```
import pandas as pd
import numpy as np
import os
import matplotlib.pyplot as plt
import warnings
import seaborn as sns
%matplotlib inline
```

```
df = pd.read_csv('/content/Epileptic Seizure Recognition.csv')
df
```

	Unnamed	X1	X2	X3	X4	X5	X6	X7	X8	X9	...	X170
0	X21.V1.791	135	190	229	223	192	125	55	-9	-33	...	-17
1	X15.V1.924	386	382	356	331	320	315	307	272	244	...	164
2	X8.V1.1	-32	-39	-47	-37	-32	-36	-57	-73	-85	...	57
3	X16.V1.60	-105	-101	-96	-92	-89	-95	-102	-100	-87	...	-82
4	X20.V1.54	-9	-65	-98	-102	-78	-48	-16	0	-21	...	4
...	...	...	...	...	...	...	...	...	...	...	...	...
11495	X22.V1.114	-22	-22	-23	-26	-36	-42	-45	-42	-45	...	15
11496	X19.V1.354	-47	-11	28	77	141	211	246	240	193	...	-65
11497	X8.V1.28	14	6	-13	-16	10	26	27	-9	4	...	-65
11498	X10.V1.932	-40	-25	-9	-12	-2	12	7	19	22	...	121
11499	X16.V1.210	29	41	57	72	74	62	54	43	31	...	-59

```
df = df.replace({'y' : {2:0,3:0,4:0,5:0}})
df
```

	Unnamed	X1	X2	X3	X4	X5	X6	X7	X8	X9	...	X170
0	X21.V1.791	135	190	229	223	192	125	55	-9	-33	...	-17
1	X15.V1.924	386	382	356	331	320	315	307	272	244	...	164
2	X8.V1.1	-32	-39	-47	-37	-32	-36	-57	-73	-85	...	57
3	X16.V1.60	-105	-101	-96	-92	-89	-95	-102	-100	-87	...	-82
4	X20.V1.54	-9	-65	-98	-102	-78	-48	-16	0	-21	...	4
...	...	...	...	...	...	...	...	...	...	...	...	...
11495	X22.V1.114	-22	-22	-23	-26	-36	-42	-45	-42	-45	...	15
11496	X19.V1.354	-47	-11	28	77	141	211	246	240	193	...	-65
11497	X8.V1.28	14	6	-13	-16	10	26	27	-9	4	...	-65
11498	X10.V1.932	-40	-25	-9	-12	-2	12	7	19	22	...	121
11499	X16.V1.210	29	41	57	72	74	62	54	43	31	...	-59

```
x = df.iloc[:,1:179]
x
```

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	...	X169	X170	X
0	135	190	229	223	192	125	55	-9	-33	-38	...	8	-17	
1	386	382	356	331	320	315	307	272	244	232	...	168	164	
2	-32	-39	-47	-37	-32	-36	-57	-73	-85	-94	...	29	57	
3	-105	-101	-96	-92	-89	-95	-102	-100	-87	-79	...	-80	-82	
4	-9	-65	-98	-102	-78	-48	-16	0	-21	-59	...	10	4	

```
y = df.y
y
```

```
0        0
1        1
2        0
3        0
4        0
        ..
11495    0
11496    1
11497    0
11498    0
11499    0
Name: y, Length: 11500, dtype: int64
```

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test= train_test_split(x, y, test_size= 0.25, random_state=1)
```

```
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis,  QuadraticDiscriminantAnalysis
from sklearn.svm import SVC
from sklearn.neural_network import MLPClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import confusion_matrix, accuracy_score, log_loss, precision_score, recall_score, f1_score, roc_auc_score
```

```
classifiers = [
    LogisticRegression(),
    SVC(kernel="rbf", C=0.025, probability=True),
    DecisionTreeClassifier(),
    KNeighborsClassifier(n_neighbors=5),
    GaussianNB(),
    RandomForestClassifier(),
    GradientBoostingClassifier(),
    MLPClassifier()]
```

