```
import time
st_time = time.time()
```

```
# !pip·install·autoviz
```

```
from autoviz.AutoViz_Class import AutoViz_Class
%matplotlib inline
```

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import f1_score, confusion_matrix, ConfusionMatrixDisplay
from sklearn.preprocessing import StandardScaler ,RobustScaler
from sklearn.svm import SVC,LinearSVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.pipeline import Pipeline, make_pipeline
from sklearn.model_selection import train_test_split
```

```
trpath = '/content/drive/MyDrive/train.csv'
traindf = pd.read_csv(trpath)
```

```
traindf
```

	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mob

```
traindf.describe()
```

	battery_power	blue	clock_speed	dual_sim	fc	four_g	
count	2000.000000	2000.0000	2000.000000	2000.000000	2000.000000	2000.000000	2
mean	1238.518500	0.4950	1.522250	0.509500	4.309500	0.521500	
std	439.418206	0.5001	0.816004	0.500035	4.341444	0.499662	
min	501.000000	0.0000	0.500000	0.000000	0.000000	0.000000	
25%	851.750000	0.0000	0.700000	0.000000	1.000000	0.000000	
50%	1226.000000	0.0000	1.500000	1.000000	3.000000	1.000000	
75%	1615.250000	1.0000	2.200000	1.000000	7.000000	1.000000	
max	1998.000000	1.0000	3.000000	1.000000	19.000000	1.000000	

8 rows × 21 columns

```
traindf.isnull().sum()
```

```
battery_power    0
blue             0
clock_speed      0
dual_sim         0
fc               0
four_g           0
int_memory       0
m_dep            0
mobile_wt        0
n_cores          0
pc               0
px_height        0
px_width         0
ram              0
sc_h             0
sc_w             0
talk_time        0
three_g          0
touch_screen     0
wifi             0
price_range      0
dtype: int64
```

```
# AV = AutoViz_Class()
# piz = AV.AutoViz(
#     trpath,
#     sep=',',
```

```
#     chart_format = 'html', # 'server,'html,'png'
#     verbose = 2
# )
# pez = AV.AutoViz(
#     tspath,
#     sep=',',
#     chart_format = 'html', # 'server,'html,'png'
#     verbose = 2
# )

y = traindf['price_range']
y = pd.DataFrame(y)
y
```

	price_range
0	1
1	2
2	2
3	2
4	1
...	...
1995	0
1996	2
1997	3
1998	0
1999	3

2000 rows × 1 columns

```
x = traindf.drop(['price_range'],axis=1)
x
```

	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mob
0	842	0	2.2	0	1	0	7	0.6	
1	1021	1	0.5	1	0	1	53	0.7	
2	563	1	0.5	1	2	1	41	0.9	
3	615	1	2.5	0	0	0	10	0.8	
4	1821	1	1.2	0	13	1	44	0.6	

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2)
```

```
x_train.shape, x_test.shape, y_train.shape, y_test.shape
```

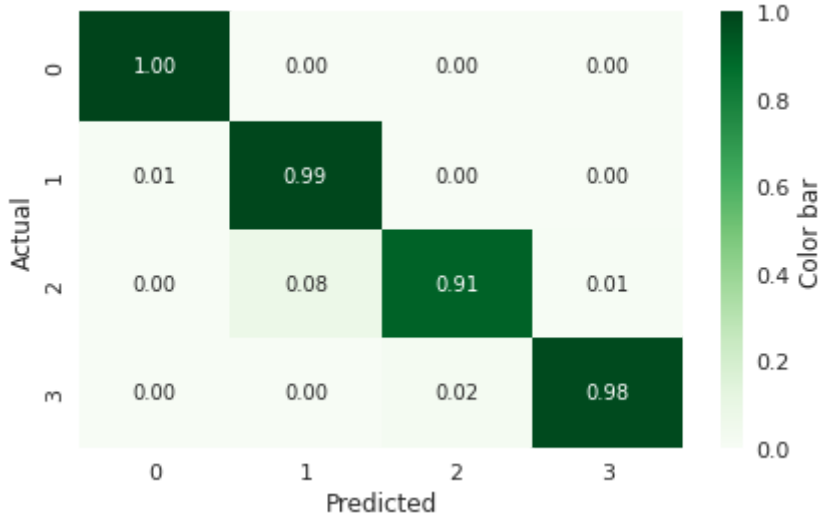
```
((1600, 20), (400, 20), (1600, 1), (400, 1))
```

```
def cr_scaler(my_scaler,my_x_train,my_x_test):
  my_scaler.fit_transform(my_x_train)
  my_scaler.fit_transform(my_x_test)
```

