Untitled4

April 14, 2022

```
[1]: import warnings
     warnings.filterwarnings('ignore')
     import numpy as np
     import sklearn
     import pandas as pd
     import seaborn as sns
     import matplotlib.pyplot as plt
[2]: from sklearn.neighbors import KNeighborsClassifier
     from sklearn.linear_model import LogisticRegression,SGDClassifier
     from sklearn.svm import SVC, LinearSVC, NuSVC
     from sklearn.tree import DecisionTreeClassifier
     from sklearn.naive_bayes import GaussianNB,MultinomialNB
     from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier, u
      \hookrightarrow Gradient Boosting Classifier
     from sklearn.metrics import classification_report
     from sklearn.pipeline import Pipeline,make_pipeline
     from sklearn.metrics import accuracy_score, log_loss
     from sklearn.preprocessing import StandardScaler, MinMaxScaler
     from sklearn.model_selection import train_test_split, GridSearchCV
     from statsmodels.stats.outliers_influence import variance_inflation_factor
     from sklearn.decomposition import PCA
     from sklearn.feature_selection import (RFE, SelectKBest,
                                            SelectPercentile,RFECV)
[3]: df = pd.read_csv('data/Train_Data.csv')
     df.head()
                             clock_speed dual_sim fc four_g
[3]:
                       blue
                                                                            m_dep \
        battery_power
                                                                 int_memory
     0
                  842
                                     2.2
                                                              0
                                                                          7
                                                                               0.6
                          0
                                                 0
                                                     1
                                     0.5
                                                                               0.7
     1
                 1021
                          1
                                                 1
                                                    0
                                                              1
                                                                         53
     2
                  563
                                     0.5
                                                     2
                                                                               0.9
                                                 1
                                                              1
                                                                         41
     3
                  615
                                     2.5
                                                    0
                                                                         10
                                                                               0.8
                          1
                                     1.2
     4
                 1821
                                                 0 13
                                                                         44
                                                                               0.6
        mobile_wt n_cores pc px_height px_width ram sc_h sc_w talk_time \
```

```
2
                                                  756 2549
     0
              188
                          2
                                        20
                                                                 9
                                                                       7
                                                                                 19
     1
              136
                          3
                             6
                                       905
                                                 1988 2631
                                                                       3
                                                                                  7
                                                                17
                                                                       2
     2
              145
                          5
                              6
                                      1263
                                                 1716 2603
                                                                11
                                                                                  9
     3
                                                                16
                                                                       8
              131
                          6
                              9
                                      1216
                                                 1786
                                                       2769
                                                                                 11
     4
              141
                          2
                             14
                                      1208
                                                 1212 1411
                                                                 8
                                                                       2
                                                                                 15
        three_g touch_screen wifi
     0
              0
                             0
                                   1
              1
                             1
                                   0
     1
     2
              1
                             1
                                   0
     3
              1
                             0
                                   0
              1
                                   0
[4]: df2 = pd.read_csv('data/Traindata_classlabels.csv')
     df2.head()
[4]:
        price_range
     1
                  2
     2
                  2
                  2
     3
     4
                  1
[5]: data = {}
     data_GS = {}
[6]: def Classification(X_train, X_test, y_train, y_test,data,name:str,Gridsearch = ___
      →False):
         try:
             y_train = y_train.values.ravel()
             y_test = y_test.values.ravel()
         except:
             pass
         classifiers = [
         KNeighborsClassifier(),
         LogisticRegression()
         if Gridsearch ==True:
             clf_parameters = [
                 {
                      "clf__n_neighbors": np.arange(2,25,1),
                      "clf__metric": __
      → ["cityblock", "cosine", "euclidean", "11", "12", "manhattan", "nan_euclidean",],
                      "clf_weights": ["uniform", "distance"],
                      "clf_algorithm": ["auto", "ball_tree", "kd_tree", "brute"],
              },
```

```
"clf__C": np.logspace(-2, -1.5, 30),
               "clf_penalty": ["11", "12", "elasticnet", "none"],
               "clf_solver": ["newton-cg", "lbfgs", "liblinear", "sag", _{\sqcup}
}
        1
   else:
       clf_parameters = [{},{}]
  data[name] = {'Model':[],'Accuracy' :[],'f1_micro' :[],'f1_macro' :[]}
  dataint = {'Model':[],'Accuracy' :[],'f1_micro' :[],'f1_macro' :[]}
  i=1
  for classifier,clf_params in zip(classifiers,clf_parameters):
      pipe = Pipeline(steps=[('clf', classifier)])
       grid = GridSearchCV(pipe,clf_params,scoring='f1_macro',cv=10,n_jobs=-1)
      try:
           grid.fit(X_train, y_train)
           pred = grid.predict(X_test)
           print("_"*32)
           print(f'{i}.',classifier)
           print(" "*32)
           print(grid.best_params_)
           print(classification_report(y_test, pred))
           i+=1
           i1 = classifier.__class__.__name__
           i2 = sklearn.metrics.accuracy_score(y_test,pred)
           i3 = sklearn.metrics.f1_score(y_test,pred,average='micro')
           i4 = sklearn.metrics.f1_score(y_test,pred,average='macro')
           dataint['Model'].append(i1)
           dataint['Accuracy'].append(i2)
           dataint['f1 micro'].append(i3)
           dataint['f1_macro'].append(i4)
           print("-"*80)
           print("-"*80)
       except Exception as e: print(e)
   classifiers = [
      DecisionTreeClassifier(),
      SVC(),
      NuSVC(),
       RandomForestClassifier(),
       AdaBoostClassifier(),
       GradientBoostingClassifier(),
       SGDClassifier()
  base estimators = classifiers
```

```
if Gridsearch==True:
       clf_parameters = [
               'clf__criterion' : ["gini", "entropy"],
           'clf__max_features':['sqrt', 'log2',None],
                      'max_depth':np.linspace(140,190,10),
                    'clf__ccp_alpha':np.logspace(-3,-2,20),#np.logspace(-2.
\rightarrow 32, -2.3,20),
           "clf__max_leaf_nodes" : [None] + np.arange(30,40,5).tolist(),
           "clf_splitter" : ["best", "random"],
           "clf__min_samples_split":np.arange(2,50,10)
       },
           {
                   'clf C':(0.8,2,30),
                   'clf__kernel':('linear','rbf','poly','sigmoid'),
                   'clf__decision_function_shape' : ['ovo', 'ovr'],
                   'clf__degree':np.arange(3,5,1)
                   },
           {
               'clf_nu': np.logspace(-1.15,-1.23,5), #np.logspace(-2,-1,30)
            'clf_kernel' : ['linear', 'poly', 'rbf', 'sigmoid' ]
               'clf_gamma' : ['scale', 'auto'],
               'clf__degree':np.arange(3,5,1),
               'clf__decision_function_shape' : ['ovo', 'ovr'],
           },
                'clf_n_estimators': [150,200,250],
           'clf__max_features': ['sqrt', 'log2',None],
           'clf_max_depth' : [None], #np.arange(4,15,2).tolist(),
           'clf__criterion' :['gini', 'entropy']
               'clf bootstrap' : [True],
                 'clf_ccp_alpha':np.logspace(-2,1,10)
       #
           },
               'clf__base_estimator':
→ [RandomForestClassifier(), DecisionTreeClassifier(criterion='entropy'), SVC(), LogisticRegress
→ 0.017433288221999882, 'penalty': '12', 'solver': 'newton-cg'})],
               'clf__algorithm' : ['SAMME', 'SAMME.R'],
               'clf_n_estimators': [50,100]
           },
             'clf__loss' :['deviance', 'exponential'],
               'clf__criterion' : ['friedman_mse', 'squared_error'],
               'clf__max_features' : [ 'sqrt', 'log2'],
```

```
'clf_learning_rate': np.logspace(-1,1,5),
               'clf_n_estimators':np.arange(100,1000,200)
          },
              'clf__loss' :['hinge', 'log', 'modified_huber','squared_hinge',_
'clf__penalty' : ['12', '11', 'elasticnet'],
              'clf_alpha': np.logspace(-4,-2,30),
              'clf_learning_rate' :[__
}
          1
  else:
      clf_parameters = [{}]*len(classifiers)
  for classifier,clf_params in zip(classifiers,clf_parameters):
      pipe = Pipeline(steps=[('standardscaler', StandardScaler()),('clf', _
→classifier)])
      grid = GridSearchCV(pipe,clf_params,scoring='f1_macro',cv=10,n_jobs=-1)
      try:
          grid.fit(X_train, y_train)
          pred = grid.predict(X_test)
          print(" "*32)
          print(f'{i}.',classifier)
          print(" "*32)
          print(grid.best_params_)
          print(classification_report(y_test, pred))
          i=i+1
          i1 = classifier.__class__.__name__
          i2 = sklearn.metrics.accuracy_score(y_test,pred)
          i3 = sklearn.metrics.f1_score(y_test,pred,average='micro')
          i4 = sklearn.metrics.f1_score(y_test,pred,average='macro')
          dataint['Model'].append(i1)
          dataint['Accuracy'].append(i2)
          dataint['f1_micro'].append(i3)
          dataint['f1_macro'].append(i4)
          print("-"*80)
          print("-"*80)
      except Exception as e: print(e)
   classifiers = \Gamma
  GaussianNB(),
  MultinomialNB()
   if Gridsearch==True:
```

```
clf_parameters = [
           {
           'clf__var_smoothing':np.logspace(-20,-10,10)
          },
           {
               'clf__alpha':[0] + np.logspace(-2,5,5).tolist(),
               'clf__fit_prior':[True,False]
           }
            ٦
   else:
       clf_parameters = [{}]*len(classifiers)
  for classifier,clf_params in zip(classifiers,clf_parameters):
      pipe = Pipeline(steps=[('minmaxscalar',MinMaxScaler()),('clf',__
→classifier)])
       grid = GridSearchCV(pipe,clf_params,scoring='f1_macro',cv=10,n_jobs=-1)
      try:
           grid.fit(X_train, y_train)
           pred = grid.predict(X_test)
           print("_"*32)
           print(f'{i}.',classifier)
           print("_"*32)
           print(grid.best_params_)
           print(classification_report(y_test, pred))
           i=i+1
           i1 = classifier.__class__.__name__
           i2 = sklearn.metrics.accuracy_score(y_test,pred)
           i3 = sklearn.metrics.f1_score(y_test,pred,average='micro')
           i4 = sklearn.metrics.f1_score(y_test,pred,average='macro')
           dataint['Model'].append(i1)
           dataint['Accuracy'].append(i2)
           dataint['f1_micro'].append(i3)
           dataint['f1_macro'].append(i4)
           print("-"*80)
           print("-"*80)
       except Exception as e: print(e)
  data[name] = dataint
     print(dataint)
```

1 With and Without Grid Search

1.1 Raw data

support

KNeighborsClassifier()

{}			
	precision	recall	f1-score

0	0.96	0.98	0.97	103
1	0.88	0.92	0.90	92
2	0.88	0.87	0.88	101
3	0.97	0.91	0.94	104
accuracy			0.92	400
macro avg	0.92	0.92	0.92	400
weighted avg	0.92	0.92	0.92	400

2. LogisticRegression()

{}

C	precision	recall	f1-score	support
0 1 2 3	0.88 0.60 0.49 0.67	0.86 0.60 0.46 0.74	0.87 0.60 0.47 0.70	103 92 101 104
accuracy macro avg weighted avg	0.66 0.66	0.66 0.67	0.67 0.66 0.67	400 400 400

⁻⁻⁻⁻⁻

^{3.} DecisionTreeClassifier()

⁻⁻⁻⁻⁻

{}					
	precision	recall	f1-score	support	
0	0.93	0.90	0.92	103	
1		0.90		92	
2	0.81	0.80	0.81	101	
3	0.90	0.90	0.90	104	
accuracy			0.86	400	
macro avg	0.86	0.86	0.86	400	
weighted avg	0.86	0.86	0.86	400	
4. SVC()					
{}					
	precision	recall	f1-score	support	
0	0.95	0.94	0.95	103	
1	0.85	0.84	0.84	92	
2	0.80	0.89	0.85	101	
3	0.97	0.88	0.92	104	
2.coura.cu					
accuracy			0.89	400	
accuracy macro avg	0.89	0.89	0.89 0.89	400 400	
· ·	0.89 0.89	0.89 0.89	0.89		
macro avg			0.89	400	
macro avg			0.89	400	
macro avg			0.89	400	
macro avg			0.89	400	
macro avg weighted avg 5. NuSVC()			0.89 0.89	400	
macro avg weighted avg 5. NuSVC()	0.89	0.89	0.89 0.89	400 400	
macro avg weighted avg 5. NuSVC() {}	0.89	0.89 recall	0.89 0.89	400 400 support	

0.92

0.88

0.88

0.88

104

400

400

400

3

accuracy

macro avg

weighted avg

0.97

0.88

0.89

0.88

0.88

0.88

6. RandomForestClassifier()

U	precision	recall	f1-score	support
0	0.95 0.83	0.98 0.84	0.97 0.83	103 92
2	0.82	0.80	0.81	101 104
accuracy	0.01	0.02	0.89	400
macro avg	0.89 0.89	0.89 0.89	0.89	400 400

7. AdaBoostClassifier()

{}

	precision	recall	f1-score	support
0	0.90	0.25	0.39	103
1	0.46	0.86	0.60	92
2	0.48	0.84	0.61	101
3	0.96	0.21	0.35	104
accuracy			0.53	400
macro avg	0.70	0.54	0.49	400
weighted avg	0.71	0.53	0.48	400

8. GradientBoostingClassifier()

{}	precision	recall	f1-score	support
0	0.95	0.97	0.96	103
1	0.85	0.88	0.87	92
2	0.87	0.83	0.85	101
3	0.94	0.93	0.94	104
accuracy			0.91	400
macro avg	0.90	0.90	0.90	400
weighted avg	0.90	0.91	0.90	400

9. SGDClassifier() {} precision recall f1-score support 0 0.99 1.00 1.00 103 1 0.52 0.58 0.55 92 2 0.55 0.48 0.51 101 0.96 3 0.99 0.98 104 0.77 400 accuracy 0.76 0.76 0.76 400 macro avg weighted avg 0.76 0.77 0.76 400 10. GaussianNB() -----{} precision recall f1-score support 0 0.95 0.97 0.96 103 0.71 1 0.71 0.71 92 2 0.68 0.67 0.68 101 3 0.91 0.90 0.91 104 accuracy 0.82 400 0.81 0.81 0.81 400 macro avg weighted avg 0.82 0.82 0.82 400

11. MultinomialNB()

precision recall f1-score support 0 0.71 0.79 0.90 103 0.46 0.60 0.52 92 2 0.41 0.44 0.42 101 0.66 0.59 0.62 104

accuracy			0.58	400	
•	0.61	0.58	0.59	400	
weighted avg				400	
WITH GridSean	 cch				
1. KNeighbors	sClassifier()				
{'clf_algori	 ithm': 'auto'	 , 'clfm	etric': 'c	ityblock',	'clf_n_neighbors': 18,
clfweights				-	-
-	precision		f1-score	support	
0	0.97	0.98	0.98	103	
1				92	
2				101	
3				104	
accuracy			0.94	400	
macro avg	0.94	0.94	0.94	400	
weighted avg	0.94	0.94	0.94	400	
2. LogisticRe	egression()				
{'clfC': 0.cg'}	.028072162039	 41177, 'c	:lfpenalt	y': '12',	'clfsolver': 'newton-
	precision	recall	f1-score	support	
0	1.00	0.99	1.00	103	
1	0.98	0.99	0.98	92	
2	0.98	0.98	0.98	101	
3	0.99	0.99	0.99	104	
accuracy			0.99	400	
macro avg	0.99	0.99	0.99	400	
weighted avg		0.99	0.99	400	

