

# Capstone Project 3

## Mobile Price Range Prediction

Team Project By: **AlmaNorthern**

Pradeep Kumar Yadav  
Y Ishwar Rao

Piyush M. Sonavane  
Ganesh P. Patil

Data Science Trainee  
**AlmaBetter**

# Content :

1. Defining problem statement
2. Data Summary
3. EDA and feature engineering
4. Feature Selection
5. Preparing dataset for modeling
6. Applying Model
7. Model Validation and Selection
8. 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 :**
- **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:

- **Mobile\_wt** - Weight of mobile phone
- **N\_cores** - Number of course of processor
- **Pc** - primary camera mega pixels
- **Px\_height** - Pixel Resolution Height
- **Px\_width** - pixel resolution width
- **Ram** - random access memory in megabytes
- **Sc\_h** - Screen height of mobile in cm
- **Sc\_w** - Screen width of mobile in cm
- **Talk\_time** - Longest time data single battery charge will last when you are

## Data Summary:

**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 ) ,

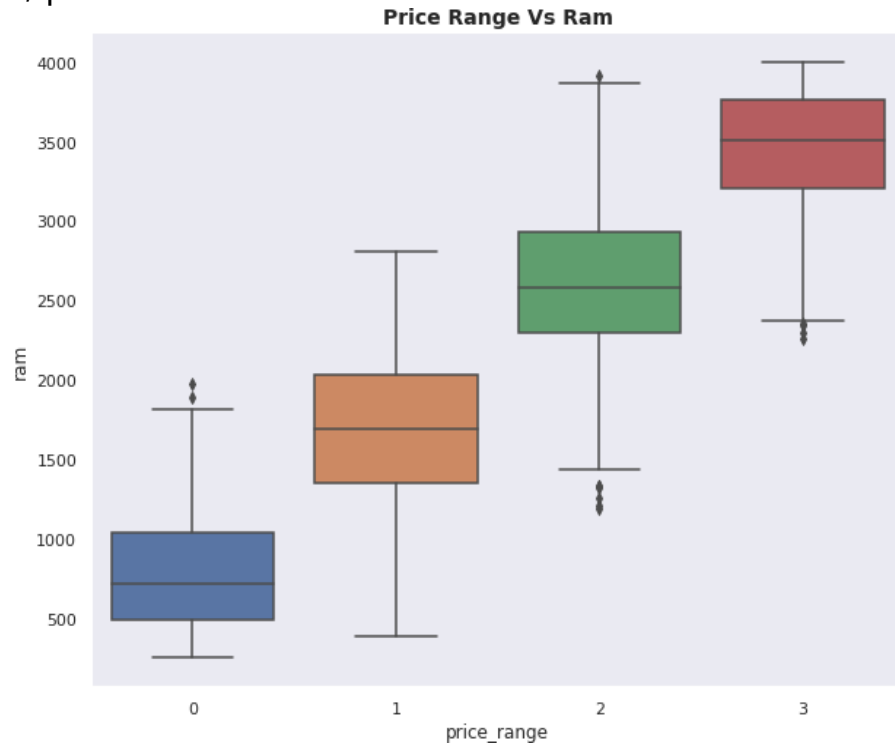
1 ( medium cost ) ,

2 ( high cost ) ,

And 3 ( very high cost ) .