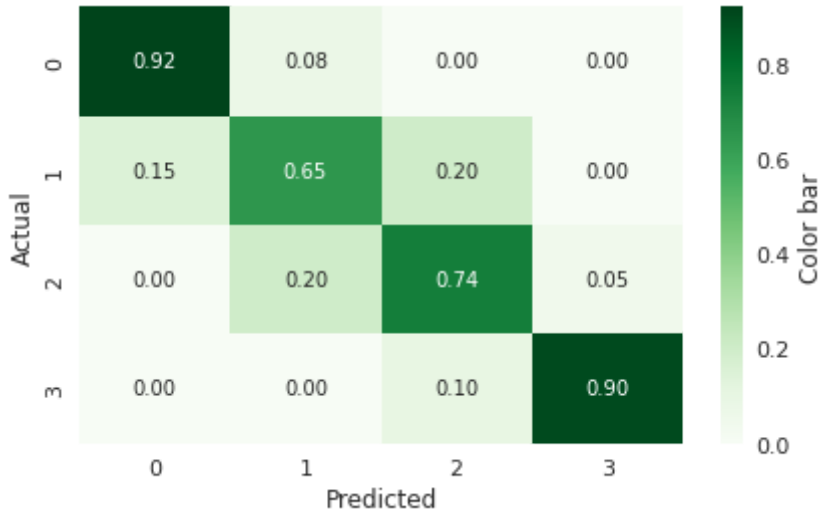
```
def my_modelfit(my_model,my_x_train,my_y_train):
  my_model.fit(my_x_train, my_y_train)


def my_predict(my_model,my_x_test):
  y_pred = my_model.predict(my_x_test)
  return y_pred


def my_f1_score(my_y_test,my_y_pred):
  f1 = f1_score(my_y_test, my_y_pred, average="micro")
  return f1


def my_conf_matrix(my_y_test,my_y_pred):
  cm = confusion_matrix(my_y_test, my_y_pred)
  cm_norm = np.round(cm/np.sum(cm,axis=1).reshape(-1,1),2)
  sns.heatmap(cm_norm,cmap='Greens',annot=True,
              cbar_kws={'orientation' : 'vertical','label' : 'Color bar'},
              fmt='.2f'
              )
  plt.xlabel('Predicted')
  plt.ylabel('Actual')
  plt.show()
  # cm_display = ConfusionMatrixDisplay(cm)
  # cm_display.plot()
  # plt.show()


clf = SVC(kernel= 'linear',C=5)
clf1 = GaussianNB()
clf2 = KNeighborsClassifier(n_neighbors=5)
```

```
clf3 = DecisionTreeClassifier()
clf4 = RandomForestClassifier()
clf5 = SVC(kernel= 'rbf',C=5)
clf5 = LinearSVC()
Sscaler = StandardScaler()
Rscaler = RobustScaler()

my_pipe = make_pipeline(cr_scaler(Sscaler, x_train, x_test), my_modelfit(clf, x_train, y_t
y_pred_res = my_predict(clf,x_test)
print(my_f1_score(y_test,y_pred_res))
my_conf_matrix(y_test,y_pred_res)

my_pipe1 = make_pipeline(cr_scaler(Sscaler, x_train, x_test), my_modelfit(clf1, x_train, y
y_pred_res1 = my_predict(clf1,x_test)
print(my_f1_score(y_test,y_pred_res1))
my_conf_matrix(y_test,y_pred_res1)

my_pipe2 = make_pipeline(cr_scaler(Sscaler, x_train, x_test), my_modelfit(clf2, x_train, y
y_pred_res2 = my_predict(clf2,x_test)
print(my_f1_score(y_test,y_pred_res2))
my_conf_matrix(y_test,y_pred_res2)

my_pipe3 = make_pipeline(cr_scaler(Sscaler, x_train, x_test), my_modelfit(clf3, x_train, y
y_pred_res3 = my_predict(clf3,x_test)
print(my_f1_score(y_test,y_pred_res3))
my_conf_matrix(y_test,y_pred_res3)

my_pipe4 = make_pipeline(cr_scaler(Sscaler, x_train, x_test), my_modelfit(clf4, x_train, y
y_pred_res4 = my_predict(clf4,x_test)
print(my_f1_score(y_test,y_pred_res4))
my_conf_matrix(y_test,y_pred_res4)

my_pipe5 = make_pipeline(cr_scaler(Sscaler, x_train, x_test), my_modelfit(clf5, x_train, y
y_pred_res5 = my_predict(clf5,x_test)
print(my_f1_score(y_test,y_pred_res5))
my_conf_matrix(y_test,y_pred_res5)
```