^{3.} DecisionTreeClassifier()

```
{'clf__ccp_alpha': 0.004832930238571752, 'clf__criterion': 'entropy',
'clf__max_features': None, 'clf__max_leaf_nodes': None,
'clf__min_samples_split': 2, 'clf__splitter': 'best'}
             precision
                        recall f1-score
                                             support
           0
                  0.90
                            0.92
                                       0.91
                                                  103
           1
                  0.81
                            0.79
                                       0.80
                                                  92
           2
                  0.82
                           0.85
                                      0.83
                                                  101
           3
                  0.94
                           0.90
                                      0.92
                                                  104
                                       0.87
                                                  400
   accuracy
                                       0.87
  macro avg
                  0.87
                            0.87
                                                  400
weighted avg
                  0.87
                            0.87
                                       0.87
                                                  400
4. SVC()
{'clf_C': 30, 'clf_decision_function_shape': 'ovo', 'clf_degree': 3,
'clf__kernel': 'linear'}
             precision recall f1-score
                                             support
           0
                  0.97
                            1.00
                                      0.99
                                                 103
           1
                  0.99
                            0.96
                                       0.97
                                                  92
           2
                  0.98
                            0.98
                                       0.98
                                                  101
           3
                  0.99
                            0.99
                                       0.99
                                                  104
                                       0.98
                                                  400
   accuracy
  macro avg
                  0.98
                            0.98
                                       0.98
                                                  400
weighted avg
                  0.98
                            0.98
                                       0.98
                                                  400
5. NuSVC()
{'clf__decision_function_shape': 'ovo', 'clf__degree': 3, 'clf__gamma': 'scale',
'clf_kernel': 'linear', 'clf_nu': 0.0707945784384138}
             precision
                        recall f1-score
                                             support
           0
                  0.97
                            0.99
                                       0.98
                                                  103
           1
                  0.96
                            0.96
                                       0.96
                                                  92
           2
                  0.98
                            0.94
                                       0.96
                                                  101
                  0.97
                            0.99
                                      0.98
                                                  104
   accuracy
                                       0.97
                                                 400
  macro avg
                  0.97
                            0.97
                                       0.97
                                                  400
```

weighted ave	S	0.97	0.97	0.97	400	
6. RandomFor	restClas	sifier()			
	 tran':	True '	 clf crit	erion!: 'e	ntrony!	'clfmax_depth': None,
'clfmax_fe						ciimax_depth . None,
0				f1-score		
	•				11	
()	0.95	0.96	0.96	103	
1	L	0.87	0.92	0.89	92	
2	2	0.92	0.87	0.89	101	
3	3	0.96	0.94	0.95	104	
	_			0.02	400	
accuracy		0 02	0.92		400 400	
macro ava	•	0.92		0.92	400	
weighted ave	3	0.93	0.93	0.93	400	
7. AdaBoost({'clf_algor'clf_n_esti	 rithm':	 'SAMME.	 R', 'clf ₋	_base_esti	.mator': F	RandomForestClassifier(),
	preci	sion	recall	f1-score	support	
()	0.95	0.08	0.97	103	
				0.86	92	
				0.81	101	
		0.93	0.90		104	
accuracy	τ			0.89	400	
macro avg	5	0.89	0.89	0.89	400	
weighted ave	5	0.89	0.89	0.89	400	
	Boosting erion': 'clfm	Classif 'square	ier() d_error'; ures': ']		n_estin	ce': 0.1, 'clfloss': nators': 300}
()	0.98	0.97	0.98	103	

1 0.84 0.88 0.86 92 2 0.83 0.84 0.84 101 3 0.96 0.92 0.94 104 accuracy 0.91 400 weighted avg 0.91 0.91 0.91 400						
3 0.96 0.92 0.94 104 accuracy 0.91 400 macro avg 0.90 0.90 0.90 400 weighted avg 0.91 0.91 0.91 400	1	0.84	0.88	0.86	92	
accuracy	2	0.83	0.84	0.84	101	
macro avg 0.90 0.90 0.90 400 weighted avg 0.91 0.91 0.91 400	3	0.96	0.92	0.94	104	
macro avg 0.90 0.90 0.90 400 weighted avg 0.91 0.91 0.91 400						
<pre>weighted avg</pre>						
9. SGDClassifier()	_					
{'clfalpha': 0.004520353656360241, 'clflearning_rate': 'optimal', 'clfloss': 'log', 'clfpenalty': 'l1'}	weighted avg	0.91	0.91	0.91	400	
{'clfalpha': 0.004520353656360241, 'clflearning_rate': 'optimal', 'clfloss': 'log', 'clfpenalty': 'l1'}						
'clfloss': 'log', 'clfpenalty': 'l1'}	9. SGDClassif	 fier()				
'clfloss': 'log', 'clfpenalty': 'l1'} precision recall f1-score	Jielf alpha		365636024	1	oorning rote	al. lontimall
precision recall f1-score support 0 0.98 0.98 0.98 103 1 0.68 0.79 0.73 92 2 0.79 0.65 0.71 101 3 0.98 0.99 0.99 104 accuracy 0.86 400 macro avg 0.86 0.85 0.85 400 weighted avg 0.86 0.86 0.86 400	-				earming_rave	e. Optimai,
1 0.68 0.79 0.73 92 2 0.79 0.65 0.71 101 3 0.98 0.99 0.99 104 accuracy			-		support	
1 0.68 0.79 0.73 92 2 0.79 0.65 0.71 101 3 0.98 0.99 0.99 104 accuracy						
2 0.79 0.65 0.71 101 3 0.98 0.99 0.99 104 accuracy 0.86 400 macro avg 0.86 0.85 0.85 400 weighted avg 0.86 0.86 0.86 400						
3 0.98 0.99 0.99 104 accuracy 0.86 400 macro avg 0.86 0.85 0.85 400 weighted avg 0.86 0.86 0.86 400						
accuracy 0.86 400 macro avg 0.86 0.85 0.85 400 weighted avg 0.86 0.86 0.86 400	2				101	
macro avg 0.86 0.85 0.85 400 weighted avg 0.86 0.86 0.86 400	3	0.98	0.99	0.99	104	
macro avg 0.86 0.85 0.85 400 weighted avg 0.86 0.86 0.86 400	accuracy			0.86	400	
<pre>weighted avg</pre>	•	0.86	0.85	0.85	400	
{'clfvar_smoothing': 1e-20}	_					
{'clfvar_smoothing': 1e-20}						
{'clfvar_smoothing': 1e-20}						
precision recall f1-score support 0 0.95 0.97 0.96 103 1 0.71 0.71 0.71 92 2 0.68 0.67 0.68 101 3 0.91 0.90 0.91 104 accuracy macro avg 0.81 0.81 0.81 400	10. Gaussian	NB()				
precision recall f1-score support 0 0.95 0.97 0.96 103 1 0.71 0.71 0.71 92 2 0.68 0.67 0.68 101 3 0.91 0.90 0.91 104 accuracy macro avg 0.81 0.81 0.81 400			-201			
0 0.95 0.97 0.96 103 1 0.71 0.71 0.71 92 2 0.68 0.67 0.68 101 3 0.91 0.90 0.91 104 accuracy 0.82 400 macro avg 0.81 0.81 0.81 400	(CIIvai_si			f1-score	support	
1 0.71 0.71 0.71 92 2 0.68 0.67 0.68 101 3 0.91 0.90 0.91 104 accuracy 0.82 400 macro avg 0.81 0.81 0.81 400		precision	recarr	II SCOLE	suppor t	
2 0.68 0.67 0.68 101 3 0.91 0.90 0.91 104 accuracy 0.82 400 macro avg 0.81 0.81 0.81 400	0	0.95	0.97	0.96	103	
3 0.91 0.90 0.91 104 accuracy 0.82 400 macro avg 0.81 0.81 0.81 400	1	0.71	0.71	0.71	92	
3 0.91 0.90 0.91 104 accuracy 0.82 400 macro avg 0.81 0.81 0.81 400	2	0.68	0.67			
macro avg 0.81 0.81 400						
macro avg 0.81 0.81 400	200112001			A 82	400	
<u> </u>	•	Λ 01	Λ 01			
weighted avg 0.82 0.82 400	•					
	weighted avg	0.82	0.82	0.82	400	

11. MultinomialNB()

```
{'clf_alpha': 0, 'clf_fit_prior': False}
              precision
                           recall f1-score
                                                support
           0
                   0.88
                              0.78
                                        0.82
                                                    103
                   0.49
                              0.50
                                        0.49
                                                     92
           1
           2
                   0.41
                              0.46
                                        0.43
                                                    101
           3
                   0.63
                              0.62
                                        0.63
                                                    104
                                        0.59
                                                    400
    accuracy
   macro avg
                   0.60
                              0.59
                                        0.59
                                                    400
                              0.59
                                        0.60
weighted avg
                   0.61
                                                    400
```

1.2 Feature Engineering

```
[8]: # df4.drop('four_g',axis = 1,inplace=True)
     df4 = df.copy()
     df4['sc\_diag'] = np.sqrt(df['sc\_h']**2 + df['sc\_w']**2)
     df4['sc\_area'] = df['sc\_h'] * df['sc\_w']
     df4['px_diag'] = np.sqrt(df['px_height']**2 + df['px_width']**2)
     df4['px_area'] = df['px_width'] * df['px_height']
     df4['talk_per_mAh'] = df['talk_time']/ df['battery_power']
     df4['connectivity'] = df['three_g'] + df['four_g'] + df['blue'] + df['wifi'] +

→df['dual_sim']
     df4['ram_per_core'] = df['ram'] / df['n_cores']
     df4['weight_by_thickness'] = df['mobile_wt'] / df['m_dep']
     df4['total_clock_speed'] = df['clock_speed'] * df['n_cores']
     df4['fc_res'] = [1 if df['fc'][i] >=8 else 0 for i in range(len(df))]
     df4['pc_res'] = [1 if df['pc'][i] >=8 else 0 for i in range(len(df)) ]
     df4['sc_res'] = [1 if( df['px_width'][i] * df['px_height'][i] ) >= 921600 else_
     →0 for i in range(len(df))]
     df4['DPIx'] = df['px_width'] / (df['sc_w']*0.394 + 1)
     df4['DPIy'] = df['px height'] / (df['sc h']*0.394 + 1)
     df4['DPI'] = (df['px_width'] * df['px_height']) / (df4['sc_area'] * (0.394)**2_{U}
     + 1)
     df4
     X_train_e, X_test_e, y_train_e, y_test_e =
_
     -train_test_split(df4,df2,random_state=5,shuffle=True,test_size=0.2)
```

1.3 After Feature Engineering

1. KNeighborsClassifier()

{}

G	precision	recall	f1-score	support
0	0.33	0.45	0.38	103
1	0.26	0.27	0.27	92
2	0.24	0.23	0.24	101
3	0.29	0.20	0.24	104
accuracy			0.29	400
macro avg	0.28	0.29	0.28	400
weighted avg	0.28	0.29	0.28	400

2. LogisticRegression()

{}

	precision	recall	f1-score	support
0	0.92	0.68	0.78	103
1	0.42	0.58	0.48	92
2	0.32	0.24	0.27	101
3	0.61	0.71	0.65	104
accuracy			0.55	400
macro avg	0.57	0.55	0.55	400
weighted avg	0.57	0.55	0.55	400

3. DecisionTreeClassifier()

{}

		precision	recall	f1-score	support
	0	0.96	0.92	0.94	103
	1	0.84	0.87	0.86	92
	2	0.82	0.83	0.83	101
	3	0.90	0.90	0.90	104
accura	су			0.88	400
macro a	ıvg	0.88	0.88	0.88	400
weighted a	ıvg	0.88	0.88	0.88	400

4. SVC() precision recall f1-score support 0 0.96 0.94 0.95 103 1 0.80 0.82 0.81 92 2 0.77 0.81 0.79 101 3 0.94 0.88 0.91 104 accuracy 0.86 400 macro avg 0.87 0.86 0.86 400 weighted avg 0.87 0.86 0.87 400
{} precision recall f1-score support 0 0.96 0.94 0.95 103 1 0.80 0.82 0.81 92 2 0.77 0.81 0.79 101 3 0.94 0.88 0.91 104 accuracy 0.86 400 macro avg 0.87 0.86 0.86 400 weighted avg 0.87 0.86 0.87 400
{} precision recall f1-score support 0 0.96 0.94 0.95 103 1 0.80 0.82 0.81 92 2 0.77 0.81 0.79 101 3 0.94 0.88 0.91 104 accuracy 0.86 400 macro avg 0.87 0.86 0.86 400 weighted avg 0.87 0.86 0.87 400
precision recall f1-score support 0 0.96 0.94 0.95 103 1 0.80 0.82 0.81 92 2 0.77 0.81 0.79 101 3 0.94 0.88 0.91 104 accuracy 0.86 400 macro avg 0.87 0.86 0.86 400 weighted avg 0.87 0.86 0.87 400
0 0.96 0.94 0.95 103 1 0.80 0.82 0.81 92 2 0.77 0.81 0.79 101 3 0.94 0.88 0.91 104 accuracy 0.86 400 macro avg 0.87 0.86 0.86 400 weighted avg 0.87 0.86 0.87 400 5. NuSVC()
1 0.80 0.82 0.81 92 2 0.77 0.81 0.79 101 3 0.94 0.88 0.91 104 accuracy 0.86 400 macro avg 0.87 0.86 0.86 400 weighted avg 0.87 0.86 0.87 400
1 0.80 0.82 0.81 92 2 0.77 0.81 0.79 101 3 0.94 0.88 0.91 104 accuracy 0.86 400 macro avg 0.87 0.86 0.86 400 weighted avg 0.87 0.86 0.87 400
3 0.94 0.88 0.91 104 accuracy 0.86 400 macro avg 0.87 0.86 0.86 400 weighted avg 0.87 0.86 0.87 400
accuracy 0.86 400 macro avg 0.87 0.86 0.86 400 weighted avg 0.87 0.86 0.87 400 5. NuSVC()
macro avg 0.87 0.86 0.86 400 weighted avg 0.87 0.86 0.87 400
macro avg 0.87 0.86 0.86 400 weighted avg 0.87 0.86 0.87 400
weighted avg 0.87 0.86 0.87 400
5. NuSVC()

 {}

precision recall f1-score support
r
0 00 00 00 00 00 00 00
0 0.96 0.95 0.96 103 1 0.81 0.82 0.81 92
2 0.77 0.81 0.79 101
3 0.94 0.89 0.92 104
accuracy 0.87 400
macro avg 0.87 0.87 400
weighted avg 0.87 0.87 0.87 400
6. RandomForestClassifier()
{} precision recall f1-score support
precision recall f1-score support

0.95

104

3

0.97

0.93

accurac				0.89	400	
macro av	ğ	0.89	0.89	0.89	400	
weighted av	g	0.89	0.89	0.89	400	
7. AdaBoost	υ ⊥ ձ	assilier()				
{}						
		precision	recall	f1-score	support	
	О	0.93	0.24	0.38	103	
	1	0.45	0.84	0.58	92	
	2	0.52	0.81	0.63	101	
	3	0.95	0.39	0.56	104	
accurac	v			0.56	400	
·	•	0.71	0.57		400	
weighted av	_			0.54		
worghood av	5	0.72	0.00	0.01	100	
8. Gradient	 Boo	ostingClassi	 fier()			
{}						
			11	£1		
		precision	recall	f1-score	support	
	0			f1-score 0.98		
) 1	0.98	0.97			
		0.98 0.84	0.97 0.91	0.98 0.87	103	
:	1	0.98 0.84	0.97 0.91 0.83	0.98 0.87	103 92	
	1 2 3	0.98 0.84 0.89	0.97 0.91 0.83	0.98 0.87 0.86 0.96	103 92 101 104	
accurac	1 2 3	0.98 0.84 0.89 0.96	0.97 0.91 0.83 0.96	0.98 0.87 0.86 0.96	103 92 101 104	
accurac macro av	1 2 3 y	0.98 0.84 0.89	0.97 0.91 0.83 0.96	0.98 0.87 0.86 0.96	103 92 101 104 400 400	
accurac macro av	1 2 3 y	0.98 0.84 0.89 0.96	0.97 0.91 0.83 0.96	0.98 0.87 0.86 0.96	103 92 101 104 400 400	
accurac macro av	1 2 3 y	0.98 0.84 0.89 0.96	0.97 0.91 0.83 0.96	0.98 0.87 0.86 0.96	103 92 101 104 400 400	
accurac macro av	1 2 3 y	0.98 0.84 0.89 0.96	0.97 0.91 0.83 0.96	0.98 0.87 0.86 0.96	103 92 101 104 400 400	
accurac macro av	1 2 3 y y =	0.98 0.84 0.89 0.96	0.97 0.91 0.83 0.96	0.98 0.87 0.86 0.96	103 92 101 104 400 400	
accuracy macro avy weighted avy	1 2 3 y y =	0.98 0.84 0.89 0.96	0.97 0.91 0.83 0.96	0.98 0.87 0.86 0.96	103 92 101 104 400 400	
accurac macro av weighted av	1 2 3 y y =	0.98 0.84 0.89 0.96 0.92 0.92	0.97 0.91 0.83 0.96	0.98 0.87 0.86 0.96 0.92 0.92	103 92 101 104 400 400 400	
accuracy macro ave weighted ave weighted ave serving s	1 2 3 y g g g if:	0.98 0.84 0.89 0.96 0.92 0.92	0.97 0.91 0.83 0.96 0.92 0.92	0.98 0.87 0.86 0.96 0.92 0.92	103 92 101 104 400 400 400	
accuracy macro ave weighted ave selection and selection are selection as a selection and selection are selection as a selection and selection are selection as a selection are selection are selection as a selection are selection are selection as a selection are selection are selection as a selection are se	1 2 3 y y =	0.98 0.84 0.89 0.96 0.92 0.92	0.97 0.91 0.83 0.96 0.92 0.92	0.98 0.87 0.86 0.96 0.92 0.92 	103 92 101 104 400 400 400	