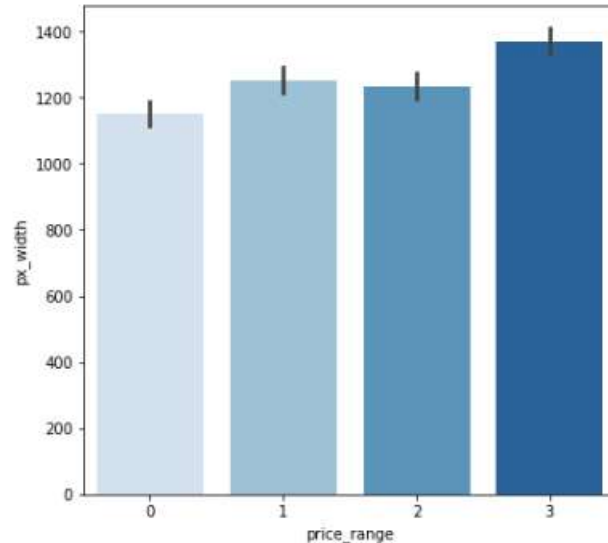
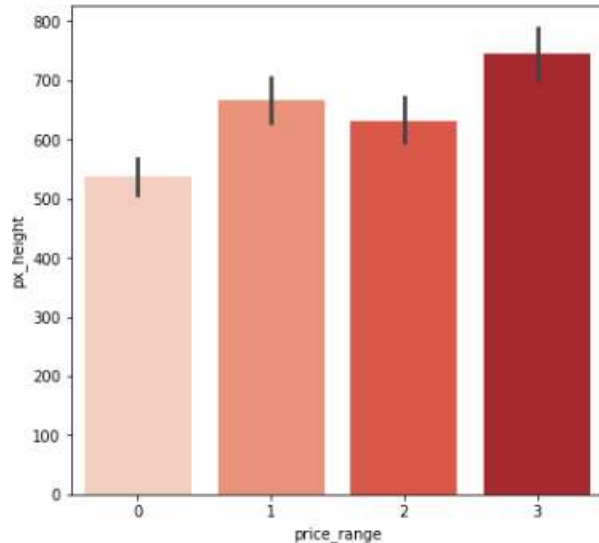
# EDA and Feature engineering Relation Between Price Range & Ram :

- 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 :

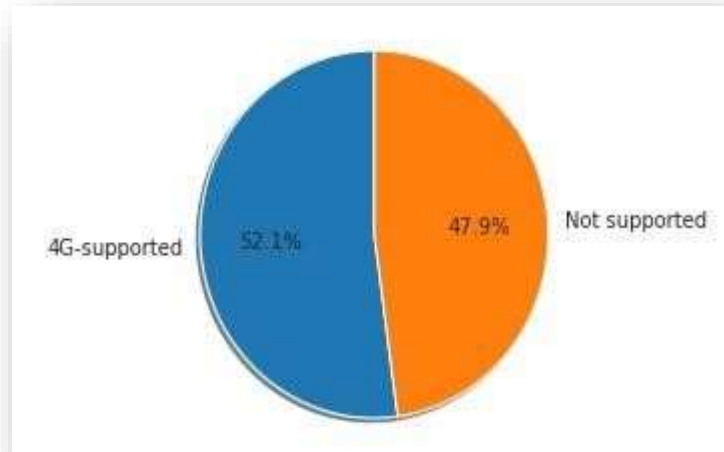
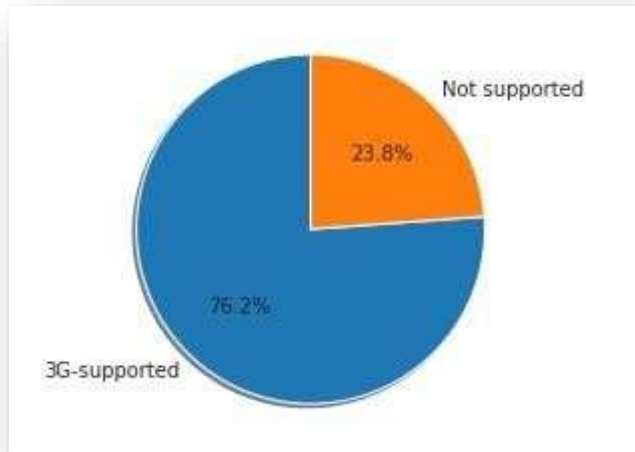
- From the above bar plot, we can see that the average pixel height and width are highest for the price range 3(very high cost).
- Low-cost phones have smaller average pixel width and pixel height.
- We can observe from this Bar plot that pixel height and pixel width are roughly equal in