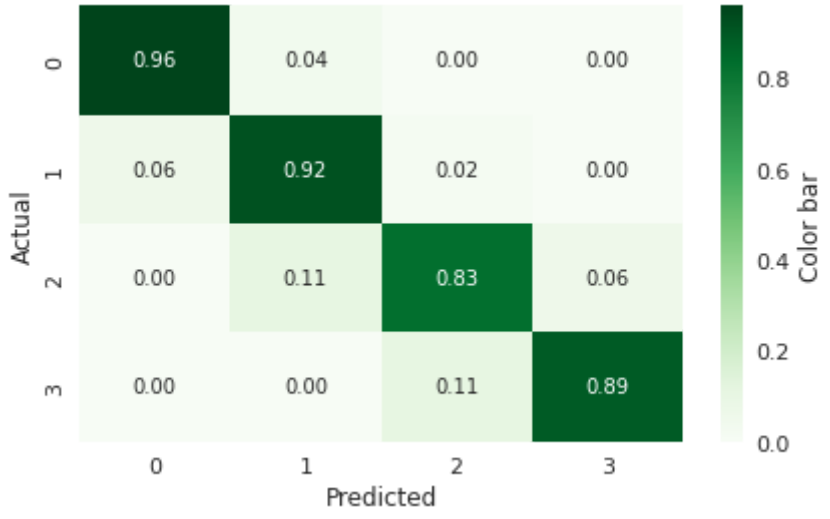
```
0.9725
```



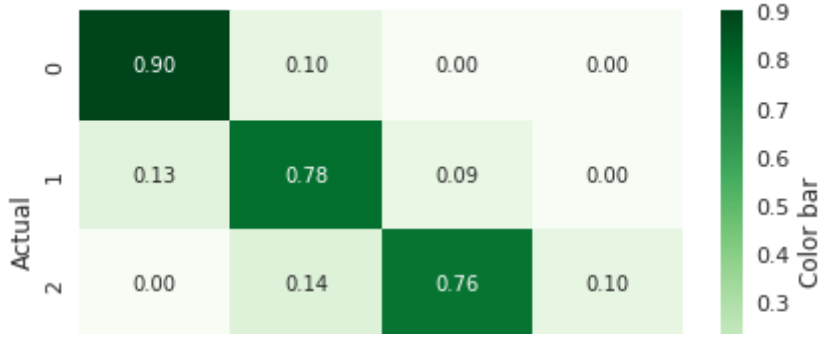

```
0.8075
```

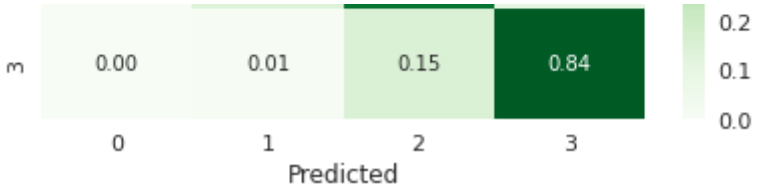



```
0.8975
```

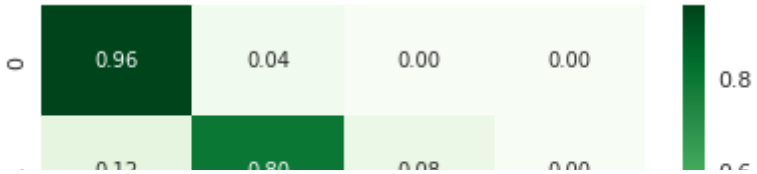



```
0.8225
```



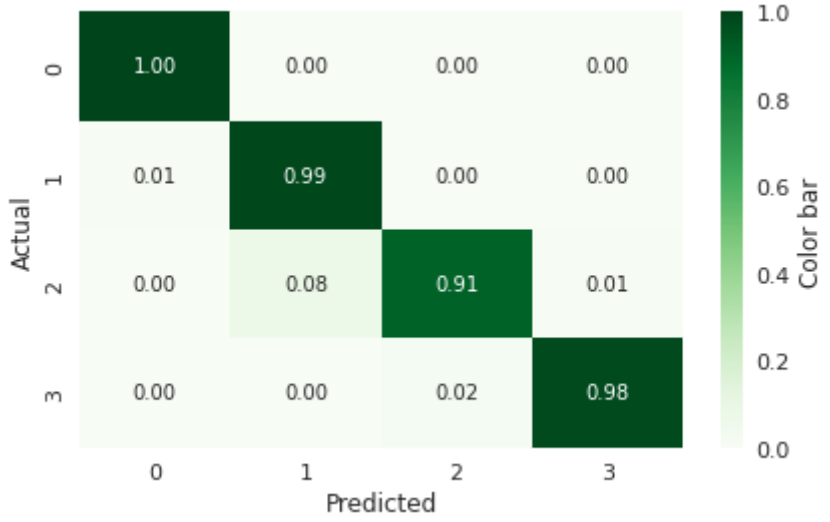



```
0.865
```

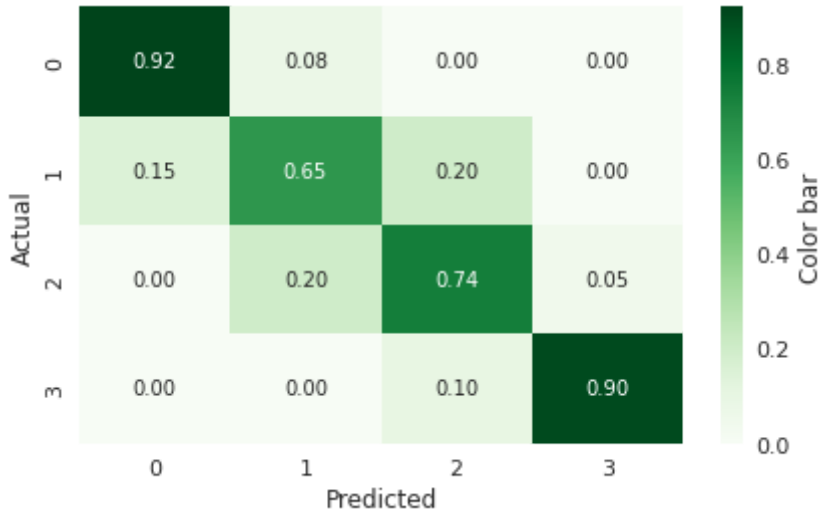



```
my_pipe6 = make_pipeline(cr_scaler(Rscaler, x_train, x_test), my_modelfit(clf, x_train, y_
y_pred_res6 = my_predict(clf,x_test)
print(my_f1_score(y_test,y_pred_res6))
my_conf_matrix(y_test,y_pred_res6)

my_pipe7 = make_pipeline(cr_scaler(Rscaler, x_train, x_test), my_modelfit(clf1, x_train, y
y_pred_res7 = my_predict(clf1,x_test)
print(my_f1_score(y_test,y_pred_res7))
my_conf_matrix(y_test,y_pred_res7)

my_pipe8 = make_pipeline(cr_scaler(Rscaler, x_train, x_test), my_modelfit(clf2, x_train, y
