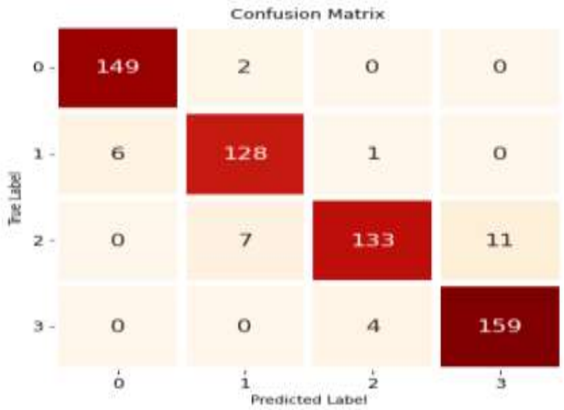
Train metrics

	precision	recall	f1-score	support
0	0.98	0.98	0.98	349
1	0.93	0.96	0.94	365
2	0.95	0.89	0.92	349
3	0.95	0.97	0.96	337
accuracy			0.95	1400
macro avg	0.95	0.95	0.95	1400
weighted avg	0.95	0.95	0.95	1400

Test metrics

	precision	recall	f1-score	support
0	0.96	0.99	0.97	151
1	0.93	0.95	0.94	135
2	0.96	0.88	0.92	151
3	0.94	0.98	0.95	163
accuracy			0.95	600
macro avg	0.95	0.95	0.95	600
weighted avg	0.95	0.95	0.95	600





Best Hyperparameters

So we had chosen Kneighbors classifier for the prediction and the best hyperparameters obtained are as below

Best hyperparameters :

Train : (algorithm='auto', leaf_size=30, metric=‘Euclidean', metric_params=None, n_jobs=None, n_neighbors=11, p=2, weights='distance’)

Test : (algorithm='auto', leaf_size=30, metric='euclidean', metric_params=None, n_jobs=None, n_neighbors=17, p=2, weights='distance')

CONCLUSION

- Ram , Battery_power features were found to be the most relevant features for predicting price range of mobiles and dropping negative correlation features which are clock speed , mobile_wt , touch_screen
- Knn gives acc score of 95% and Xg boost 91%.
- Xgboost and KNN both are given best roc_auc_accuracy score of 99%.
- In case of Xgboost hyper parameter(using grid_search cv) gives very good result.
- Logistic regression is giving the less results among all the algorithms
- So we conclude that kneighbors classifier and Xgboost is giving the best results for these dataset
- So we can say that in the price range prediction as the ram and battery_power increases the price range will increase for sure .

THANK YOU