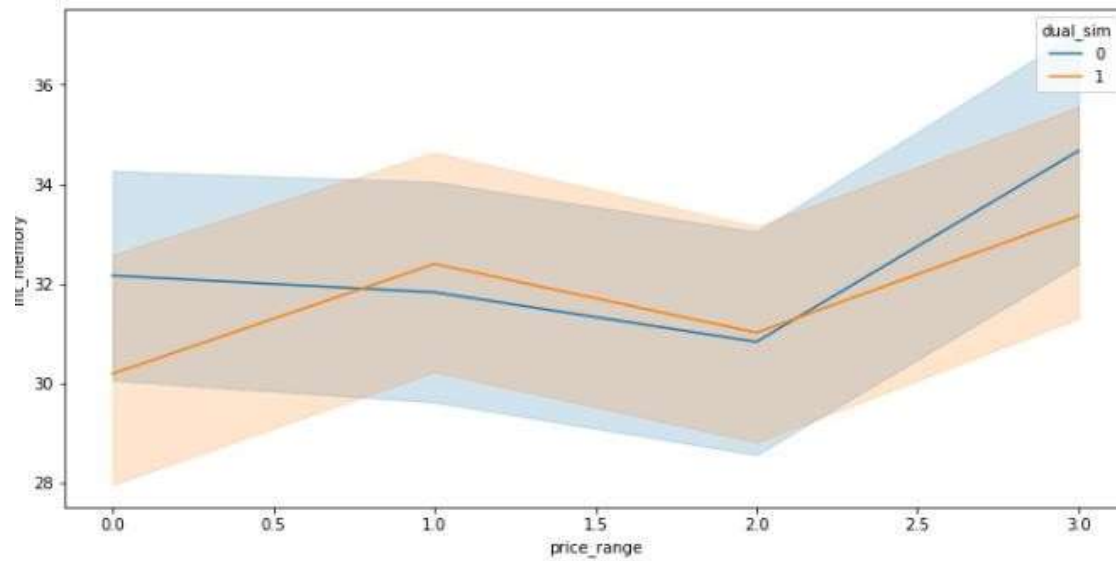


# Exploratory Data Analysis :

- 3G-4G supported and Non-supported



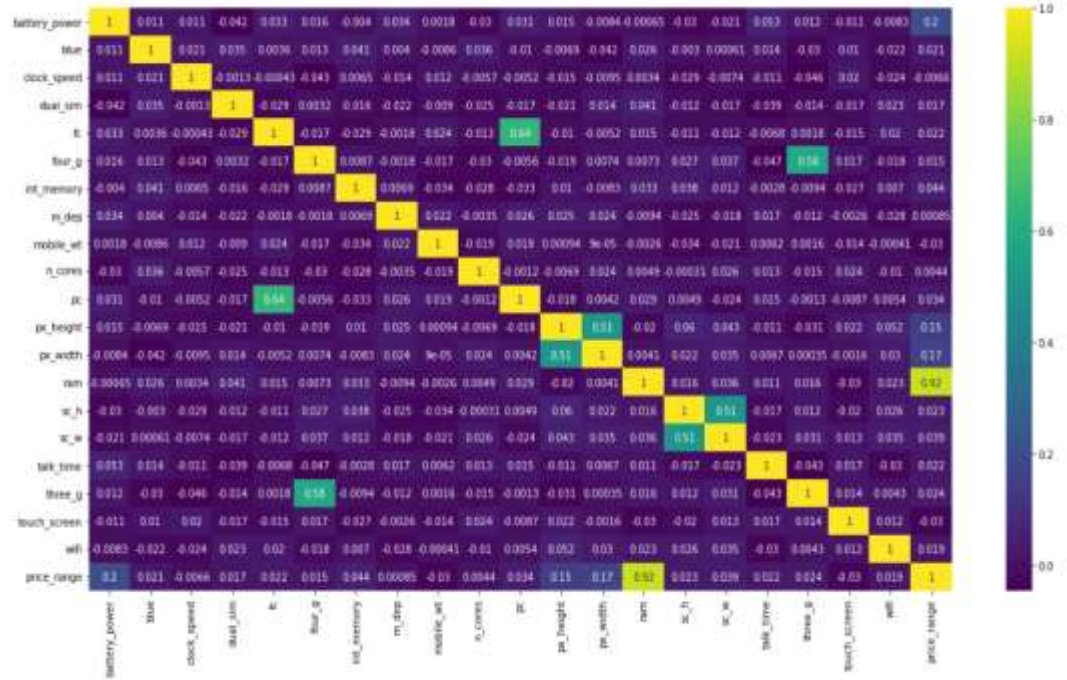
# Multivariate analysis – int\_memory , mobile\_wt :