```
for clf in classifiers:
    clf.fit(x_train, y_train)
    name = clf.__class__.__name__
    print("="*30)
    print(name)
    print('****Results****')
    train_predictions = clf.predict(x_test)
    # calculate score
    acc = accuracy_score(y_test, train_predictions)
    precision = precision_score(y_test, train_predictions, average = 'macro')
    recall = recall_score(y_test, train_predictions, average = 'macro')
    f_score = f1_score(y_test, train_predictions, average = 'macro')
    print("Precision: {:.4%}".format(precision))
    print("Recall: {:.4%}".format(recall))
    print("F-score: {:.4%}".format(f_score))
    print("Accuracy: {:.4%}".format(acc))
    print("="*30)
```

```
==============================
LogisticRegression
****Results****
Precision: 54.5516%
Recall: 56.3696%
F-score: 54.1073%
Accuracy: 64.8348%
```

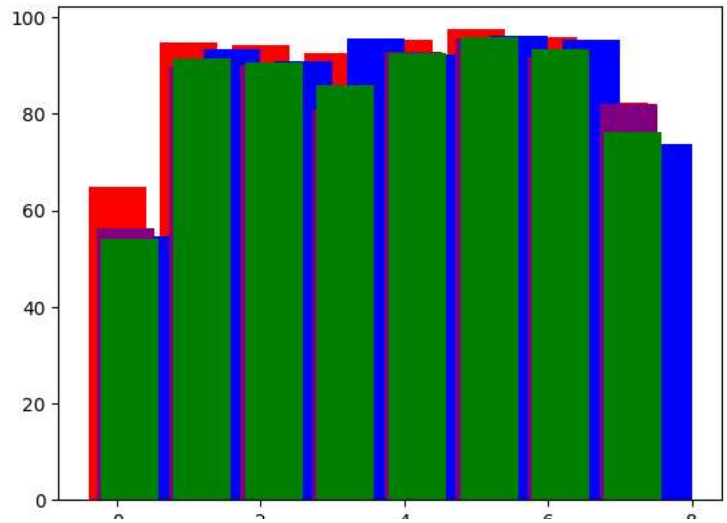
```
==============================
==============================
SVC
****Results****
Precision: 93.4795%
Recall: 89.7367%
F-score: 91.4551%
Accuracy: 94.8870%
==============================
==============================
DecisionTreeClassifier
****Results****
Precision: 91.0051%
Recall: 90.1021%
F-score: 90.5441%
Accuracy: 94.1565%
==============================
==============================
KNeighborsClassifier
****Results****
Precision: 95.5377%
Recall: 81.0161%
F-score: 86.0339%
Accuracy: 92.5913%
==============================
==============================
GaussianNB
****Results****
Precision: 92.2753%
Recall: 92.7916%
F-score: 92.5305%
Accuracy: 95.3043%
==============================
==============================
RandomForestClassifier
****Results****
Precision: 96.1993%
Recall: 95.6293%
F-score: 95.9110%
Accuracy: 97.4609%
==============================
==============================
GradientBoostingClassifier
****Results****
Precision: 95.2895%
Recall: 91.6545%
F-score: 93.3348%
Accuracy: 96.0000%
==============================
==============================
MLPClassifier
```

```
dm = pd.read_csv('/content/ML_Perfomance.csv')
dm
```

	ML_Algorithm	Accuracy	Precision	Recall	F1_score
0	Log R	64.83	54.55	56.36	54.10
1	SVM	94.88	93.47	89.73	91.45
2	DT	94.11	91.00	90.10	90.54
3	KNN	92.59	95.53	81.01	86.03
4	NB	95.30	92.27	92.79	92.53
5	RF	97.46	96.19	95.62	95.91
6	GB	96.00	95.28	91.65	93.33
7	MLP	82.26	73.78	81.90	76.10

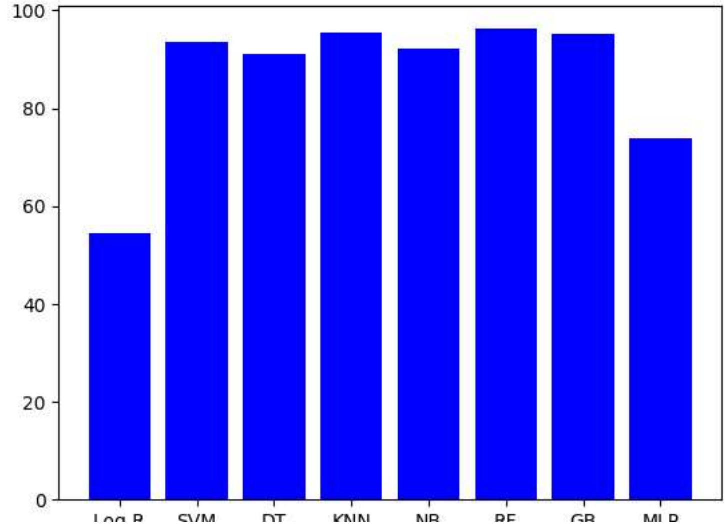
```
X_axis = np.arange(len(dm.ML_Algorithm))
plt.bar(dm.ML_Algorithm,dm.Accuracy,color='Red')
plt.bar(X_axis + 0.6,dm.Precision,color='Blue')
plt.bar(X_axis + 0.12,dm.Recall,color='Purple')
plt.bar(X_axis + 0.18,dm.F1_score,color='Green')
```