2	0.52	0.65	0.58	101	
3	0.99	0.97	0.98	104	
accuracy			0.76	400	
macro avg	0.75	0.75	0.74	400	
weighted avg	0.76	0.76	0.75	400	
10. GaussianN	NB()				
	precision	recall	f1-score	support	
0	0.89	0.94	0.92	103	
1	0.64	0.60	0.62	92	
2				101	
3				104	
accuracy			0.76	400	
macro avg	0.76	0.76	0.76	400	
weighted avg	0.77	0.76		400	
weighted avg	0.77	0.76		400	
weighted avg	0.77	0.76		400	
				400	
11. Multinomi				400	
11. Multinomi	ialNB()	 	0.76		
11. Multinomi		 	0.76		
	ialNB()	recall	0.76	support	
	ialNB() precision	recall	0.76	support	
	precision 0.75 0.37	recall 0.71 0.39	0.76 	support 103 92	
11. Multinomi 	precision 0.75 0.37	recall 0.71 0.39 0.48	0.76 	support 103 92	
11. Multinomi	precision 0.75 0.37 0.44	recall 0.71 0.39 0.48	0.76 	support 103 92 101 104	
11. Multinomi 	precision 0.75 0.37 0.44 0.56	recall 0.71 0.39 0.48 0.53	0.76 f1-score 0.73 0.38 0.46 0.54 0.53	support 103 92 101 104 400	
11. Multinomi {} 0 1 2 3 accuracy macro avg	precision 0.75 0.37 0.44 0.56	recall 0.71 0.39 0.48 0.53	0.76 f1-score 0.73 0.38 0.46 0.54 0.53 0.53	support 103 92 101 104 400 400	
11. Multinomi 	precision 0.75 0.37 0.44 0.56	recall 0.71 0.39 0.48 0.53	0.76 f1-score 0.73 0.38 0.46 0.54 0.53	support 103 92 101 104 400	
11. Multinomi {} 0 1 2 3 accuracy macro avg	precision 0.75 0.37 0.44 0.56	recall 0.71 0.39 0.48 0.53	0.76 f1-score 0.73 0.38 0.46 0.54 0.53 0.53	support 103 92 101 104 400 400	
11. Multinomi {} 0 1 2 3 accuracy macro avg	precision 0.75 0.37 0.44 0.56	recall 0.71 0.39 0.48 0.53	0.76 f1-score 0.73 0.38 0.46 0.54 0.53 0.53	support 103 92 101 104 400 400	

^{1.} KNeighborsClassifier()

<pre>{'clf_algorithm': 'auto', 'clf_metric': 'cosine', 'clf_n_neighbors': 15, 'clf_weights': 'distance'}</pre>								
	precision	recall	f1-score	support				
0	0.67	0.50	0.57	103				
1	0.30	0.32		92				
2			0.38	*-				
3	0.42	0.46	0.44	104				
accuracy			0.42	400				
macro avg	0.44	0.42	0.43	400				
weighted avg	0.44	0.42	0.43	400				

2. LogisticRegression()

accuracy macro avg

weighted avg

{'clfC':	0.01, '	clfpenalt	cy': 'none	', 'clfsol	lver': 'newton-cg']	}
	prec	ision re	ecall f1-s	score supp	port	
	0	0.91	0.92	0.92	103	
	1	0.73	0.75	0.74	92	
	2	0.67	0.63	0.65	101	
	3	0.81	0.82	0.81	104	

0.78

0.78

0.78

0.78

0.78

400

400

400

3. DecisionTreeClassifier()

0.78

0.78

{'clf__ccp_alpha': 0.00206913808111479, 'clf__criterion': 'entropy',

^{&#}x27;clf min samples split': 2, 'clf splitter': 'best'}

ı
rt
)3
92
)1
)4
00
00

^{&#}x27;clf__max_features': None, 'clf__max_leaf_nodes': None,

4. SVC()	: 'linear'} precision 0.98	 sion_func		: 'ovo', 'clf	degree': 3,
4. SVC()	, 'clfdeci : 'linear'} precision 0.98	 sion_func			degree': 3,
	, 'clfdeci : 'linear'} precision 0.98	sion_funct			degree': 3,
'clfkernel' 0 1 2	: 'linear'} precision 0.98				degree*: 5,
0 1 2	precision 0.98	recall	f1-score	_	
1 2				support	
2	0.96	0.99	0.99	103	
		0.97	0.96	92	
3	0.96	0.96	0.96	101	
	0.99	0.97	0.98	104	
accuracy			0.97	400	
macro avg	0.97	0.97	0.97	400	
weighted avg	0.97	0.97	0.97	400	
5. NuSVC()		clfnu'	0.0616595	50018614822}	'clfgamma': 'scale',
0	0.98	1.00	0.99	103	
1			0.97	92	
2			0.97	101	
3	0.98	1.00	0.99	104	
accuracy			0.98	400	
	0.98	0.98			
weighted avg					
	 stClassifier				

1	0.84	0.95	0.89	92
2	0.95	0.89	0.92	101
3	0.99	0.96	0.98	104
accuracy			0.94	400
macro avg	0.94	0.94	0.93	400
weighted avg	0.94	0.94	0.94	400

7. AdaBoostClassifier()

{'clf_algorithm': 'SAMME.R', 'clf_base_estimator': RandomForestClassifier(),

'clf__n_estimators': 100}

	precision	recall	f1-score	support	
0	0.98	0.91	0.94	103	
1	0.76	0.85	0.80	92	
2	0.80	0.78	0.79	101	
3	0.94	0.92	0.93	104	
accuracy			0.87	400	
macro avg	0.87	0.87	0.87	400	
weighted avg	0.87	0.87	0.87	400	

8. GradientBoostingClassifier()

{'clf_criterion': 'friedman_mse', 'clf_learning_rate': 1.0, 'clf_loss':

'deviance', 'clf__max_features': 'sqrt', 'clf__n_estimators': 300}

	precision	recall	f1-score	support
0 1 2 3	0.99 0.83 0.82 0.93	0.94 0.87 0.83 0.92	0.97 0.85 0.82 0.93	103 92 101 104
accuracy macro avg weighted avg	0.89 0.89	0.89 0.89	0.89 0.89 0.89	400 400 400

^{9.} SGDClassifier()

{'clf_al	-	: 0.01, 'clf_ ': '11'}	 _learnin	g_rate': '	optimal',	'clfloss':	'log',
		precision	recall	f1-score	support		
	0	0.97	0.98	0.98	103		
	1	0.73	0.75	0.74	92		
	2	0.77	0.73	0.75	101		
	3	0.96	0.98	0.97	104		
accur	racy			0.86	400		
macro	avg	0.86	0.86	0.86	400		
weighted	avg	0.86	0.86	0.86	400		

10. GaussianNB()

{'clf var smoothing': 1e-20}

C CIIVGI_DI	. 10	20)		
	precision	recall	f1-score	support
0	0.89	0.94	0.92	103
1	0.64	0.60	0.62	92
2	0.60	0.68	0.64	101
3	0.92	0.80	0.86	104
accuracy			0.76	400
macro avg	0.76	0.76	0.76	400
weighted avg	0.77	0.76	0.76	400

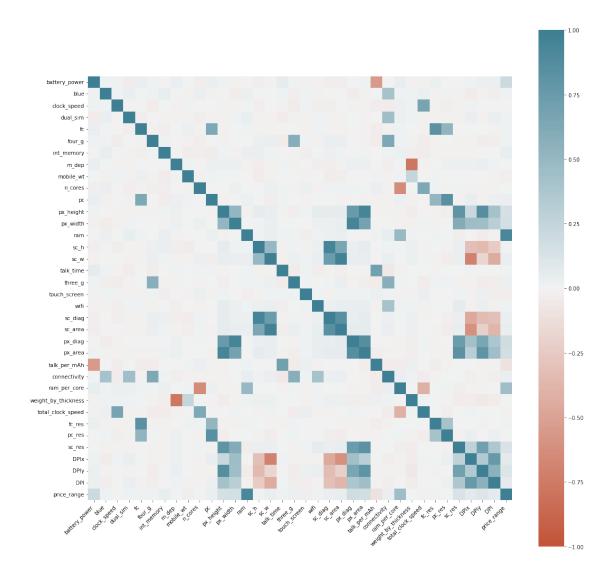
11. MultinomialNB()

{'clf_alpha': 0, 'clf_fit_prior': True} precision recall f1-score support 0 0.75 0.71 0.73 103 1 0.37 0.39 0.38 92 0.44 2 0.48 0.46 101 0.56 0.53 0.54 104 accuracy 0.53 400 macro avg 0.53 0.53 0.53 400 weighted avg 0.54 0.53 0.53 400

1.4 Feature Selection

1.4.1 Correlation

```
[10]: df5 = df4.copy()
      df5['price_range'] = df2
      corr = df5.corr()
      plt.figure(figsize = (18,18))
      ax = sns.heatmap(
          corr,
          vmin=-1, vmax=1, center=0,
          cmap=sns.diverging_palette(20, 220, n=200),
          square=True
      ax.set_xticklabels(
          ax.get_xticklabels(),
          rotation=45,
          horizontalalignment='right'
      );
      d = 0.01
      col = []
      for idx,i in enumerate(corr['price_range']):
          if i \le d and i \ge -d:
              col.append(corr.columns[idx])
              print(corr.columns[idx])
      df5 = df5.drop(col,axis = 1)
      df5 = df5.drop('price_range',axis = 1)
      df5
      X_train_ec, X_test_ec, y_train_ec, y_test_ec =
__
      -train_test_split(df5,df2,random_state=5,shuffle=True,test_size=0.2)
     clock_speed
     m_dep
     n_cores
     weight_by_thickness
     total_clock_speed
     pc_res
```



1.4.2 After Feature Engg + Correlation

{}

precision recall f1-score support
0 0.33 0.45 0.38 103

1	0.26	0.27	0.27	92
2	0.25	0.24	0.24	101
3	0.30	0.20	0.24	104
accuracy			0.29	400
macro avg	0.29	0.29	0.28	400
weighted avg	0.29	0.29	0.28	400

2. LogisticRegression()

------{}

	precision	recall	f1-score	support
0 1 2 3	0.92 0.40 0.35 0.56	0.67 0.63 0.14 0.75	0.78 0.49 0.20 0.64	103 92 101 104
accuracy macro avg weighted avg	0.56 0.56	0.55 0.55	0.55 0.53 0.53	400 400 400

3. DecisionTreeClassifier()

{}

	precision	recall	f1-score	support
0 1 2 3	0.93 0.85 0.85 0.92	0.93 0.86 0.85	0.93 0.85 0.85	103 92 101
accuracy macro avg weighted avg	0.92 0.89 0.89	0.91 0.89 0.89	0.92 0.89 0.89 0.89	400 400 400

4. SVC()

{}

	precision	recall	f1-score	support	
0	0.98	0.94	0.96	103	
1				92	
2				101	
3				104	
Ü	0.51	0.50	0.02	101	
accuracy			0.89	400	
macro avg	0.89	0.88	0.88	400	
weighted avg	0.89	0.89	0.89	400	
5. NuSVC()					
{}					
	precision	recall	f1-score	support	
0	0.98	0.94	0.96	103	
1	0.81	0.87	0.84	92	
2	0.80	0.83	0.82	101	
3	0.96	0.89		104	
			0.89	400	
accuracy	0.00	0.00			
macro avg		0.88		400	
weighted avg	0.89	0.89	0.89	400	
6. RandomFore	estClassifier	·()			
{}					
-	precision	recall	f1-score	support	
0	0.99	0.95	0.97	103	
1	0.79	0.88	0.84	92	
2	0.81	0.84	0.83	101	
3	1.00	0.90	0.95	104	
2001122011			0.90	400	
accuracy	0.00	0 00	0.90	400	
macro avg	0.90	0.89	0.90	400	

0.90

400

0.90

0.90

macro avg weighted avg

7. AdaBoostClassifier()

U.	

{}				
	precision	recall	f1-score	support
0	0.87	0.46	0.60	103
1	0.51	0.77	0.61	92
2	0.67	0.72	0.70	101
3	0.84	0.79	0.81	104
accuracy			0.68	400
macro avg	0.72	0.68	0.68	400
weighted avg	0.73	0.68	0.68	400

8. GradientBoostingClassifier()

{}	precision	recall	f1-score	support
0	0.98	0.97	0.98	103
1	0.85	0.89	0.87	92
2	0.86	0.82	0.84	101
3	0.94	0.94	0.94	104
accuracy			0.91	400
macro avg	0.91	0.91	0.91	400
weighted avg	0.91	0.91	0.91	400

9. SGDClassifier()

-----{}

	precision	recall	f1-score	support
0	1.00	0.97	0.99	103
1	0.58	0.57	0.57	92
2	0.61	0.64	0.63	101
3	0.98	0.98	0.98	104
accuracy			0.80	400
macro avg	0.79	0.79	0.79	400
weighted avg	0.80	0.80	0.80	400

10. GaussianN	IB()				
{}					
	precision	recall	f1-score	support	
0	0.90	0.93	0.91	103	
1				92	
2	0.61	0.69	0.65	101	
3	0.93	0.80		104	
accuracy			0.77	400	
macro avg	0.77	0.76	0.77		
weighted avg				400	
11. Multinomi	.alNB()				
 {}					
	precision	recall	f1-score	support	
0	0.75	0.70	0.72	103	
1	0.41	0.42	0.42	92	
2				101	
3	0.58	0.51	0.54	104	
accuracy			0.53	400	
•	0.54	0.53	0.53	400	
weighted avg	0.54	0.53	0.53	400	
 WITH GridSear	 ch				
1. KNeighbors	Classifier()				
			etric': 'co	osine', 'c	lfn_neighbors': 18,
			f1-score	support	
0	0.71	0.52	0.60	103	

2	0.38	0.42	0.40	101		
3	0.42	0.47	0.45	104		
accuracy			0.43	400		
macro avg	0.45	0.43	0.44	400		
weighted avg	0.46	0.43	0.44	400		
weighted avg	0.40	0.40	0.11	400		
2. LogisticRe	egression()					
{'clfC': 0.	.029209037170	322485, '	clfpenal	ty': '12',	<pre>'clfsolver':</pre>	'newton-
cg'}						
	precision	recall	f1-score	support		
0	0.90	0.92	0.91	103		
1	0.75	0.72	0.73	92		
2	0.65	0.66	0.66	101		
3	0.80	0.79	0.79	104		
_						
accuracy			0.78	400		
macro avg	0.77	0.77	0.77	400		
weighted avg	0.77	0.78	0.77	400		
weighted avg	0.11	0.10	0.77	100		
3. DecisionTr		-()				
0. 5001510111	. 000140011101					
{'clf ccn al	nha' 0 0037	 7926901907	3225 'clf	criterio	n': 'entropy',	
clfmax_fea	-					
clfmax_rec					,	
cri_min_sam			_			
	precision	recall	11-score	support		
^	0.04	0.00	0.04	100		
0	0.94	0.93	0.94	103		

0.86

0.85

0.93

0.90

0.89

0.90

92

101

104

400

400

400

1

2

3

accuracy

macro avg

weighted avg

0.82

0.88

0.93

0.89

0.90

0.90

0.82

0.92

0.89

0.90

^{4.} SVC()

		sion_func	tion_shape	': 'ovo',	'clfdegree': 3,
'clfkernel':		11	£1		
P	recision	recall	II-score	support	
0	0.99	0.99	0.99	103	
1	0.97	0.98	0.97	92	
2	0.96	0.97	0.97	101	
3	0.99	0.97	0.98	104	
accuracy			0.98	400	
macro avg	0.98	0.98	0.98	400	
weighted avg	0.98		0.98	400	
5. NuSVC()					
		-		_	3, 'clfgamma': 'scale',
<pre>'clfkernel':</pre>					2}
p	recision	recall	f1-score	support	
0	0.98	1 00	0.99	103	
1		0.97		92	
2 3	0.98	0.96 0.99	0.97 0.99	101 104	
3	0.96	0.99	0.99	104	
accuracy			0.98	400	
•	0.98	0.98		400	
weighted avg	0.98	0.98	0.98	400	
6. RandomForest	Classifier	 ')			
o. Italiaomi orebu	Olubbiliol				
{'clf_bootstra	.p': True, '	clfcri	terion': 'e	entropy',	'clfmax_depth': None,
clfmax_featu	_				_
	recision		f1-score		
0	0.96	0.95	0.96	103	
1			0.89	92	
2	0.91	0.89		101	
3		0.94		104	
2601720			0.02	400	
accuracy	0.00	2 22	0.93	400	

0.93

400

macro avg 0.93 0.93

weighted	avg	0.93	0.93	0.93	400	
7. AdaBoo	stCla	assifier()				
	gori	thm': 'SAMME.	 R', 'clf	base_est:	imator': Random	ForestClassifier(),
'clfn_e	stima	ators': 100}				
		precision	recall	f1-score	support	
	0	0.98	0.95	0.97	103	
	1	0.82	0.87	0.85	92	
	2	0.84	0.86	0.85	101	
	3	0.98	0.93	0.96	104	
accur	acy			0.91	400	
macro	avg	0.91	0.90	0.90	400	
weighted	avg	0.91	0.91	0.91	400	
			ures': '	sqrt', 'cli	n_estimators	1, 'clfloss': ': 700}
	0	0.08	0.06	0.97	102	
	1				92	
	2			0.82	101	
	3	0.94	0.92	0.93	104	
accur				0.89	400	
macro	•	0.89	0.89			
weighted	_		0.89		400	
9. SGDCla						
	ssif	ier()				
	 .pha'	: 0.007278953			earning_rate':	'optimal',
	 .pha'		penalty'	: '11'}	earning_rate': support	'optimal',

1	0.73	0.76	0.74	92
2	0.78	0.75	0.77	101
3	0.99	0.99	0.99	104
accuracy			0.88	400
macro avg	0.87	0.87	0.87	400
weighted avg	0.88	0.88	0.88	400