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

# Multivariate analysis :

- Pc is correlated with Fc.
- px\_height and px\_width are moderately correlated.
- Sc\_h and sc\_w are moderately correlated.
- Ram is highly correlated with price\_range.



# Preparing dataset for modeling:

Task : multiclass  
classification

Train set : (1340 , 20)

Test set : (660 , 20)

Response : 0-1-2-3

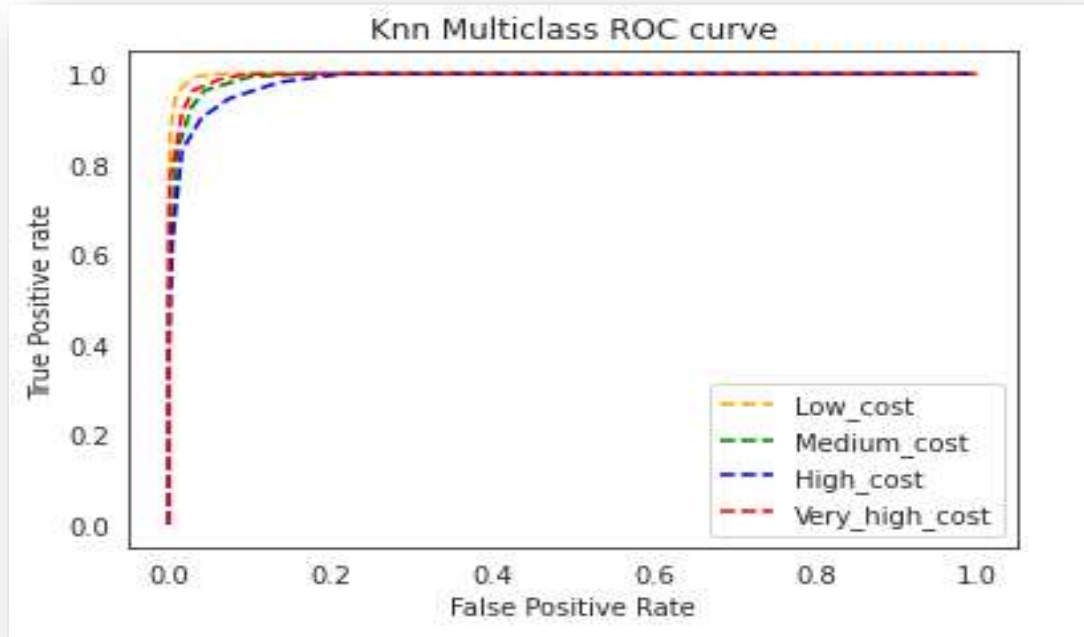
battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	n_cores	pc	px_height
842	0	2.2	0	1	0	7	0.6	188	2	2	20
1021	1	0.5	1	0	1	53	0.7	136	3	6	905
563	1	0.5	1	2	1	41	0.9	145	5	6	1263
615	1	2.5	0	0	0	10	0.8	131	6	9	1216
1821	1	1.2	0	13	1	44	0.6	141	2	14	1208
1859	0	0.5	1	3	0	22	0.7	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