```
<BarContainer object of 8 artists>
```



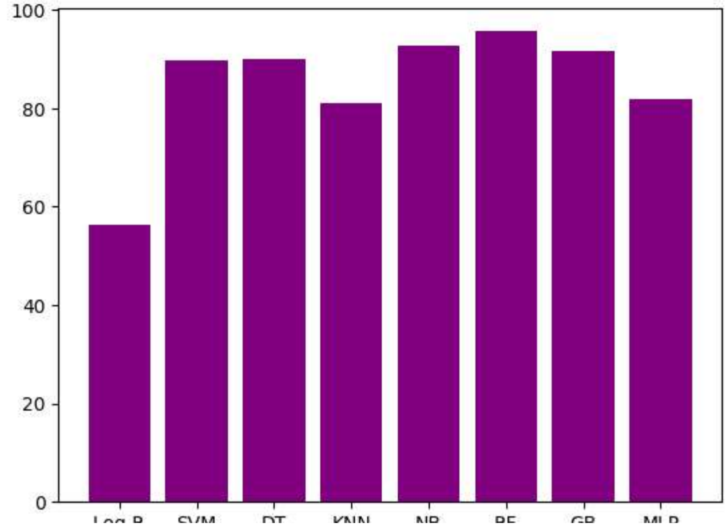
```
plt.bar(dm.ML_Algorithm,dm.Precision,color='Blue')
```

```
<BarContainer object of 8 artists>
```



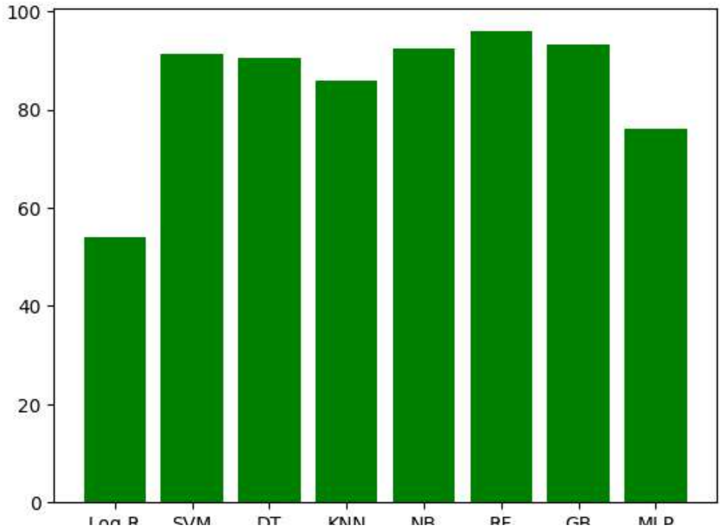
```
plt.bar(dm.ML_Algorithm,dm.Recall,color='Purple')
```

```
<BarContainer object of 8 artists>
```



```
plt.bar(dm.ML_Algorithm,dm.F1_score,color='Green')
```

```
<BarContainer object of 8 artists>
```


100
80
60
40
20
0
Log R
SVM
DT
KNN
NB
RF
GB
MLP