10. GaussianNB()

{'clf__var_smoothing': 1e-20}

	precision	recall	f1-score	support
0	0.90	0.93	0.91	103
1	0.64	0.63	0.64	92
2	0.61	0.69	0.65	101
3	0.93	0.80	0.86	104
accuracy			0.77	400
macro avg	0.77	0.76	0.77	400
weighted avg	0.78	0.77	0.77	400

11. MultinomialNB()

.	precision	recall	f1-score	support
0	0.76	0.70	0.73	103
1	0.41	0.42	0.41	92
2	0.41	0.48	0.44	101
3	0.58	0.51	0.54	104
accuracy			0.53	400
macro avg	0.54	0.53	0.53	400
weighted avg	0.54	0.53	0.53	400

1.4.3 Variance Inflation Factor

```
[12]: scalar = StandardScaler()
     X_tranform = scalar.fit_transform(df5)
     df6 = pd.DataFrame(X_tranform,columns=df5.columns)
     vif = [variance_inflation_factor(df6.values,i) for i in range(df6.shape[1])]
     d = 5 # Threshold for Variance Inflation,
     col = []
     for idx,i in enumerate(vif):
         if i >= d:
            col.append(df6.columns[idx])
            print(df6.columns[idx])
     df6 = df6.drop(col,axis = 1)
     X_train_ecv, X_test_ecv, y_train_ecv, y_test_ecv =_
      -train_test_split(df6,df2,random_state=5,shuffle=True,test_size=0.2)
    blue
    dual_sim
    four_g
    px_height
    px_width
    sc_h
    SC_W
    talk_time
    three_g
    wifi
    sc_diag
    sc_area
    px_diag
    px_area
    talk_per_mAh
    connectivity
    DPIx
    DPIy
[13]: Classification(X_train_ecv, X_test_ecv, y_train_ecv, __
     print('WITH GridSearch'+'\n'+"-"*80+'\n'+"-"*80 )
     Classification(X_train_ecv, X_test_ecv, y_train_ecv,_

    KNeighborsClassifier()

    {}
                 precision recall f1-score
                                               support
               0
                      0.82
                               0.84
                                         0.83
                                                   103
```

1	0.57	0.66	0.61	92
2	0.59	0.57	0.58	101
3	0.84	0.71	0.77	104
accuracy			0.70	400
macro avg	0.70	0.70	0.70	400
weighted avg	0.71	0.70	0.70	400

2. LogisticRegression()

------{}

	precision	recall	f1-score	support
0 1 2 3	0.93 0.85 0.84 0.94	0.97 0.84 0.83 0.92	0.95 0.84 0.84 0.93	103 92 101 104
accuracy macro avg weighted avg	0.89	0.89	0.89 0.89 0.89	400 400 400

3. DecisionTreeClassifier()

{}

		precision	recall	f1-score	support
	0	0.93	0.89	0.91 0.81	103 92
	2 3	0.78 0.89	0.79 0.89	0.79 0.89	101 104
accura macro a weighted a	avg	0.85 0.85	0.85 0.85	0.85 0.85 0.85	400 400 400

{}

^{4.} SVC()

0 1 2 3 accuracy macro avg weighted avg 5. NuSVC()	0.84 0.79 0.90	0.84 0.81 0.88	0.84 0.80 0.89 0.87		
1 2 3 accuracy macro avg weighted avg	0.84 0.79 0.90	0.84 0.81 0.88	0.84 0.80 0.89 0.87	92 101 104 400 400	
accuracy macro avg weighted avg	0.79 0.90 0.87	0.81 0.88 0.87	0.80 0.89 0.87 0.87	101 104 400 400	
accuracy macro avg weighted avg	0.90	0.88	0.89 0.87 0.87	104 400 400	
accuracy macro avg weighted avg	0.87	0.87	0.87 0.87	400 400	
macro avg			0.87	400	
macro avg			0.87	400	
weighted avg					
	 			400	
5. NuSVC()					
5. NuSVC()					
{}					
	precision	recall	f1-score	support	
0	0.95	0.93	0.94	103	
1			0.83		
2				101	
3				104	
accuracy				400	
•	0.86			400	
weighted avg	0.87	0.86	0.87	400	
6. RandomFores	stClassifier	()			
{}					
	precision	recall	f1-score	support	
0	0.96	0.94	0.95	103	
1	0.78	0.87	0.82	92	
2	0.82	0.75	0.78	101	
3	0.91	0.91	0.91	104	
accuracy			0.87	400	
	0.87 0.87	0.87	0.87	400	
macro avg weighted avg		0.87	0.87	400	

7. AdaBoostClassifier()

G	pı	recision	recall	f1-score	support
	0	0.94	0.58	0.72	103
	1	0.58	0.90	0.71	92
	2	0.64	0.70	0.67	101
	3	0.84	0.66	0.74	104

 accuracy
 0.71
 400

 macro avg
 0.75
 0.71
 0.71
 400

 weighted avg
 0.76
 0.71
 0.71
 400

8. GradientBoostingClassifier()

{}				
	precision	recall	f1-score	support
0	0.05	0.04	0.05	100
0	0.95	0.94	0.95	103
1	0.79	0.84	0.81	92
2	0.86	0.76	0.81	101
3	0.91	0.97	0.94	104
accuracy			0.88	400
macro avg	0.88	0.88	0.88	400
weighted avg	0.88	0.88	0.88	400

SGDClassifier()

-----{}

G	precision	recall	f1-score	support
0	0.94	0.98	0.96	103
1	0.57	0.70	0.62	92
2	0.65	0.46	0.53	101
3	0.91	0.95	0.93	104
accuracy			0.78	400
macro avg	0.77	0.77	0.76	400
weighted avg	0.77	0.78	0.77	400

10. Gaussian	IB()				
{}					
	precision	recall	f1-score	support	
0	0.91	0.93	0.92	103	
1			0.68	92	
2				101	
3				104	
0.000000000			0.70	400	
accuracy	0.77	0.77		400	
<pre>macro avg weighted avg</pre>			0.77 0.77	400 400	
#018H00# 478	0.11	0.10	0111	100	
11. Multinomi	raing()				
{}					
	precision	recall	f1-score	support	
0	0.85	0.79	0.82	103	
1	0.40	0.48	0.43	92	
2	0.42	0.32	0.36	101	
3	0.56	0.63	0.60	104	
accuracy			0.56	400	
•	0.56	0.55		400	
weighted avg			0.56	400	
WITH GridSear	rch				
1. KNeighbors	Classifier()				
_			etric': 'c	ityblock',	'clfn_neighbors': 24,
'clfweights			f1_g	aun art	
	precision	recall	f1-score	support	
0	0.86	0.88	0.87	103	

2	0.70	0.73	0.71	101
	0.88	0.81	0.84	104
accuracy macro avg weighted avg	0.78 0.79	0.78 0.79	0.79 0.78 0.79	400 400 400

2. LogisticRegression()

{'clf_C': 0.01, 'clf_penalty': 'none', 'clf_solver': 'saga'}

0 0.93 0.97 0.95 103 1 0.84 0.84 0.84 92 2 0.85 0.82 0.83 101	0 -
2 0.85 0.82 0.83 101	
3 0.94 0.93 0.94 104	
accuracy 0.89 400	
macro avg 0.89 0.89 0.89 400	
weighted avg 0.89 0.89 0.89 400	

support

precision

{'clf_ccp_alpha': 0.0011288378916846896, 'clf_criterion': 'gini',

recall f1-score

^{&#}x27;clf__max_features': None, 'clf__max_leaf_nodes': None,
'clf__min_samples_split': 12, 'clf__splitter': 'random'}

	-			
0	0.93	0.97	0.95	103
1	0.84	0.86	0.85	92
2	0.87	0.70	0.78	101
3	0.84	0.94	0.89	104
accuracy			0.87	400
macro avg	0.87	0.87	0.87	400
weighted avg	0.87	0.87	0.87	400

4. SVC()

DecisionTreeClassifier()

⁴ gra()

{'clfC'	: 0.8, 'clfdec	ision_fun	ction_shape	e': 'ovo',	<pre>'clfdegree': 3,</pre>
'clfker	nel': 'linear'}				
	precision	recall	f1-score	support	

	precision	recall	f1-score	support	
0	0.93	0.97	0.95	103	
1	0.84	0.84	0.84	92	
2	0.83	0.81	0.82	101	
3	0.93	0.91	0.92	104	
accuracy			0.89	400	
macro avg	0.88	0.88	0.88	400	
weighted avg	0.88	0.89	0.88	400	

5. NuSVC()

{'clf__decision_function_shape': 'ovo', 'clf__degree': 3, 'clf__gamma': 'scale',
'clf__kernel': 'linear', 'clf__nu': 0.06456542290346556}

	precision	recall	f1-score	support	
0	0.91	0.93	0.92	103	
1	0.76	0.74	0.75	92	
2	0.72	0.75	0.74	101	
3	0.90	0.87	0.88	104	
accuracy			0.82	400	
macro avg	0.82	0.82	0.82	400	
weighted avg	0.83	0.82	0.83	400	

6. RandomForestClassifier()

weighted avg

{'clf_bootstrap': True, 'clf_criterion': 'entropy', 'clf_max_depth': None,
'clf_max_features': None, 'clf_n_estimators': 250}

precision recall f1-score support 0 0.95 0.94 0.95 103 0.80 0.90 1 0.85 92 2 0.90 0.77 0.83 101 3 0.93 0.95 0.94 104 accuracy 0.89 400 macro avg 0.89 0.89 0.89 400

0.89

0.90

0.89

400

7. AdaBoostClassifier()

{'clf_algorithm': 'SAMME', 'clf_base_estimator': RandomForestClassifier(),

'clf__n_estimators': 50}

	precision	recall	f1-score	support
	brecipion	recarr	11 20016	Suppor 0
0	0.96	0.93	0.95	103
1	0.79	0.88	0.83	92
2	0.84	0.76	0.80	101
3	0.91	0.92	0.92	104
accuracy			0.88	400
macro avg	0.87	0.87	0.87	400
weighted avg	0.88	0.88	0.87	400

8. GradientBoostingClassifier()

{'clf_criterion': 'friedman_mse', 'clf_learning_rate': 0.1, 'clf_loss':

'deviance', 'clf__max_features': 'sqrt', 'clf__n_estimators': 700} precision recall f1-score support

	1			
0	0.97	0.96	0.97	103
1	0.82	0.88	0.85	92
2	0.85	0.78	0.81	101
3	0.92	0.94	0.93	104
accuracy			0.89	400
macro avg	0.89	0.89	0.89	400
weighted avg	0.89	0.89	0.89	400

O CCDC1---ifi---()

9. SGDClassifier()

{'clf_alpha': 0.005298316906283708, 'clf_learning_rate': 'optimal',

^{&#}x27;clf loss': 'log', 'clf penalty': 'l1'}

.cli_loss.:	.rog.' .cri-	_penarty:	: .TI.}		
	precision	recall	f1-score	support	
0	0.92	0.97	0.94	103	
1	0.71	0.66	0.69	92	

2	0.72	0.70	0.71	101
3	0.92	0.94	0.93	104
accuracy			0.82	400
macro avg	0.82	0.82	0.82	400
weighted avg	0.82	0.82	0.82	400

10. GaussianNB()

{'clf__var_smoothing': 1e-20}

	precision	recall	f1-score	support
0 1 2 3	0.91 0.68 0.63 0.86	0.93 0.67 0.63 0.85	0.92 0.68 0.63 0.85	103 92 101 104
accuracy macro avg weighted avg	0.77 0.77	0.77 0.78	0.78 0.77 0.77	400 400 400

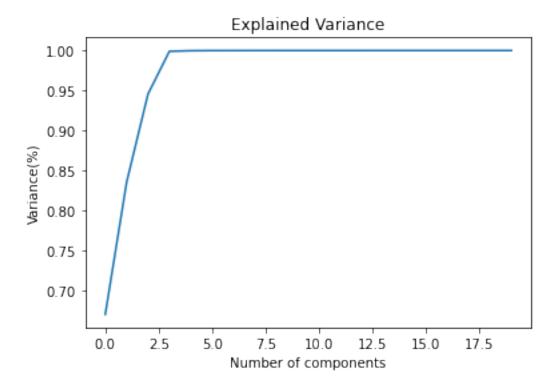
11. MultinomialNB()

{'clfalpha'	: 0.562341325	<pre>'clffit_prior': True}</pre>		
	precision	recall	f1-score	support
0	0.85	0.78	0.81	103
1	0.39	0.48	0.43	92
2	0.42	0.32	0.36	101
3	0.56	0.63	0.60	104
accuracy			0.56	400
macro avg	0.56	0.55	0.55	400
weighted avg	0.56	0.56	0.55	400

1.5 PCA

1.5.1 On Raw data

```
[14]: pca = PCA()
     principalComp = pca.fit_transform(df)
     plt.figure()
     plt.plot(np.cumsum(pca.explained_variance_ratio_))
     plt.xlabel('Number of components')
     plt.ylabel('Variance(%)')
     plt.title('Explained Variance')
     plt.show()
     n_components=4
     pca = PCA(n_components=n_components)
     new_data = pca.fit_transform(df)
     df7 = pd.DataFrame(new_data,columns = [f'column {i}' for i in_
      →range(n_components)])
     df7
     X_train, X_test, y_train, y_test =
      -train_test_split(df7,df2,random_state=5,shuffle=True,test_size=0.2)
     Classification(X_train, X_test, y_train, y_test,data,'PCA_raw')
     print('WITH GridSearch'+'\n'+"-"*80+'\n'+"-"*80 )
     Classification(X_train, X_test, y_train, __
```



1. KNeighborsClassifier()

{}

S	precision	recall	f1-score	support
0	0.96	0.98	0.97	103
1	0.88	0.92	0.90	92
2	0.89	0.87	0.88	101
3	0.97	0.92	0.95	104
accuracy			0.93	400
macro avg	0.92	0.92	0.92	400
weighted avg	0.93	0.93	0.93	400

2. LogisticRegression()

{}

	precision	recall	f1-score	support
0 1	0.96 0.94	0.99 0.95	0.98 0.94	103 92
2	0.95 0.98	0.93 0.96	0.94 0.97	101 104
accuracy macro avg	0.96	0.96	0.96	400 400
weighted avg	0.96	0.96	0.96	400

3. DecisionTreeClassifier()

	precision	recall	f1-score	support
0	0.92	0.93	0.93	103
1	0.84	0.86	0.85	92
2	0.85	0.85	0.85	101
3	0.93	0.90	0.92	104
accuracy			0.89	400
macro avg	0.89	0.89	0.89	400
weighted avg	0.89	0.89	0.89	400

4. SVC()				
{}			6.4	
	precision	recall	il-score	support
0	0.98	0.98	0.98	103
1			0.92	92
2	0.91	0.90	0.91	101
3	0.99	0.93	0.96	104
20017201			0 9/1	400
accuracy macro avg	0.94	0.94		
weighted avg				
5. NuSVC()				
J. Nubve()				
{}				
	precision	recall	f1-score	support
0	0.08	0.96	0.97	103
1			0.89	
2			0.87	
3			0.94	104
			0.00	400
accuracy	0.92	0.00		400 400
<pre>macro avg weighted avg</pre>	0.92	0.92	0.92	400
merkuren avk	0.52	0.32	0.92	400
6. RandomFore	estClassifier	()		
{}				
	precision	recall	f1-score	support
0	0.95	0.95	0.95	103
1	0.88	0.93	0.91	92
2	0.94	0.89	0.91	101
_	0.00	0.05	0 00	404

0.96

104

3

0.96

0.95

				100	
accuracy	0.00		0.93	400	
_	0.93				
weighted avg	0.93	0.93	0.93	400	
7. AdaBoostCl	assifier()				
	precision	recall	f1-score	support	
0	0.82	0.70	0.75	103	
1	0.56	0.72	0.63	92	
2	0.70	0.72	0.71	101	
3	0.92	0.79	0.85	104	
accuracy			0.73	400	
•	0.75	0.73	0.74	400	
weighted avg				400	
	oostingClass	 ifier()			
		 ifier()			
	oostingClass	ifier()			
	precision	ifier() recall			
Θ	precision	recall	f1-score	support	
{}	precision	recall	f1-score	support	
(} 0 1	precision 0.96 0.84	recall 0.93 0.95	f1-score 0.95 0.89 0.88	support 103 92	
0 1 2 3	precision 0.96 0.84 0.90	recall 0.93 0.95 0.86	f1-score 0.95 0.89 0.88 0.93	support 103 92 101 104	
0 1 2 3 accuracy	precision 0.96 0.84 0.90 0.95	recall 0.93 0.95 0.86 0.91	f1-score 0.95 0.89 0.88 0.93	support 103 92 101 104 400	
{} 0 1 2 3 accuracy macro avg	precision 0.96 0.84 0.90 0.95	recall 0.93 0.95 0.86 0.91	f1-score 0.95 0.89 0.88 0.93 0.91	support 103 92 101 104 400 400	
{} 0 1 2 3 accuracy macro avg	precision 0.96 0.84 0.90 0.95	recall 0.93 0.95 0.86 0.91	f1-score 0.95 0.89 0.88 0.93 0.91	support 103 92 101 104 400 400	
{} 0 1 2 3 accuracy macro avg weighted avg	precision 0.96 0.84 0.90 0.95	recall 0.93 0.95 0.86 0.91 0.91	f1-score 0.95 0.89 0.88 0.93 0.91	support 103 92 101 104 400 400	
{} 0 1 2 3 accuracy macro avg weighted avg	precision 0.96 0.84 0.90 0.95 0.91	recall 0.93 0.95 0.86 0.91 0.91	f1-score 0.95 0.89 0.88 0.93 0.91	support 103 92 101 104 400 400	
0 1 2 3 accuracy macro avg weighted avg	precision 0.96 0.84 0.90 0.95	recall 0.93 0.95 0.86 0.91 0.91	f1-score 0.95 0.89 0.88 0.93 0.91	support 103 92 101 104 400 400	
0 1 2 3 accuracy macro avg weighted avg	precision 0.96 0.84 0.90 0.95 0.91	recall 0.93 0.95 0.86 0.91 0.91	f1-score 0.95 0.89 0.88 0.93 0.91 0.91 0.91	support 103 92 101 104 400 400 400	
{} 0 1 2 3 accuracy macro avg weighted avg	precision 0.96 0.84 0.90 0.95 0.91 0.91 precision	recall 0.93 0.95 0.86 0.91 0.91 recall	f1-score 0.95 0.89 0.88 0.93 0.91 0.91 0.91	support 103 92 101 104 400 400 400 support	