# Applying Model

## Implementing KNeighbours Classifier

TPR(True  
Positive rate)  
=  $TP/(TP+FN)$

FPR(False  
Positiverate)  
=  $FP/(FP+TN)$



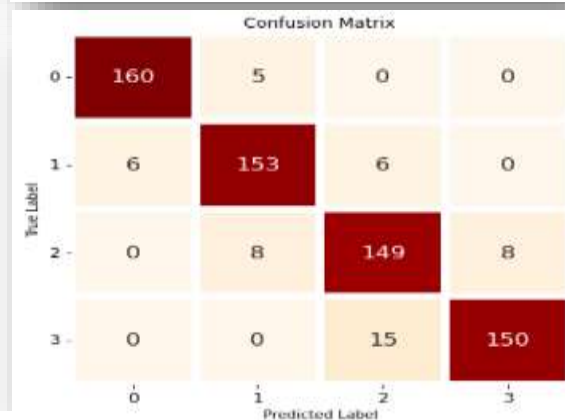
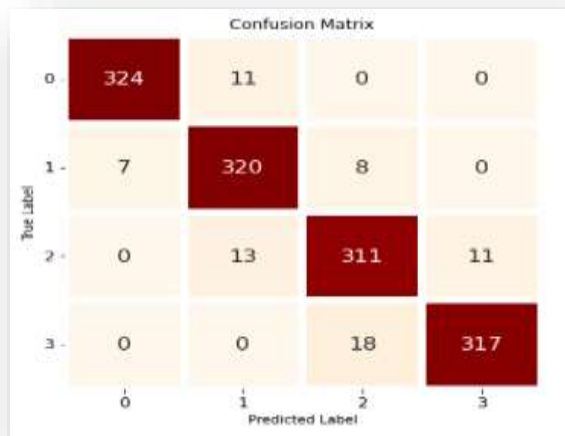
# Implementing KNeighbours Classifier :

## Train metrics

	precision	recall	f1-score	support
0	0.98	0.96	0.97	228
1	0.93	0.96	0.94	212
2	0.93	0.93	0.93	229
3	0.96	0.95	0.96	228
accuracy			0.95	897
macro avg	0.95	0.95	0.95	897
weighted avg	0.95	0.95	0.95	897

## Test metrics

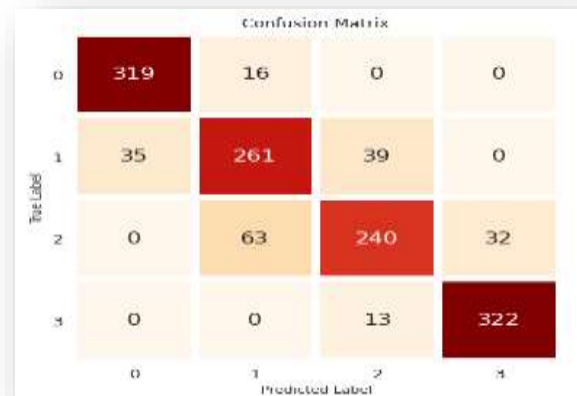
	precision	recall	f1-score	support
0	0.96	0.96	0.96	165
1	0.92	0.93	0.92	165
2	0.88	0.90	0.89	165
3	0.95	0.92	0.93	165
accuracy			0.93	660
macro avg	0.93	0.93	0.93	660
weighted avg	0.93	0.93	0.93	660



# Implementing Random Forest Classifier :

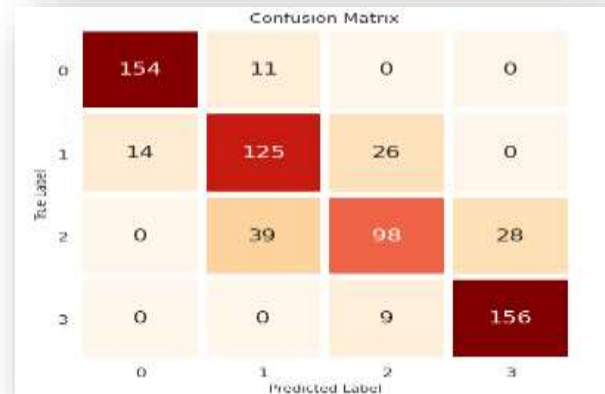
## Train metrics

	precision	recall	f1-score	support
0	0.90	0.95	0.93	335
1	0.77	0.78	0.77	335
2	0.82	0.72	0.77	335
3	0.91	0.96	0.93	335
accuracy			0.85	1340
macro avg	0.85	0.85	0.85	1340
weighted avg	0.85	0.85	0.85	1340