y_pred_res8 = my_predict(clf2,x_test)
print(my_f1_score(y_test,y_pred_res8))
my_conf_matrix(y_test,y_pred_res8)

my_pipe9 = make_pipeline(cr_scaler(Rscaler, x_train, x_test), my_modelfit(clf3, x_train, y
y_pred_res9 = my_predict(clf3,x_test)
print(my_f1_score(y_test,y_pred_res9))
my_conf_matrix(y_test,y_pred_res9)

my_pipe10 = make_pipeline(cr_scaler(Rscaler, x_train, x_test), my_modelfit(clf4, x_train,
y_pred_res10 = my_predict(clf4,x_test)
print(my_f1_score(y_test,y_pred_res10))
my_conf_matrix(y_test,y_pred_res10)

my_pipe11 = make_pipeline(cr_scaler(Rscaler, x_train, x_test), my_modelfit(clf5, x_train,
y_pred_res11 = my_predict(clf5,x_test)
print(my_f1_score(y_test,y_pred_res11))
my_conf_matrix(y_test,y_pred_res11)
```

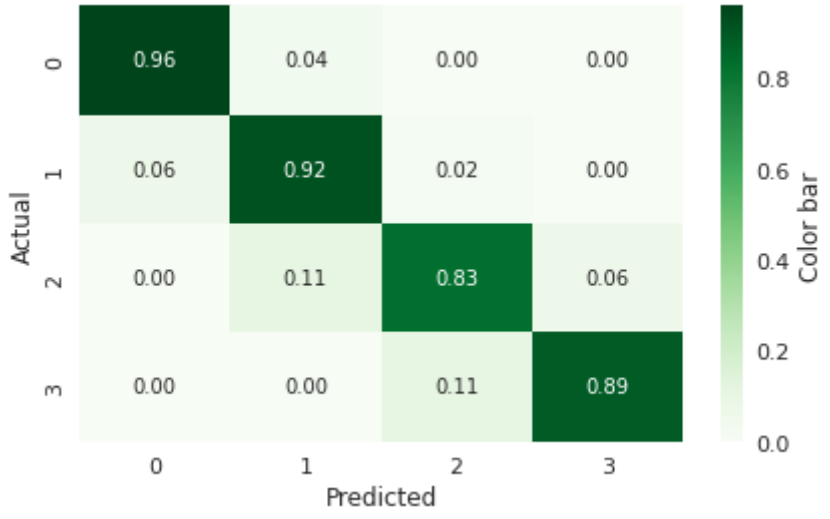
```
0.9725
```




```
0.8075
```




```
0.8975
```




```
0.8275
```