2 0.55 0.30 0.38 101 3 0.92 0.98 0.95 104 accuracy 0.75 400 macro avg 0.74 0.75 0.73 400 weighted avg 0.75 0.75 0.74 400
accuracy 0.75 400 macro avg 0.74 0.75 0.73 400
accuracy 0.75 400 macro avg 0.74 0.75 0.73 400
macro avg 0.74 0.75 0.73 400
macro avg 0.74 0.75 0.73 400
•
5
10. GaussianNB()
{}
precision recall f1-score support
0 0.95 0.94 0.95 103
1 0.70 0.71 0.70 92
0 00 00 000 000 000
3 0.87 0.90 0.89 104
accuracy 0.80 400
macro avg 0.80 0.80 0.80 400
weighted avg 0.80 0.80 0.80 400
11 Mul+inomiolND()
 11. MultinomialNB()
11. MultinomialNB()
11. MultinomialNB(){} precision recall f1-score support
11. MultinomialNB()
11. MultinomialNB() precision recall f1-score support 0 0.84 0.69 0.76 103 1 0.36 0.49 0.41 92 2 0.31 0.23 0.26 101 3 0.52 0.58 0.55 104 accuracy 0.50 400 macro avg 0.51 0.50 0.49 400
11. MultinomialNB()
11. MultinomialNB()
11. MultinomialNB() precision recall f1-score support 0 0.84 0.69 0.76 103 1 0.36 0.49 0.41 92 2 0.31 0.23 0.26 101 3 0.52 0.58 0.55 104 accuracy 0.50 400 macro avg 0.51 0.50 0.49 400 weighted avg 0.51 0.50 0.50 400
11. MultinomialNB()
11. MultinomialNB() precision recall f1-score support 0 0.84 0.69 0.76 103 1 0.36 0.49 0.41 92 2 0.31 0.23 0.26 101 3 0.52 0.58 0.55 104 accuracy 0.50 400 macro avg 0.51 0.50 0.49 400 weighted avg 0.51 0.50 0.50 400

^{1.} KNeighborsClassifier()

<pre>{'clf_algori 'clf_weights</pre>	-		etric': 'eı	ıclidean',	'clfn_neighbors': 22,
0	precision		f1-score	support	
0	0.98	0.98	0.98	103	
1	0.90	0.95	0.92	92	
2	0.92	0.91	0.92	101	
3	0.99	0.95	0.97	104	
accuracy			0.95	400	
macro avg	0.95	0.95	0.95	400	
weighted avg	0.95	0.95	0.95	400	

2. LogisticRegression()

{'clfC': 0.	01, 'clfper	nalty': '	12', 'clf_	_solver':	'newton-cg'}
	precision	recall	f1-score	support	
0	1.00	0.98	0.99	103	
1	0.92	0.99	0.95	92	
2	0.95	0.91	0.93	101	
3	0.97	0.96	0.97	104	
accuracy			0.96	400	
macro avg	0.96	0.96	0.96	400	
weighted avg	0.96	0.96	0.96	400	

3. DecisionTreeClassifier()

^{&#}x27;clf_min_samples_split': 2, 'clf_splitter': 'best'}

CTTmTH_Sam	bres_sbric	. 2, С11_	-phirener .	Desc J
	precision	recall	f1-score	support
0	0.94	0.94	0.94	103
1	0.82	0.87	0.84	92
2	0.85	0.81	0.83	101
3	0.93	0.91	0.92	104
accuracy			0.89	400
macro avg	0.88	0.88	0.88	400

^{{&#}x27;clf_ccp_alpha': 0.0014384498882876629, 'clf_criterion': 'entropy',

^{&#}x27;clf__max_features': None, 'clf__max_leaf_nodes': None,

	avg	0.89	0.89	0.89	400	
4. SVC()						
			 sion_func	tion_shape	': 'ovo', 'c	lfdegree': 3,
'clfker		'linear'} recision	recall	f1-score	support	
	0	1 00	0.98	0.99	103	
	1			0.99	92	
	2	0.93		0.96	101	
	3					
	3	0.98	0.95	0.97	104	
accur	cacy			0.96	400	
macro	avg	0.96	0.96	0.96	400	
weighted	avg	0.96	0.96	0.96	400	
	nel':	'linear',	clfnu'		5784384138}	, 'clfgamma': 'scale'
	rnel': pi	'linear', recision	'clfnu' recall	: 0.0707949 f1-score	5784384138} support	, 'clfgamma': 'scale'
	rnel': pi 0	'linear', recision 1.00	'clfnu' recall 0.98	: 0.070794! f1-score 0.99	5784384138} support 103	, 'clfgamma': 'scale'
	rnel': pi 0 1	'linear', recision 1.00 0.94	'clfnu' recall 0.98 0.99	: 0.0707948 f1-score 0.99 0.96	5784384138} support 103 92	, 'clfgamma': 'scale'
	o 1 2	'linear', recision 1.00 0.94 0.95	'clfnu' recall 0.98 0.99 0.94	: 0.0707948 f1-score 0.99 0.96 0.95	5784384138} support 103 92 101	, 'clfgamma': 'scale'
	rnel': pi 0 1	'linear', recision 1.00 0.94	'clfnu' recall 0.98 0.99	: 0.0707948 f1-score 0.99 0.96 0.95	5784384138} support 103 92	, 'clfgamma': 'scale'
	o 1 2 3	'linear', recision 1.00 0.94 0.95	'clfnu' recall 0.98 0.99 0.94	: 0.0707948 f1-score 0.99 0.96 0.95	5784384138} support 103 92 101 104	, 'clfgamma': 'scale'
'clfker	o 1 2 3	'linear', recision 1.00 0.94 0.95 0.98	'clfnu' recall 0.98 0.99 0.94 0.96	: 0.0707948 f1-score 0.99 0.96 0.95 0.97	5784384138} support 103 92 101 104 400	, 'clfgamma': 'scale'
'clfker	o 1 2 3 cacy avg	'linear', recision 1.00 0.94 0.95 0.98	'clfnu' recall 0.98 0.99 0.94 0.96	: 0.0707948 f1-score 0.99 0.96 0.95 0.97	5784384138} support 103 92 101 104 400 400	, 'clfgamma': 'scale'
accur macro weighted	o 1 2 3 cacy avg avg	'linear', recision 1.00 0.94 0.95 0.98	'clfnu' recall 0.98 0.99 0.94 0.96 0.97	: 0.0707945 f1-score 0.99 0.96 0.95 0.97	5784384138} support 103 92 101 104 400 400	, 'clfgamma': 'scale'
accur macro weighted	o 1 2 3 cacy avg avg	'linear', recision 1.00 0.94 0.95 0.98 0.97 0.97	'clfnu' recall 0.98 0.99 0.94 0.96	: 0.0707945 f1-score 0.99 0.96 0.95 0.97	5784384138} support 103 92 101 104 400 400	, 'clfgamma': 'scale'
accur macro weighted6. Random {'clf_bc	o 1 2 3 cacy avg avg avg cotstrap	'linear', recision 1.00 0.94 0.95 0.98 0.97 0.97	'clfnu' recall 0.98 0.99 0.94 0.96 0.97 0.97 () 2', 'clfcri	: 0.0707948 f1-score 0.99 0.96 0.95 0.97 0.97 0.97	5784384138} support 103 92 101 104 400 400 400	, 'clfgamma': 'scale'

1 2 3	0.89 0.91 0.98	0.92 0.92 0.94	0.90 0.92 0.96	92 101 104	
accuracy			0.94	400	
macro avg	0.93	0.93	0.93	400	
weighted avg	0.94	0.94	0.94	400	

7. AdaBoostClassifier()

{'clf_algorithm': 'SAMME', 'clf_base_estimator': RandomForestClassifier(),

'clf__n_estimators': 50}

	precision	recall	f1-score	support
0	0.96	0.94	0.95	103
1	0.88	0.93	0.91	92
2	0.90	0.91	0.91	101
3	0.97	0.92	0.95	104
accuracy			0.93	400
macro avg	0.93	0.93	0.93	400
weighted avg	0.93	0.93	0.93	400

8. GradientBoostingClassifier()

{'clf criterion': 'squared error'. 'clf learning rate': 0.31622776601683794

f cii_ciirei	1011 . squar	ed_error	, ciiiea.	rming_rai	Le . 0.31022//0001003/94,
'clfloss':	'deviance',	'clfmax	_features':	'log2',	<pre>'clfn_estimators': 500}</pre>
	precision	recall	f1-score	support	
0	0.97	0.95	0.96	103	
1	0.86	0.95	0.90	92	
2	0.92	0.86	0.89	101	
3	0.95	0.94	0.95	104	
accuracy			0.93	400	
macro avg	0.92	0.93	0.92	400	
weighted avg	0.93	0.93	0.93	400	

^{9.} SGDClassifier()

{'clfalpha'	: 0.003290344	562312667	5, 'clfl	earning_rate':	'optimal',
<pre>'clfloss':</pre>	'squared_hing	ge', 'clf_	_penalty':	'11'}	
	precision	recall	f1-score	support	
0	1.00	0.98	0.99	103	
1	0.77	0.55	0.65	92	
2	0.66	0.84	0.74	101	
3	0.97	0.97	0.97	104	
accuracy			0.84	400	
macro avg	0.85	0.84	0.84	400	
weighted avg	0.85	0.84	0.84	400	

10. GaussianNB()

{'clf var smoothing': 1e-20}

·	00000000			
	precision	recall	f1-score	support
0	0.95	0.94	0.95	103
1	0.70	0.71	0.70	92
2	0.67	0.64	0.66	101
3	0.87	0.90	0.89	104
			0.00	400
accuracy			0.80	400
macro avg	0.80	0.80	0.80	400
weighted avg	0.80	0.80	0.80	400

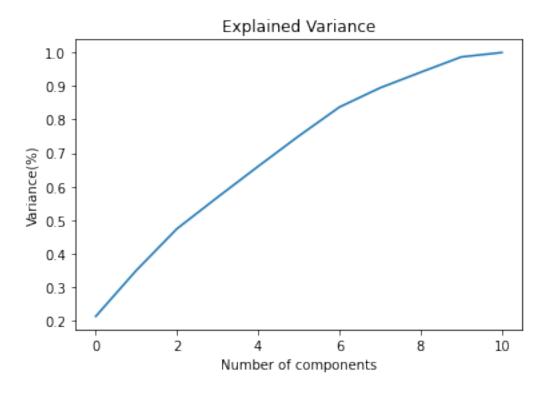
11. MultinomialNB()

II. MULTINOMIAIND()

{'clf_alpha': 0, 'clf_fit_prior': False} precision recall f1-score support 0 0.82 0.78 0.80 103 1 0.41 0.40 0.41 92 2 0.29 0.22 0.25 101 0.49 0.63 0.55 104 accuracy 0.51 400 macro avg 0.50 0.51 0.50 400 weighted avg 0.50 0.51 0.50 400

1.5.2 On df6(After VIF)

```
[15]: pca = PCA()
      principalComp = pca.fit_transform(df6)
      plt.figure()
      plt.plot(np.cumsum(pca.explained_variance_ratio_))
      plt.xlabel('Number of components')
      plt.ylabel('Variance(%)')
      plt.title('Explained Variance')
      plt.show()
      n_components=10
      pca = PCA(n_components=n_components)
      new_data = pca.fit_transform(df6)
      df7 = pd.DataFrame(new_data,columns = [f'column {i}' for i in_
      →range(n_components)])
      df7
      X_train_ecvp, X_test_ecvp, y_train_ecvp, y_test_ecvp =
      -train_test_split(df7,df2,random_state=5,shuffle=True,test_size=0.2)
      Classification(X_train_ecvp, X_test_ecvp, y_train_ecvp, y_test_ecvp.values.
       →ravel(),data,'PCA_e+c+v')
      print('WITH GridSearch'+'\n'+"-"*80+'\n'+"-"*80 )
      Classification(X_train_ecvp, X_test_ecvp, y_train_ecvp, y_test_ecvp.values.
       →ravel(),data_GS,'PCA_e+c+v_GS',Gridsearch=True)
```



1.	KNeighborsClassifier()	١

V			C 4	
	precision	recall	f1-score	support
0	0.86	0.81	0.83	103
1	0.56	0.70	0.62	92
2	0.60	0.58	0.59	101
3	0.85	0.74	0.79	104
accuracy			0.71	400
macro avg	0.72	0.71	0.71	400
weighted avg	0.72	0.71	0.71	400

2. LogisticRegression()

U		precision	recall	f1-score	support
	0	0.93	0.96	0.94	103
	1	0.83	0.82	0.82	92
	2	0.83	0.83	0.83	101
	3	0.94	0.92	0.93	104
accura	acy			0.89	400
macro a	avg	0.88	0.88	0.88	400
weighted a	avg	0.88	0.89	0.88	400

3. DecisionTreeClassifier()

	precision	recall	f1-score	support
0	0.89	0.86	0.88	103
1	0.67	0.77	0.72	92
2	0.70	0.55	0.62	101
3	0.79	0.87	0.83	104

accur	-	0.76	0.76	0.77	400 400	
	_	0.77			400	
4. SVC()						
{}						
		precision	recall	f1-score	support	
	0			0.92	103	
	1			0.79		
	2			0.81		
	3	0.94	0.89	0.92	104	
accur	cacy			0.86	400	
macro	-	0.86	0.86	0.86	400	
weighted	avg	0.87	0.86	0.86	400	
5. NuSVC(()					
{}						
		precision	recall	f1-score	support	
	0	0.95	0.94	0.95	103	
	1	0.84	0.82	0.83	92	
	2	0.78	0.81	0.80	101	
	3	0.89	0.89	0.89	104	
accur	racv			0.87	400	
	-	0.87	0.87		400	
weighted	_	0.87				
6. Random	 nFore	stClassifier	 ()			
6. Random	 nFore	stClassifier	 ()			
	 nFore	stClassifier		f1-score	support	
	 nFore				support	
		precision	recall 0.93	0.92		

0	0.70	0.71	0.75	101	
2					
3	0.90	0.91	0.91	104	
accuracy			0.84	400	
macro avg	0.83	0.84	0.83	400	
weighted avg	0.84	0.84	0.84	400	
7. AdaBoostCl					
7. Adaboosto	lassifier()				
{}		7.7	C.4		
	precision	recall	il-score	support	
0			0.83	103	
1			0.64	92	
2	0.58	0.73	0.65	101	
3	0.98	0.51	0.67	104	
accuracy			0.69	400	
macro avg	0.77	0.69	0.70	400	
weighted avg				400	
0					
O Cmodian+Da		fica()			
8. GradientBo	ostingciassi	iler()			
{}					
	precision	recall	f1-score	support	
0	0.93	0.89	0.91	103	
1	0.70	0.78	0.74	92	
2	0.74	0.73	0.74	101	
3	0.93	0.88	0.90	104	
accuracy			0.82	400	
macro avg	0.82	0.82	0.82	400	
weighted avg	0.83	0.82	0.82	400	
Merkinen and	0.05	0.02	0.02	400	
9. SGDClassif	fier()				

precision recall f1-score support

0	0.92	0.97	0.94	103	
1			0.45	92	
2			0.60	101	
3			0.94	104	
accuracy			0.75	400	
macro avg	0.74	0.74	0.73	400	
weighted avg	0.74	0.75	0.74	400	
		·			
 10. GaussianN	 IB()				
{}		11	£4		
	precision	recall	II-score	support	
0	0.90	0.92	0.91	103	
1	0.66		0.66	92	
2	0.57	0.61		101	
3				104	
accuracy			0.73	400	
macro avg	0.73	0.72	0.73	400	
weighted avg	0.73	0.73	0.73	400	
 11. Multinomi	.alNB()				
C)	precision	recall	f1-score	support	
0	0.94	0.59	0.73	103	
1	0.44		0.58	92	
2	0.53		0.47	101	
3	0.84			104	
J	0.01	0.01	0.10	101	
accuracy			0.62	400	
macro avg	0.69	0.62		400	
weighted avg				400	
5					
WITH GridSear	ch				