## Test metrics

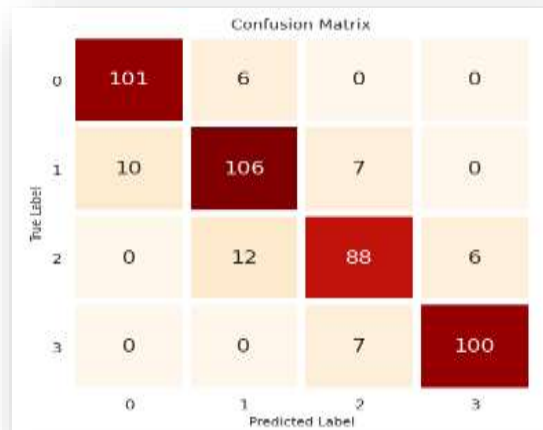
	precision	recall	f1-score	support
0	0.92	0.93	0.92	165
1	0.71	0.76	0.74	165
2	0.74	0.59	0.66	165
3	0.85	0.95	0.89	165
accuracy			0.81	660
macro avg	0.80	0.81	0.80	660
weighted avg	0.80	0.81	0.80	660



# Implementing Gradient Boosting Classifier :

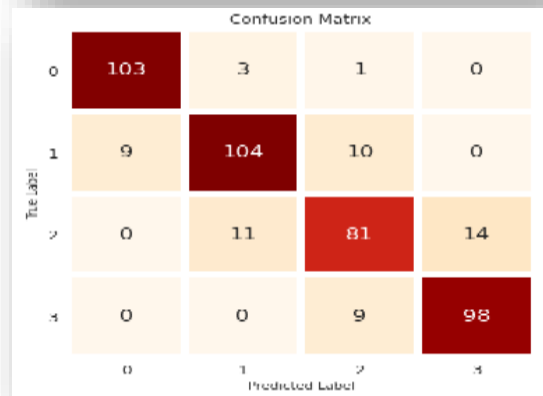
## Train metrics

Classification Report					
	precision	recall	f1-score	support	
0	0.91	0.94	0.93	107	
1	0.85	0.86	0.86	123	
2	0.86	0.83	0.85	106	
3	0.94	0.93	0.94	107	
accuracy			0.89	443	
macro avg	0.89	0.89	0.89	443	
weighted avg	0.89	0.89	0.89	443	



## Test metrics

Classification Report					
	precision	recall	f1-score	support	
0	0.92	0.96	0.94	107	
1	0.88	0.85	0.86	123	
2	0.80	0.76	0.78	106	
3	0.88	0.92	0.89	107	
accuracy			0.87	443	
macro avg	0.87	0.87	0.87	443	
weighted avg	0.87	0.87	0.87	443	





# Implementing XGBClassifier :

## Train metrics

Classification Report					
	precision	recall	f1-score	support	
0	0.92	0.97	0.95	107	
1	0.92	0.88	0.90	123	
2	0.87	0.85	0.86	106	
3	0.92	0.93	0.93	107	
accuracy			0.91	443	
macro avg	0.91	0.91	0.91	443	
weighted avg	0.91	0.91	0.91	443	

Confusion Matrix				
True Label	0	1	2	3
0	104	3	0	0
1	9	108	6	0
2	0	7	90	9
3	0	0	7	100
Predicted Label				

## Test metrics

Classification Report					
	precision	recall	f1-score	support	
0	0.91	0.94	0.93	107	
1	0.88	0.86	0.87	123	
2	0.84	0.81	0.83	106	
3	0.90	0.92	0.91	107	
accuracy			0.88	443	
macro avg	0.88	0.88	0.88	443	
weighted avg	0.88	0.88	0.88	443	

Confusion Matrix				
True Label	0	1	2	3
0	101	6	0	0
1	10	106	7	0
2	0	9	86	11
3	0	0	9	98
Predicted Label				

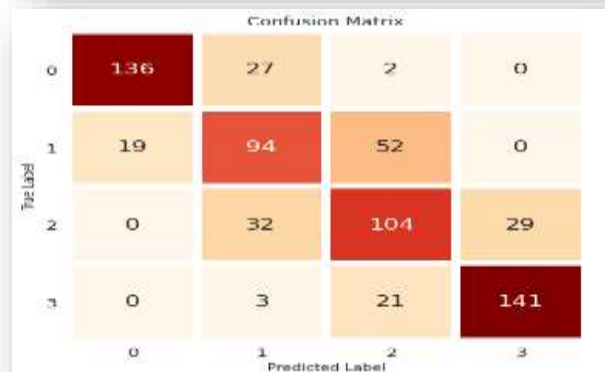
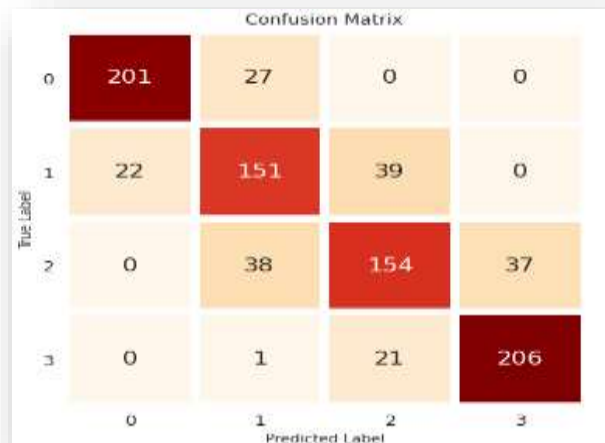
# Implementing Logistic regression :

## Train metrics

	precision	recall	f1-score	support
0	0.90	0.88	0.89	228
1	0.70	0.71	0.70	212
2	0.72	0.67	0.70	229
3	0.85	0.90	0.87	228
accuracy			0.79	897
macro avg	0.79	0.79	0.79	897
weighted avg	0.79	0.79	0.79	897

## Test metrics

	precision	recall	f1-score	support
0	0.88	0.82	0.85	165
1	0.60	0.57	0.59	165
2	0.58	0.63	0.60	165
3	0.83	0.85	0.84	165
accuracy			0.72	660
macro avg	0.72	0.72	0.72	660
weighted avg	0.72	0.72	0.72	660



# Model Validation & Selection contd...

## Observations:

1. As seen in the above slides Random forest classifier is not giving great results , Gradient Boosting Classifier is bit better than Random forest in recall and precision
2. XGboost classifier is giving the better results than GB but the recall of random forest classifier is somewhat similar
3. KNeighbors is giving the best results among all of the algorithms
4. Logistic regression is giving low results among all of them

# Model Validation & Selection contd...

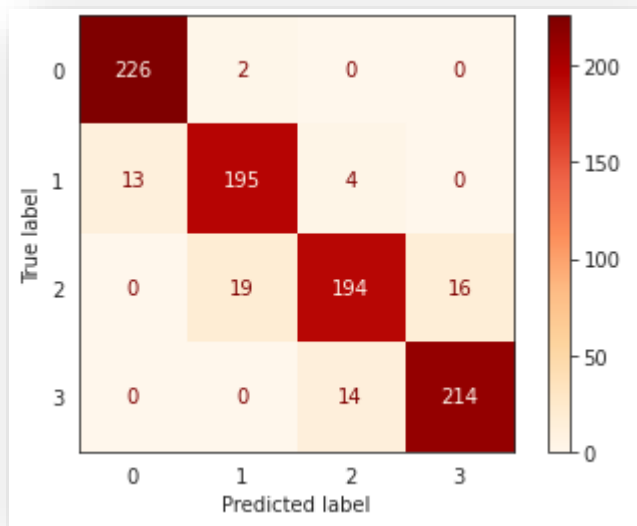
So we had chosen Kneighbors classifier for the prediction and the best hyperparameters obtained are as below