1. KNeighborsClassifier()

{'clf_algorithm': 'auto', 'clf_metric': 'cityblock', 'clf_n_neighbors': 14,
'clf_weights': 'distance'}

-	precision	recall	f1-score	support
0	0.88	0.88	0.88	103
1	0.67	0.72	0.69	92
2	0.67	0.63	0.65	101
3	0.82	0.81	0.82	104
accuracy			0.76	400
macro avg	0.76	0.76	0.76	400
weighted avg	0.76	0.76	0.76	400

2. LogisticRegression()

{'clfC': 0.	0117210229753	348, 'cli	fpenalty'	: 'none',	<pre>'clfsolver':</pre>	'saga'}
	precision	recall	f1-score	support		
0	0.93	0.96	0.94	103		
1	0.83	0.82	0.82	92		
2	0.84	0.83	0.84	101		
3	0.94	0.93	0.94	104		
accuracy			0.89	400		
macro avg	0.89	0.89	0.89	400		
weighted avg	0.89	0.89	0.89	400		

3. DecisionTreeClassifier()

{'clf_ccp_alpha': 0.001623776739188721, 'clf_criterion': 'gini',

^{&#}x27;clf__min_samples_split': 2, 'clf__splitter': 'best'}

	precision	recall	fl-score	support
_	0.07	0.07	0.07	100
0	0.87	0.87	0.87	103
1	0.64	0.70	0.67	92
2	0.69	0.58	0.63	101
3	0.81	0.88	0.84	104

^{&#}x27;clf__max_features': None, 'clf__max_leaf_nodes': None,

accuracy			0.76	400	
macro avg	0.76	0.76	0.75	400	
weighted avg	0.76	0.76	0.76	400	
4. SVC()					
		 ision_fun	ction_shape	e': 'ovo', 'o	clfdegree': 3,
'clfkernel'	<pre>: 'linear'} precision</pre>	recall	f1-score	support.	
	precibion	rccarr	II bcorc	Suppor 0	
0	0.92	0.95	0.93	103	
1	0.81	0.80	0.81	92	
2	0.82	0.82	0.82	101	
3	0.94	0.91	0.93	104	
accuracy			0.88	400	
macro avg	0.87	0.87	0.87	400	
weighted avg	0.87	0.88	0.87	400	
5. NuSVC()					
					, 'clfgamma': 'scale'
'clfkernel'	precision				
	precision	recall	11-score	support	
0	0 94	0.88	0 91	103	
0				103 92	
1	0.73	0.85	0.78	92	
1 2 3	0.73 0.77	0.85 0.73	0.78 0.75 0.88	92 101	
1 2 3 accuracy	0.73 0.77 0.90	0.85 0.73	0.78 0.75	92 101 104	
1 2 3 accuracy macro avg	0.73 0.77	0.85 0.73 0.87	0.78 0.75 0.88	92 101 104 400	
1 2 3 accuracy macro avg	0.73 0.77 0.90	0.85 0.73 0.87	0.78 0.75 0.88 0.83	92 101 104 400 400	
1 2 3 accuracy	0.73 0.77 0.90	0.85 0.73 0.87	0.78 0.75 0.88 0.83	92 101 104 400 400	

^{6.} RandomForestClassifier()

^{{&#}x27;clf_bootstrap': True, 'clf_criterion': 'entropy', 'clf_max_depth': None, 'clf_max_features': 'sqrt', 'clf_n_estimators': 200}

	precision	recall	f1-score	support
0	0.92	0.93	0.93	103
1	0.76	0.78	0.77	92
2	0.78	0.74	0.76	101
3	0.90	0.91	0.91	104
accuracy			0.84	400
macro avg	0.84	0.84	0.84	400
weighted avg	0.84	0.84	0.84	400

7. AdaBoostClassifier()

{'clf_algorithm': 'SAMME.R', 'clf_base_estimator': RandomForestClassifier(), 'clf__n_estimators': 100}

	precision	recall	f1-score	support
(0 0.92	0.95	0.93	103
:	1 0.78	0.80	0.79	92
:	2 0.79	0.71	0.75	101
;	3 0.88	0.90	0.89	104
accurac	у		0.84	400
macro av	g 0.84	0.84	0.84	400
weighted av	g 0.84	0.84	0.84	400

8. GradientBoostingClassifier()

{'clf__criterion': 'friedman_mse', 'clf__learning_rate': 0.1, 'clf__loss': 'deviance', 'clf__max_features': 'log2', 'clf__n_estimators': 900}

	precision	recall	f1-score	support
0	0.91	0.91	0.91	103
1	0.70	0.77	0.74	92
2	0.75	0.71	0.73	101
3	0.92	0.88	0.90	104
accuracy			0.82	400
macro avg	0.82	0.82	0.82	400
weighted avg	0.83	0.82	0.82	400

9. SGDClassifier()

{'clf_alpha': 0.0009236708571873865, 'clf_learning_rate': 'optimal',

'clf loss' 'log' 'clf nenalty' 'll'}

.cli_loss.:	'log', 'CII_	_benaith.	: .TI.}	
	precision	recall	f1-score	support
0	0.93	0.96	0.95	103
1	0.63	0.62	0.63	92
2	0.66	0.66	0.66	101
3	0.94	0.93	0.94	104
accuracy			0.80	400
macro avg	0.79	0.79	0.79	400
weighted avg	0.80	0.80	0.80	400
3				

10. GaussianNB()

{'clf	var	smoothing'	:	1e-20}
-------	-----	------------	---	--------

	precision	recall	f1-score	support
0 1 2	0.90 0.66 0.57	0.92 0.65 0.61	0.91 0.66 0.59	103 92 101
3	0.79	0.71	0.75	104
accuracy macro avg weighted avg	0.73 0.73	0.72 0.73	0.73 0.73 0.73	400 400 400

11. MultinomialNB()

	: False}	fit_prior'	: 0, 'clf:	{'clfalpha'
support	f1-score	recall	precision	
103	0.80	0.76	0.86	0
92	0.55	0.59	0.51	1
101	0.55	0.56	0.54	2
104	0.75	0.73	0.77	3
400	0.66			accuracy

```
0.67
                             0.66
                                       0.66
                                                   400
  macro avg
weighted avg
                   0.68
                             0.66
                                       0.67
                                                   400
```

1.6 Reverse Feature Elimination

```
[17]: X_train_e, X_test_e, y_train_e, y_test_e = __
       →train_test_split(df4,df2,random_state=5,shuffle=True,test_size=0.2)
[18]: rfe = RFE(GradientBoostingClassifier(random_state=42),
                n features to select=10)
```

```
rfe.fit(X_train_e, y_train_e.values.ravel())
rfe_features = X_train_e.columns[rfe.support_]
dfr = df4[rfe_features]
print(f"===== {len(rfe_features)} features were selected =====")
print(f"{', '.join(rfe_features)}")
X_train_er, X_test_er, y_train_er, y_test_er =
-train_test_split(dfr,df2,random_state=5,shuffle=True,test_size=0.2)
Classification(X_train_er, X_test_er, y_train_er, y_test_er,data,'RFE_e')
print('WITH GridSearch'+'\n'+"-"*80+'\n'+"-"*80 )
Classification(X_train_er, X_test_er, y_train_er,_
```

===== 10 features were selected ===== battery_power, mobile_wt, px_height, px_width, ram, px_diag, px_area, talk_per_mAh, ram_per_core, weight_by_thickness

KNeighborsClassifier()

	precision	recall	f1-score	support
0 1 2 3	0.43 0.24 0.30 0.30	0.50 0.33 0.28 0.17	0.47 0.28 0.29 0.22	103 92 101 104
accuracy macro avg weighted avg	0.32 0.32	0.32 0.32	0.32 0.31 0.31	400 400 400

^{2.} LogisticRegression()

{ }				
	precision	recall	f1-score	support
0	0.89	0.85	0.87	103
1	0.46	0.49	0.48	92
2	0.29	0.24	0.26	101
3	0.60	0.69	0.64	104
accuracy			0.57	400
macro avg	0.56	0.57	0.56	400
weighted avg	0.56	0.57	0.57	400

3. DecisionTreeClassifier()

{}				
	precision	recall	f1-score	support
0	0.94	0.92	0.93	103
1	0.84	0.87	0.86	92
2	0.83	0.84	0.84	101
3	0.91	0.89	0.90	104
accuracy			0.88	400
macro avg	0.88	0.88	0.88	400
weighted avg	0.88	0.88	0.88	400
_				

4. SVC()

	precision	recall	f1-score	support
0	0.98	0.96	0.97	103
1	0.85	0.92	0.89	92
2	0.88	0.84	0.86	101
3	0.95	0.93	0.94	104
accuracy			0.92	400
macro avg	0.91	0.91	0.91	400
weighted avg	0.92	0.92	0.92	400

-	MQUQ()			

5.	NuSV	\sim $_{l}$	(٦
ο.	Nubv	U,	`	,

{}

	precision	recall	f1-score	support
0	0.96	0.95	0.96	103
1	0.84	0.89	0.86	92
2	0.87	0.84	0.85	101

3	0.95	0.93	0.94	104
			0.01	400
accuracy			0.91	400
macro avg	0.90	0.90	0.90	400
weighted avg	0.91	0.91	0.91	400

6. RandomForestClassifier()

IJ		precision	recall	f1-score	support
		•			••
	0	0.99	0.96	0.98	103
	1	0.86	0.90	0.88	92
	2	0.84	0.89	0.87	101
	3	0.98	0.91	0.95	104
accur	cacy			0.92	400
macro	avg	0.92	0.92	0.92	400
weighted	avg	0.92	0.92	0.92	400

7. AdaBoostClassifier()

{}				
	precision	recall	f1-score	support
0	0.92	0.55	0.69	103
1	0.58	0.80	0.67	92
2	0.63	0.85	0.73	101
3	0.91	0.64	0.75	104
accuracy			0.71	400
macro avg	0.76	0.71	0.71	400

weighted avg	0.76	0.71	0.71	400	
8. GradientBo	ostingClassi	 fier()			
{}					
	precision	recall	f1-score	support	
0	0.97	0.96	0.97	103	
1	0.84	0.93	0.89	92	
2	0.90	0.85	0.87	101	
3	0.97	0.93	0.95	104	
accuracy			0.92	400	
macro avg	0.92	0.92		400	
weighted avg			0.92	400	
9. SGDClassif	 ier()				
{}					
	precision	recall	f1-score	support	
0	0.95	0.99	0.97	103	
1	0.54	0.48	0.51	92	
2	0.55	0.59	0.57	101	
3	0.97	0.94	0.96	104	
accuracy			0.76	400	
•	0.75	0.75		400	
weighted avg					
10. GaussianN	IB()				
{}					
C	precision	recall	f1-score	support	
0	0.92	0.95	0.94	103	
1	0.68		0.68	92	
2			0.68	101	
3	0.93	0.85	0.88	104	
0	0.00	0.00	0.00	101	

accuracy			0.80	400	
•	0.80	0.79		400	
_	0.80			400	
 11. Multinomi	ialNB()				
 [}					
	precision	recall	f1-score	support	
0	0.85	0.70	0.77	103	
1	0.40	0.46	0.42	92	
2	0.45	0.50	0.47	101	
3	0.68	0.64	0.66	104	
accuracy			0.58	400	
macro avg	0.59	0.57	0.58	400	
weighted avg	0.60	0.58	0.59	400	
 1. KNeighbors	 sClassifier()				
			etric': 'c	osine', 'clf_	_n_neighbors': 9,
cliweights	s': 'distance precision		f1-score	support	
0	0.92	0.83	0.87	103	
1	0.55	0.51	0.53	92	
2	0.44	0.49	0.46	101	
3	0.63	0.68	0.66	104	
accuracy			0.63	400	
macro avg	0.64	0.63	0.63	400	
veighted avg	0.64	0.63	0.63	400	
			·		
2. LogisticRe	egression()				

{'clf_C': 0.024920211513780568, 'clf_penalty': 'l2', 'clf_solver': 'newtoncg'}

-	precision	recall	f1-score	support
0	0.91	0.92	0.92	103
1	0.71	0.65	0.68	92
2	0.65	0.67	0.66	101
3	0.86	0.88	0.87	104
accuracy			0.79	400
macro avg	0.78	0.78	0.78	400
weighted avg	0.78	0.79	0.78	400

3. DecisionTreeClassifier()

{'clf__ccp_alpha': 0.003359818286283781, 'clf__criterion': 'entropy',

	precision	recall	II-score	support
0	0.94	0.94	0.94	103
1	0.79	0.91	0.85	92
2	0.84	0.76	0.80	101
3	0.92	0.88	0.90	104
accuracy			0.87	400
macro avg	0.87	0.87	0.87	400
weighted avg	0.88	0.87	0.87	400

4. SVC()

'clf__kernel': 'linear'}

	precision	recall	f1-score	support	
0	1.00	0.99	1.00	103	
1	0.94	0.99	0.96	92	
2	0.96	0.94	0.95	101	
3	0.99	0.97	0.98	104	
accuracy			0.97	400	
macro avg	0.97	0.97	0.97	400	

⁻⁻⁻⁻⁻

^{&#}x27;clf__max_features': None, 'clf__max_leaf_nodes': None,

weighted avg	0.	97 0.	97 0.9	7 400	
5. NuSVC()			_		
		ion_shape'	- : 'ovo', 'c	lfdegree'	: 3, 'clfgamma': 'scale',
'clfkernel	': 'linea	r', 'clf	nu': 0.0676	08297539198	18}
	precisi	on reca	ll f1-scor	e support	
0	1	00 0.9	99 1.0	0 103	
1		93 0.1			
2			93 0.9		
3			97 0.9		
accuracy				7 400	
macro avg			97 0.9		
weighted avg	0.	97 0.5	97 0.9	7 400	
<pre>{'clfboots 'clfmax_fe</pre>	atures':	None, 'clf		ors': 200}	'clfmax_depth': None,
0	0.	96 0.1	95 0.9	6 103	
1	0.	89 0.	95 0.9	2 92	
2	0.	91 0.	92 0.9	2 101	
3	0.	98 0.	92 0.9	5 104	
accuracy			0.9	4 400	
macro avg	0	93 0	94 0.9		
weighted avg			94 0.9		
7. AdaBoostC		·()	-		
<pre>{'clf_algor 'clf_n_esti</pre>			- clfbase_e	stimator':	RandomForestClassifier(),
211 - 11 - 62 01	precisi		ll f1-scor	e support	
0	1.	00 0.	95 0.9	8 103	

1	0.85	0.96	0.90	92	
2	0.88	0.88	0.88	101	
3	0.98	0.92	0.95	104	
accuracy			0.93	400	
macro avg	0.93	0.93	0.93	400	
weighted avg	0.93	0.93	0.93	400	
8. GradientBo		 fior()			
o. Gradientho	OSCINGCIASSI	rier()			
{'clf criter	ion': 'square	ed error'.	'clf lea	rning rate':	0.31622776601683794,
	_			_	f_n_estimators': 300}
0111000 .	precision		f1-score	~	IM_OSOIMAGOIS . GOO,
	Processi			z appoz c	
0	0.99	0.97	0.98	103	
1			0.90	92	
2		0.84		101	
3	0.95	0.90	0.93	104	
accuracy			0.91	400	
macro avg	0.91	0.91	0.91	400	
weighted avg	0.91	0.91	0.91	400	
9. SGDClassif	ier()				
{'clf_alpha'				_	': 'optimal',
'clfloss':					
	precision	recall :	f1-score	support	
0	1 00	0.00	0.00	100	
0	1.00	0.98	0.99	103	
1	0.69	0.65	0.67	92	
2	0.68	0.72	0.70	101	
3	0.97	0.98	0.98	104	
2663726			0 04	400	
accuracy	A 04	0 00	0.84 0.83	400	
macro avg	0.84	0.83	0.83	400	

0.84

400

0.84

0.84

weighted avg

^{10.} GaussianNB()

```
{'clf_var_smoothing': 1e-20}
             precision
                          recall f1-score
                                              support
           0
                  0.92
                             0.95
                                       0.94
                                                  103
           1
                  0.68
                             0.67
                                       0.68
                                                   92
           2
                  0.66
                             0.70
                                       0.68
                                                  101
                  0.93
                             0.85
                                       0.88
                                                  104
                                       0.80
                                                  400
   accuracy
                  0.80
                             0.79
                                       0.79
                                                  400
  macro avg
weighted avg
                  0.80
                             0.80
                                       0.80
                                                  400
11. MultinomialNB()
{'clf_alpha': 31.622776601683793, 'clf_fit_prior': True}
             precision recall f1-score support
           0
                  0.82
                             0.76
                                       0.79
                                                  103
                             0.46
           1
                  0.44
                                       0.45
                                                   92
           2
                  0.48
                             0.46
                                       0.47
                                                  101
           3
                  0.67
                           0.73
                                       0.70
                                                  104
                                       0.60
                                                  400
   accuracy
```

0.60

0.61

400

400

0.60

0.60

1.7 Reverse Feature Elimination with CV

0.60

0.61

macro avg

weighted avg

Classification(X_train_er, X_test_er, y_train_er,_ →y_test_er,data_GS,'RFECV_e_GS',Gridsearch=True)

===== 10 features were selected ===== battery_power, mobile_wt, px_height, px_width, ram, px_diag, px_area,

talk_per_mAh, ram_per_core, weight_by_thickness

1. KNeighborsClassifier()

	 	 	 	 	 	_
רז						

Ü	precision	recall	f1-score	support
0	0.43	0.50	0.47	103 92
1 2	0.24 0.30	0.33 0.28	0.28	101
3	0.30	0.17	0.22	104
accuracy			0.32	400
macro avg	0.32	0.32	0.31	400
weighted avg	0.32	0.32	0.31	400

2. LogisticRegression()

{}

	precision	recall	f1-score	support
0 1 2 3	0.89 0.46 0.29 0.60	0.85 0.49 0.24 0.69	0.87 0.48 0.26 0.64	103 92 101 104
accuracy macro avg weighted avg	0.56 0.56	0.57 0.57	0.57 0.56 0.57	400 400 400

3. DecisionTreeClassifier()

	precision	recall	f1-score	support
0	0.95	0.92	0.94	103
1	0.84	0.88	0.86	92

	0.00	0 04	0 04	404	
2			0.84	101	
3	0.91	0.89	0.90	104	
accuracy			0.89	400	
macro avg	0.88	0.88	0.88	400	
weighted avg	0.89	0.89	0.89	400	
4. SVC()					
{}					
1,5	precision	recall	f1-score	gunnort	
	precision	recarr	11-SCOIE	support	
0	0.98	0.96	0.97	103	
1			0.89		
2			0.86		
3			0.94		
3	0.95	0.95	0.94	104	
accuracy			0.92	400	
	0.91	0.91			
weighted avg					
8	0.02	0.02	0.02		
5. NuSVC()					
	nrecision	 	f1-score	support	
	precision	recall	f1-score	support	
	_		f1-score	support	
	_				
0 1	0.96 0.84	0.95 0.89	0.96 0.86	103 92	
{} 0 1 2	0.96 0.84 0.87	0.95 0.89 0.84	0.96 0.86 0.85	103 92 101	
	0.96 0.84	0.95 0.89 0.84	0.96 0.86	103 92	
	0.96 0.84 0.87	0.95 0.89 0.84	0.96 0.86 0.85	103 92 101	
0 1 2 3 accuracy	0.96 0.84 0.87 0.95	0.95 0.89 0.84 0.93	0.96 0.86 0.85 0.94	103 92 101 104	
{} 0 1 2 3 accuracy macro avg	0.96 0.84 0.87 0.95	0.95 0.89 0.84 0.93	0.96 0.86 0.85 0.94 0.91 0.90	103 92 101 104 400 400	
{} 0 1 2 3 accuracy macro avg	0.96 0.84 0.87 0.95	0.95 0.89 0.84 0.93	0.96 0.86 0.85 0.94	103 92 101 104	
{} 0 1 2 3 accuracy macro avg	0.96 0.84 0.87 0.95	0.95 0.89 0.84 0.93	0.96 0.86 0.85 0.94 0.91 0.90	103 92 101 104 400 400	
0 1 2 3 accuracy	0.96 0.84 0.87 0.95	0.95 0.89 0.84 0.93	0.96 0.86 0.85 0.94 0.91 0.90	103 92 101 104 400 400	
{} 0 1 2 3 accuracy macro avg weighted avg	0.96 0.84 0.87 0.95	0.95 0.89 0.84 0.93	0.96 0.86 0.85 0.94 0.91 0.90	103 92 101 104 400 400	
{} 0 1 2 3 accuracy macro avg	0.96 0.84 0.87 0.95	0.95 0.89 0.84 0.93	0.96 0.86 0.85 0.94 0.91 0.90	103 92 101 104 400 400	
accuracy macro avg weighted avg	0.96 0.84 0.87 0.95	0.95 0.89 0.84 0.93	0.96 0.86 0.85 0.94 0.91 0.90	103 92 101 104 400 400	
{} 0 1 2 3 accuracy macro avg weighted avg	0.96 0.84 0.87 0.95	0.95 0.89 0.84 0.93 0.90 0.91	0.96 0.86 0.85 0.94 0.91 0.90 0.91	103 92 101 104 400 400	

0 1 2 3	0.98 0.87 0.87 0.98	0.96 0.92 0.89 0.92	0.97 0.89 0.88 0.95	103 92 101 104
accuracy macro avg	0.93	0.92	0.93 0.92	400 400
weighted avg	0.93	0.93 	0.93 	400

7. AdaBoostClassifier()

{}

	precision	recall	f1-score	support
0	0.92	0.55	0.69	103
1	0.58	0.80	0.67	92
2	0.63	0.85	0.73	101
3	0.91	0.64	0.75	104
accuracy			0.71	400
macro avg	0.76	0.71	0.71	400
weighted avg	0.76	0.71	0.71	400

8. GradientBoostingClassifier()

	precision	recall	f1-score	support
0	0.97	0.96	0.97	103
1	0.84	0.93	0.89	92
2	0.90	0.85	0.87	101
3	0.97	0.93	0.95	104
accuracy			0.92	400
macro avg	0.92	0.92	0.92	400
weighted avg	0.92	0.92	0.92	400

^{9.} SGDClassifier()

{}				
	precision	recall	f1-score	support
0	0.94	0.98	0.96	103
1	0.45	0.53	0.49	92
2	0.52	0.45	0.48	101
3	0.98	0.91	0.95	104
accuracy			0.73	400
macro avg	0.72	0.72	0.72	400
weighted avg	0.73	0.72	0.73	400
10. Gaussian	NB()			
{}				
	precision	recall	f1-score	support
0	0.92	0.95	0.94	103
1	0.68	0.67	0.68	92
2	0.66	0.70	0.68	101
3	0.93	0.85	0.88	104
accuracy			0.80	400
•	0.80	0.70		400
<pre>macro avg weighted avg</pre>			0.79	400
11. Multinomi		_		
{}				
	precision	recall	f1-score	support
0	0.85	0.70	0.77	103
1	0.40	0.46	0.42	92
2	0.45	0.50	0.47	101
3	0.68	0.64	0.66	104

73

0.58

0.58

0.59

0.57

0.58

0.59

0.60

accuracy

macro avg

weighted avg

400

400

400

WITH GridSearch KNeighborsClassifier() 'clf__weights': 'distance'} precision recall f1-score support 0 0.92 0.83 0.87 103 1 0.55 0.51 0.53 92 2 0.44 0.49 0.46 101 0.68 0.63 0.66 104 accuracy 0.63 400 0.63 macro avg 0.64 0.63 400 weighted avg 0.64 0.63 0.63 400 2. LogisticRegression() {'clf_C': 0.024920211513780568, 'clf_penalty': 'l2', 'clf_solver': 'newtoncg'} precision recall f1-score support 0 0.91 0.92 0.92 103 1 0.71 0.65 0.68 92 2 0.65 0.67 0.66 101 3 0.86 0.88 0.87 104 0.79 400 accuracy 0.78 macro avg 0.78 0.78 400 weighted avg 0.78 0.78 0.79 400 3. DecisionTreeClassifier() {'clf__ccp_alpha': 0.0012742749857031334, 'clf__criterion': 'entropy',

^{&#}x27;clf__max_features': None, 'clf__max_leaf_nodes': None,

0	0.95	0.94	0.95	103	
1	0.83	0.93	0.88	92	
2	0.88	0.80	0.84	101	
3	0.92	0.90	0.91	104	
accuracy			0.90	400	
macro avg	0.89	0.90		400	
weighted avg				400	
4. SVC()					
{'clfC': 30), 'clfdeci	 sion_func	tion_shape	': 'ovo',	'clfdegree': 3,
'clfkernel'	: 'linear'}				
	precision	recall	f1-score	support	
0	1.00	0.99	1.00	103	
1	0.94	0.99	0.96	92	
2	0.96	0.94	0.95	101	
3	0.99	0.97	0.98	104	
accuracy			0.97	400	
macro avg	0.97	0.97	0.97	400	
weighted avg	0.97	0.97	0.97	400	
5. NuSVC()					
		_		_	3, 'clfgamma': 'scale',
clf_kernel'					8}
	precision	recall	f1-score	support	
0	1.00	0.99	1.00	103	
1	0.93	0.99	0.96	92	
2	0.96	0.93	0.94	101	
3	0.99	0.97	0.98	104	
accuracy			0.97	400	
macro avg	0.97	0.97	0.97	400	
weighted avg	0.97	0.97	0.97	400	

6. RandomForestClassifier()

{'clfbootstrap': True,	<pre>'clfcriterion':</pre>	'entropy', 'cl	fmax_depth': None,

<pre>'clfmax_fea</pre>	tures': 'sqrt	', 'clf	n_estimato	rs': 150}	
	precision	recall	f1-score	support	
0	0.98	0.96	0.97	103	
1	0.87	0.91	0.89	92	
2	0.89	0.90	0.90	101	
3	0.99	0.95	0.97	104	
accuracy			0.93	400	
macro avg	0.93	0.93	0.93	400	
weighted avg	0.93	0.93	0.93	400	

7. AdaBoostClassifier()

{'clf__algorithm': 'SAMME.R', 'clf__base_estimator': RandomForestClassifier(),
'clf__n_estimators': 100}

	precision	recall	f1-score	support	
0	0.99	0.96	0.98	103	
1	0.85	0.90	0.87	92	
2	0.84	0.87	0.85	101	
3	0.98	0.91	0.95	104	
accuracy			0.91	400	
macro avg	0.91	0.91	0.91	400	
weighted avg	0.92	0.91	0.91	400	

8. GradientBoostingClassifier()

{'clf__criterion': 'friedman_mse', 'clf__learning_rate': 1.0, 'clf__loss':

^{&#}x27;deviance', 'clf__max_features': 'log2', 'clf__n_estimators': 700}

	precision	recall	f1-score	support
0	0.98	0.95	0.97	103
1	0.86	0.92	0.89	92
2	0.87	0.87	0.87	101
3	0.96	0.92	0.94	104
accuracy			0.92	400

macro avg	0.92	0.92	0.92	400	
weighted avg	0.92	0.92	0.92	400	
9. SGDClassif 	1er()				
_				earning_rate':	'optimal',
'clfloss':	precision			support.	
	procession	100011	11 20010	Support	
0	0.97	0.98	0.98	103	
1			0.69	92	
2			0.68	101	
3	0.97	0.99	0.98	104	
accuracy			0.84	400	
macro avg	0.83	0.83	0.83	400	
weighted avg	0.84	0.84	0.84	400	
 10. GaussianN	B()				
	B()				
 10. GaussianN {'clfvar_sm	B()		f1-score	support	
	B() oothing': 1e precision	 -20} recall			
 {'clfvar_sm	B() oothing': 1e precision 0.92	 -20} recall	f1-score 0.94 0.68	support 103 92	
	B() oothing': 1e precision 0.92 0.68	 -20} recall	0.94	103	
{'clfvar_sm 0 1	B() oothing': 1e precision 0.92 0.68	 recall 0.95 0.67	0.94 0.68	103 92	
{'clfvar_sm 0 1 2 3	B() oothing': 1e precision 0.92 0.68 0.66	 -20} recall 0.95 0.67 0.70	0.94 0.68 0.68 0.88	103 92 101 104	
{'clfvar_sm 0 1 2 3 accuracy	B() oothing': 1e precision 0.92 0.68 0.66 0.93	 -20} recall 0.95 0.67 0.70 0.85	0.94 0.68 0.68 0.88	103 92 101 104	
{'clfvar_sm 0 1 2 3 accuracy macro avg	B() oothing': 1e precision 0.92 0.68 0.66 0.93	 -20} recall 0.95 0.67 0.70 0.85	0.94 0.68 0.68 0.88 0.80 0.79	103 92 101 104 400 400	
{'clfvar_sm 0 1 2 3 accuracy macro avg	B() oothing': 1e precision 0.92 0.68 0.66 0.93	 -20} recall 0.95 0.67 0.70 0.85	0.94 0.68 0.68 0.88 0.80 0.79	103 92 101 104 400 400	
{'clfvar_sm 0 1 2 3 accuracy macro avg weighted avg	B() oothing': 1e precision 0.92 0.68 0.66 0.93 0.80 0.80	 -20} recall 0.95 0.67 0.70 0.85	0.94 0.68 0.68 0.88 0.80 0.79	103 92 101 104 400 400	
{'clfvar_sm 0 1 2 3 accuracy macro avg weighted avg	B() oothing': 1e precision 0.92 0.68 0.66 0.93 0.80 0.80	 -20} recall 0.95 0.67 0.70 0.85	0.94 0.68 0.68 0.88 0.80 0.79	103 92 101 104 400 400	
{'clfvar_sm 0 1 2 3 accuracy macro avg weighted avg	B() oothing': 1e precision 0.92 0.68 0.66 0.93 0.80 0.80		0.94 0.68 0.68 0.88 0.80 0.79 0.80	103 92 101 104 400 400	
{'clfvar_sm 0 1 2 3 accuracy macro avg weighted avg	B() oothing': 1e precision 0.92 0.68 0.66 0.93 0.80 0.80	 -20} recall 0.95 0.67 0.70 0.85 0.79 0.80	0.94 0.68 0.68 0.88 0.80 0.79 0.80	103 92 101 104 400 400 400	
{'clfvar_sm 0 1 2 3 accuracy macro avg weighted avg	B()	 -20} recall 0.95 0.67 0.70 0.85 0.79 0.80 01683793, recall	0.94 0.68 0.68 0.88 0.80 0.79 0.80	103 92 101 104 400 400 400	

2	0.48	0.46	0.47	101
3	0.67	0.73	0.70	104
accuracy			0.60	400
macro avg	0.60	0.60	0.60	400
weighted avg	0.61	0.60	0.61	400

1.8 SelectKBest

```
[20]: kbest = SelectKBest(k=10)
     kbest.fit(X_train_e, y_train_e.values.ravel())
     # See selected features
     kbest_features = X_train_e.columns[kbest.get_support()]
     dfk = df4[kbest_features]
     print(f"===== {len(kbest_features)} features were selected =====")
     print(f"{', '.join(kbest_features)}")
     X_train_ek, X_test_ek, y_train_ek, y_test_ek =
      →train_test_split(dfk,df2,random_state=5,shuffle=True,test_size=0.2)
     Classification(X_train_ek, X_test_ek, y_train_ek, y_test_ek,data,'KBest_e')
     print('WITH GridSearch'+'\n'+"-"*80+'\n'+"-"*80 )
     Classification(X_train_ek, X_test_ek, y_train_ek,_
```

==== 10 features were selected ===== battery_power, px_height, px_width, ram, px_diag, px_area, talk_per_mAh, ram_per_core, sc_res, DPIy

1. KNeighborsClassifier()

n	
1}	

	precision	recall	f1-score	support
0 1 2 3	0.45 0.23 0.30 0.30	0.53 0.32 0.28 0.17	0.49 0.27 0.29 0.22	103 92 101 104
accuracy macro avg weighted avg	0.32 0.32	0.32 0.33	0.33 0.32 0.32	400 400 400

78

2. LogisticRegression()

ſι	

{}				
	precision	recall	f1-score	support
0	0.95	0.69	0.80	103
1	0.43	0.57	0.49	92
2	0.29	0.21	0.24	101
3	0.58	0.73	0.65	104
accuracy			0.55	400
macro avg	0.56	0.55	0.54	400
weighted avg	0.57	0.55	0.55	400

3. DecisionTreeClassifier()

{}	precision	recall	f1-score	support
0	0.94	0.93	0.94	103
1	0.89	0.86	0.87	92
2	0.84	0.88	0.86	101
3	0.91	0.90	0.91	104
accuracy			0.90	400
macro avg	0.90	0.89	0.89	400
weighted avg	0.90	0.90	0.90	400

4. SVC()

{}

G	precision	recall	f1-score	support
0	0.99	0.97	0.98	103
1	0.86	0.93	0.90	92
2	0.89	0.85	0.87	101
3	0.96	0.94	0.95	104
accuracy			0.93	400
macro avg	0.92	0.92	0.92	400
weighted avg	0.93	0.93	0.93	400