## 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')

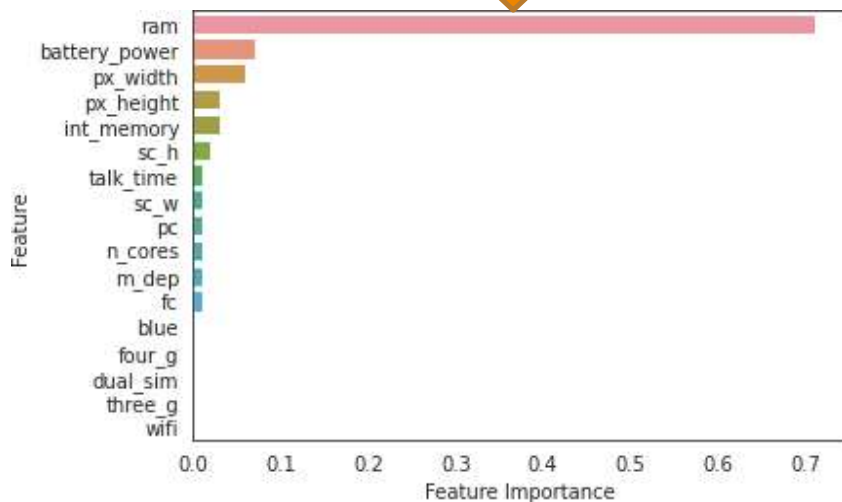
# Model Validation & Selection (Hyperparameter tuned) :



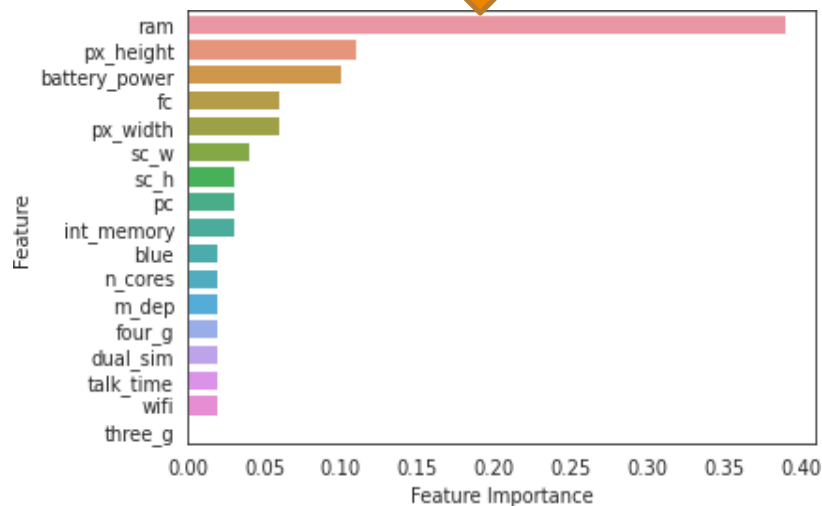
	precision	recall	f1-score	support
0	0.95	0.99	0.97	228
1	0.90	0.92	0.91	212
2	0.92	0.85	0.88	229
3	0.93	0.94	0.93	228
accuracy			0.92	897
macro avg	0.92	0.92	0.92	897
weighted avg	0.92	0.92	0.92	897

# Feature Importance:

## Random Forest Classifier



## XGBoost Classifier



## Conclusion :

- ❖ Ram , Battery\_power features were found to be the most relevant feature for predicting price range of mobiles and dropping negative correlation features which are clock speed , mobile\_wt , touch\_screen .
- ❖ Kneighbors and Xgboost are given best accuracy score 95 % test , 93 % train and 91 % train , 88 % test respectively and roc\_auc score for kneighbors is 99 % .
- ❖ Tuning the hyperparameters by GridSearch CV on kneighbors but not getting much difference in results but the best parameters n\_neighbors for train and test are 11 and 17

## Conclusion :

- ❖ So we conclude that kneighbors classifier 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