 {}						
	p	recision	recall	f1-score	support	
0		0.97	0.96	0.97	103	
1		0.86	0.90	0.88	92	
2		0.87	0.84	0.85	101	
3		0.94	0.93	0.94	104	
accuracy				0.91	400	
macro avg		0.91	0.91	0.91	400	
weighted avg		0.91	0.91	0.91	400	
	 		 ()			
	 est	 Classifier	()			
				f1-score	support	
	p:	recision	recall		support	
6. RandomFor ()	 p	recision 0.99	recall 0.96	0.98		
() 0 1 2		recision 0.99 0.87	recall 0.96	0.98 0.88	103 92 101	
(} 0 1		recision 0.99 0.87	recall 0.96 0.89	0.98 0.88	103 92	
() 0 1 2	p	0.99 0.87 0.85	recall 0.96 0.89 0.90	0.98 0.88 0.88 0.96	103 92 101	
(} 0 1 2 3 accuracy macro avg	p	0.99 0.87 0.85 0.98	recall 0.96 0.89 0.90 0.93	0.98 0.88 0.88 0.96 0.92	103 92 101 104	
(} 0 1 2 3 accuracy	p	0.99 0.87 0.85 0.98	recall 0.96 0.89 0.90 0.93	0.98 0.88 0.88 0.96 0.92	103 92 101 104 400	

precision recall f1-score support 0.29 103 0.83 0.43 0 0.83 1 0.46 0.59 92 2 0.67 0.71 0.69 101 0.86 0.75 0.80 104 0.64 400 accuracy

macro avg	0.70 0.71				
8. GradientBo	ostingClassi				
{}	precision	recall	f1-score	support	
0		0.96		103	
1		0.92			
2		0.82		101	
3	0.95	0.94	0.95	104	
accuracy			0.91	400	
•	0.91	0.91			
weighted avg	0.91	0.91	0.91	400	
9. SGDClassif					
9. SGDClassif		 recall	f1-score	support	
{}	precision				
{}	precision 0.97	0.99	0.98	103	
{} 0 1	precision 0.97 0.60	0.99 0.85	0.98 0.70	103 92	
{}	precision 0.97 0.60 0.77	0.99 0.85	0.98	103 92	
0 1 2 3	precision 0.97 0.60 0.77	0.99 0.85 0.47	0.98 0.70 0.58 0.97	103 92 101 104	
{} 0 1 2 3 accuracy	precision 0.97 0.60 0.77 0.97	0.99 0.85 0.47 0.97	0.98 0.70 0.58 0.97	103 92 101 104	
{} 0 1 2 3 accuracy macro avg	precision 0.97 0.60 0.77 0.97	0.99 0.85 0.47 0.97	0.98 0.70 0.58 0.97 0.82 0.81	103 92 101 104 400 400	
{} 0 1 2 3 accuracy	precision 0.97 0.60 0.77 0.97 0.83 0.84	0.99 0.85 0.47 0.97	0.98 0.70 0.58 0.97 0.82 0.81	103 92 101 104 400 400	
accuracy macro avg weighted avg	precision 0.97 0.60 0.77 0.97 0.83 0.84	0.99 0.85 0.47 0.97	0.98 0.70 0.58 0.97 0.82 0.81	103 92 101 104 400 400	
{} 0 1 2 3 accuracy macro avg weighted avg	precision 0.97 0.60 0.77 0.97 0.83 0.84	0.99 0.85 0.47 0.97	0.98 0.70 0.58 0.97 0.82 0.81 0.81	103 92 101 104 400 400 400	
accuracy macro avg weighted avg	precision 0.97 0.60 0.77 0.97 0.83 0.84	0.99 0.85 0.47 0.97 0.82 0.82	0.98 0.70 0.58 0.97 0.82 0.81 0.81	103 92 101 104 400 400 400	
accuracy macro avg weighted avg	precision 0.97 0.60 0.77 0.97 0.83 0.84 DB() precision 0.92	0.99 0.85 0.47 0.97	0.98 0.70 0.58 0.97 0.82 0.81 0.81	103 92 101 104 400 400 400	

3	0.91	0.83	0.87	104	
			0.77	400	
accuracy macro avg	0.77	0 77	0.77 0.77	400 400	
weighted avg	0.78		0.77	400	
0					
11. Multinomi	alNB()				
{}	precision	recall	f1-score	support	
0	0.82	0 71	0.76	103	
1			0.38	92	
2				101	
3	0.64	0.57	0.60	104	
accuracy			0.54	400	
macro avg	0.56	0.54		400	
weighted avg		0.54	0.55	400	
WITH GridSear	ch				
1. KNeighbors	 sClassifier()				
<pre>'clfalgori 'clfweights</pre>			etric': 'c	osine', 'cl	fn_neighbors': 9,
~ ~	precision		f1-score	support	
0	0.89	0.87	0.88	103	
1	0.58	0.54	0.56	92	
2	0.47	0.50	0.49	101	
3	0.67	0.67	0.67	104	
accuracy			0.65	400	
macro avg	0.65	0.65	0.65	400	
weighted avg	0.66	0.65	0.65	400	

^{2.} LogisticRegression()

	. 025929437974	 04667, 'c	lfpenalty	': '12',	'clfsolver': 'newton-
	precision	recall	f1-score	support	
0	0.94	0.90	0.92	103	
1	0.71	0.65	0.68	92	
2	0.56	0.59	0.58	101	
3	0.80	0.84	0.82	104	
accuracy			0.75	400	
macro avg	0.75	0.75	0.75	400	
weighted avg			0.75	400	
3. DecisionTr		 			
{'clfccp_a	lpha': 0.0011	288378916	846896, 'cl	fcriter	rion': 'entropy',
'clfmax_fea	atures': None	, 'clfm	ax_leaf_nod	les': None	θ,
'clfmin_sar	nples_split':	2, 'clf_	_splitter':	'random'	}
	precision	recall	f1-score	support	
	_				
0	0.94	0.93	0.94	103	
1	0.78	0.82	0.80	92	
2	0.81	0.80	0.81	101	
3	0.94	0.92	0.93	104	
accuracy			0.87	400	
macro avg	0.87	0.87		400	
weighted avg	0.87	0.87	0.87	400	
4. SVC()					
{'clf C': 2	. 'clf decis	ion funct	ion shape':	'ovo'. '	clfdegree': 3,
clf_kernel			<u></u>	,	,
	precision	recall	f1-score	support	
0	1.00	0.97	0.99	103	
1	0.91	1.00		92	
2			0.94	101	
3			0.98	104	

0.96

accuracy

400

macro avg 0.97 0.96 400 weighted avg 0.97 0.96 0.97 400 5. NuSVC() ('clfdecision_function_shape': 'ovo', 'clfdegree': 3, 'clfgamma': 'scale', 'clfkernel': 'linear', 'clfnu': 0.06760829753919818} precision recall f1-score support 0 1.00 0.98 0.99 103 1 0.91 0.99 0.95 92 2 0.95 0.92 0.93 101 3 0.99 0.96 0.98 104 accuracy 0.96 400 macro avg 0.96 0.96 0.96 400 weighted avg 0.96 0.96 0.96 400 ('clfbootstrap': True, 'clfcriterion': 'entropy', 'clfmax_depth': None, 'clfnex_features': None, 'clfnestimators': 200} precision recall f1-score support 0 0.96 0.94 0.95 103 1 0.86 0.95 0.90 92 2 0.94 0.88 0.91 101 3 0.96 0.95 0.96 104 accuracy 0.93 400 macro avg 0.93 0.93 0.93 400	weighted avg 5. NuSVC() {'clfdecisi	0.97					
5. NuSVC() ['clfdecision_function_shape': 'ovo', 'clfdegree': 3, 'clfgamma': 'scale', 'clfkernel': 'linear', 'clfnu': 0.06760829753919818}	5. NuSVC()						
5. NuSVC()	5. NuSVC()			. 			
5. NuSVC() {'clfdecision_function_shape': 'ovo', 'clfdegree': 3, 'clfgamma': 'scale', 'clfkernel': 'linear', 'clfnu': 0.06760829753919818}	5. NuSVC()						
5. NuSVC() {'clfdecision_function_shape': 'ovo', 'clfdegree': 3, 'clfgamma': 'scale', 'clfkernel': 'linear', 'clfnu': 0.06760829753919818}	5. NuSVC()						
{'clfdecision_function_shape': 'ovo', 'clfdegree': 3, 'clfgamma': 'scale', 'clfkernel': 'linear', 'clfnu': 0.06760829753919818}	{'clf_decisi						
'clfkernel': 'linear', 'clfnu': 0.06760829753919818}							
clfkernel': 'linear', 'clfnu': 0.06760829753919818} precision recall f1-score support							
precision recall f1-score support 0 1.00 0.98 0.99 103 1 0.91 0.99 0.95 92 2 0.95 0.92 0.93 101 3 0.99 0.96 0.98 104 accuracy 0.96 400 macro avg 0.96 0.96 0.96 400 weighted avg 0.96 0.96 0.96 400	'cit karnal'		_		_	, 'clfgamma':	'scale',
0 1.00 0.98 0.99 103 1 0.91 0.99 0.95 92 2 0.95 0.92 0.93 101 3 0.99 0.96 0.98 104 accuracy 0.96 400 macro avg 0.96 0.96 0.96 400 weighted avg 0.96 0.96 0.96 400 **Colf_bootstrap': True, 'clf_criterion': 'entropy', 'clf_max_depth': None, 'clf_max_features': None, 'clf_n_estimators': 200} precision recall f1-score support 0 0.96 0.94 0.95 103 1 0.86 0.95 0.90 92 2 0.94 0.88 0.91 101 3 0.96 0.95 0.96 104 accuracy 0.93 400 macro avg 0.93 0.93 0.93 400	clikerner						
1 0.91 0.99 0.95 92 2 0.95 0.92 0.93 101 3 0.99 0.96 0.98 104 accuracy 0.96 400 macro avg 0.96 0.96 0.96 400 weighted avg 0.96 0.96 0.96 400 ['clfbootstrap': True, 'clfcriterion': 'entropy', 'clfmax_depth': None, 'clfmax_features': None, 'clf_n_estimators': 200} precision recall f1-score support 0 0.96 0.94 0.95 103 1 0.86 0.95 0.90 92 2 0.94 0.88 0.91 101 3 0.96 0.95 0.96 104 accuracy 0.93 400 macro avg 0.93 0.93 0.93 400		precision	recall	il-score	support		
1 0.91 0.99 0.95 92 2 0.95 0.92 0.93 101 3 0.99 0.96 0.98 104 accuracy 0.96 400 macro avg 0.96 0.96 0.96 400 weighted avg 0.96 0.96 0.96 400 ['clfbootstrap': True, 'clfcriterion': 'entropy', 'clfmax_depth': None, 'clfmax_features': None, 'clf_n_estimators': 200} precision recall f1-score support 0 0.96 0.94 0.95 103 1 0.86 0.95 0.90 92 2 0.94 0.88 0.91 101 3 0.96 0.95 0.96 104 accuracy 0.93 400 macro avg 0.93 0.93 0.93 400	0	1 00	0 00	0.00	102		
2 0.95 0.92 0.93 101 3 0.99 0.96 0.98 104 accuracy 0.96 400 macro avg 0.96 0.96 0.96 400 weighted avg 0.96 0.96 0.96 400 [*Clf_bootstrap': True, 'clf_criterion': 'entropy', 'clf_max_depth': None, 'clf_max_features': None, 'clf_n_estimators': 200} precision recall f1-score support 0 0.96 0.94 0.95 103 1 0.86 0.95 0.90 92 2 0.94 0.88 0.91 101 3 0.96 0.95 0.96 104 accuracy 0.93 400 macro avg 0.93 0.93 0.93 400							
3 0.99 0.96 0.98 104 accuracy 0.96 400 macro avg 0.96 0.96 0.96 400 weighted avg 0.96 0.96 0.96 400 **Colf_bootstrap': True, 'clf_criterion': 'entropy', 'clf_max_depth': None, 'clf_max_features': None, 'clf_n_estimators': 200} precision recall f1-score support 0 0.96 0.94 0.95 103 1 0.86 0.95 0.90 92 2 0.94 0.88 0.91 101 3 0.96 0.95 0.96 104 accuracy 0.93 400 macro avg 0.93 0.93 0.93 400							
accuracy 0.96 400 macro avg 0.96 0.96 0.96 400 weighted avg 0.96 0.96 0.96 400							
macro avg 0.96 0.96 0.96 400 weighted avg 0.96 0.96 0.96 400 6. RandomForestClassifier() {'clfbootstrap': True, 'clfcriterion': 'entropy', 'clfmax_depth': None, 'clfmax_features': None, 'clfn_estimators': 200} precision recall f1-score support 0 0.96 0.94 0.95 103 1 0.86 0.95 0.90 92 2 0.94 0.88 0.91 101 3 0.96 0.95 0.96 104 accuracy macro avg 0.93 0.93 0.93 400	3	0.99	0.90	0.98	104		
macro avg 0.96 0.96 0.96 400 weighted avg 0.96 0.96 0.96 400 6. RandomForestClassifier() {'clfbootstrap': True, 'clfcriterion': 'entropy', 'clfmax_depth': None, 'clfmax_features': None, 'clfn_estimators': 200} precision recall f1-score support 0 0.96 0.94 0.95 103 1 0.86 0.95 0.90 92 2 0.94 0.88 0.91 101 3 0.96 0.95 0.96 104 accuracy macro avg 0.93 0.93 0.93 400	accuracu			0.96	400		
### weighted avg		0.06	0.06				
6. RandomForestClassifier()	_						
6. RandomForestClassifier() {'clfbootstrap': True, 'clfcriterion': 'entropy', 'clfmax_depth': None, 'clfmax_features': None, 'clfn_estimators': 200}	weighted avg	0.96	0.96	0.96	400		
precision recall f1-score support 0 0.96 0.94 0.95 103 1 0.86 0.95 0.90 92 2 0.94 0.88 0.91 101 3 0.96 0.95 0.96 104 accuracy macro avg 0.93 0.93 0.93 400						lfmax_depth':	None,
0 0.96 0.94 0.95 103 1 0.86 0.95 0.90 92 2 0.94 0.88 0.91 101 3 0.96 0.95 0.96 104 accuracy 0.93 400 macro avg 0.93 0.93 0.93 400	'clfmax_fea						
1 0.86 0.95 0.90 92 2 0.94 0.88 0.91 101 3 0.96 0.95 0.96 104 accuracy 0.93 400 macro avg 0.93 0.93 0.93 400		precision	recall	f1-score	support		
1 0.86 0.95 0.90 92 2 0.94 0.88 0.91 101 3 0.96 0.95 0.96 104 accuracy 0.93 400 macro avg 0.93 0.93 0.93 400	0	0.06	0 04	0.05	103		
2 0.94 0.88 0.91 101 3 0.96 0.95 0.96 104 accuracy 0.93 400 macro avg 0.93 0.93 0.93 400							
3 0.96 0.95 0.96 104 accuracy 0.93 400 macro avg 0.93 0.93 0.93 400							
accuracy 0.93 400 macro avg 0.93 0.93 0.93 400							
macro avg 0.93 0.93 0.93 400	3	0.90	0.95	0.90	104		
macro avg 0.93 0.93 0.93 400	accuracy			0.93	400		
		0.93	0.93				
	•						
	weighted avg	0.50	0.50	0.50	100		
7. AdaBoostClassifier()	7. AdaBoostCl	assifier()					
<pre>{'clf_algorithm': 'SAMME.R', 'clf_base_estimator': RandomForestClassifier(),</pre>	{'clf_algori	thm': 'SAMME	.R', 'clf_	_base_esti	mator': Ran	${\tt domForestClassif}$	ier(),
'clfn_estimators': 100}	clf_n_estim	nators': 100}					
precision recall f1-score support		precision	recall	f1-score	support		

0	0.99	0.95	0.97	103
1	0.86	0.89	0.88	92
2	0.85	0.90	0.88	101
3	0.98	0.93	0.96	104
accuracy macro avg weighted avg	0.92 0.92	0.92 0.92	0.92 0.92 0.92	400 400 400

8. GradientBoostingClassifier()

{'clf__criterion': 'squared_error', 'clf__learning_rate': 1.0, 'clf__loss':
'deviance', 'clf__max_features': 'log2', 'clf__n_estimators': 500}

·	precision	recall	f1-score	support
0	0.99 0.84	0.95 0.92	0.97 0.88	103 92
2	0.85	0.83	0.84	101 104
accuracy	0.54	0.31	0.91	400
macro avg	0.91 0.91	0.91 0.91	0.90	400 400

9. SGDClassifier()

{'clf_alpha': 0.01, 'clf_learning_rate': 'optimal', 'clf_loss':

	precision	recall	II-score	support	
0	0.99	0.99	0.99	103	
1	0.72	0.82	0.77	92	
2	0.77	0.68	0.72	101	
3	0.96	0.95	0.96	104	
accuracy			0.86	400	
macro avg	0.86	0.86	0.86	400	
weighted avg	0.86	0.86	0.86	400	

10. GaussianNB()

{'	clf	_var_	smoothing'	:	1e-20}
----	-----	-------	------------	---	--------

	precision	recall	f1-score	support
0 1 2 3	0.92 0.65 0.61 0.91	0.93 0.64 0.67 0.83	0.93 0.64 0.64 0.87	103 92 101 104
accuracy macro avg weighted avg	0.77	0.77	0.77 0.77 0.77	400 400 400

11. MultinomialNB()

{'clf_al	pha'	: 0.562341325	1903491,	'clffit_	prior': True}
		precision	recall	f1-score	support
	0	0.82	0.71	0.76	103
	1	0.36	0.41	0.38	92
	2	0.42	0.47	0.44	101
	3	0.64	0.57	0.60	104
accuracy				0.54	400
macro	avg	0.56	0.54	0.55	400
weighted	avg	0.57	0.54	0.55	400

```
[21]: pd.DataFrame(data).to_csv('data.csv')
```

[]:

pd.DataFrame(data_GS).to_csv('data_GS.csv')