Task-12

GO_STP_5247

Train SVM classifier using sklearn digits dataset

1.Measure accuracy of your model using different kernels such as rbf and linear.
2.Tune your model further using regularization and gamma parameters and try to come up with highest accurancy score
3.Use 80% of samples as training data size

```
[ ] import numpy as np
    import pandas as pd
    import sklearn
    from sklearn.datasets import load_digits
    from sklearn.model_selection import train_test_split
    from sklearn.metrics import confusion_matrix,accuracy_score
```

```
[ ] digits = load_digits()
```

```
[ ] digits.keys()
```

```
dict_keys(['data', 'target', 'target_names', 'images', 'DESCR'])
```

```
print(digits['DESCR'])
```

```
[ ] digits.target_names
```

✓ 0s completed at 1:26 PM

```
df=pd.DataFrame(digits['data'])
df.head()
```

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36
0	0.0	0.0	5.0	13.0	9.0	1.0	0.0	0.0	0.0	0.0	13.0	15.0	10.0	15.0	5.0	0.0	0.0	3.0	15.0	2.0	0.0	11.0	8.0	0.0	0.0	4.0	12.0	0.0	0.0	8.0	8.0	0.0	0.0	5.0	8.0	0.0	0.0
1	0.0	0.0	0.0	12.0	13.0	5.0	0.0	0.0	0.0	0.0	0.0	11.0	16.0	9.0	0.0	0.0	0.0	0.0	3.0	15.0	16.0	6.0	0.0	0.0	0.0	7.0	15.0	16.0	16.0	2.0	0.0	0.0	0.0	0.0	1.0	16.0	16.0
2	0.0	0.0	0.0	4.0	15.0	12.0	0.0	0.0	0.0	0.0	3.0	16.0	15.0	14.0	0.0	0.0	0.0	0.0	8.0	13.0	8.0	16.0	0.0	0.0	0.0	0.0	1.0	6.0	15.0	11.0	0.0	0.0	0.0	1.0	8.0	13.0	15.0
3	0.0	0.0	7.0	15.0	13.0	1.0	0.0	0.0	0.0	8.0	13.0	6.0	15.0	4.0	0.0	0.0	0.0	2.0	1.0	13.0	13.0	0.0	0.0	0.0	0.0	0.0	2.0	15.0	11.0	1.0	0.0	0.0	0.0	0.0	0.0	1.0	12.0
4	0.0	0.0	0.0	1.0	11.0	0.0	0.0	0.0	0.0	0.0	0.0	7.0	8.0	0.0	0.0	0.0	0.0	0.0	1.0	13.0	6.0	2.0	2.0	0.0	0.0	0.0	7.0	15.0	0.0	9.0	8.0	0.0	0.0	5.0	16.0	10.0	0.0

```
df.describe()
```

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	
count	1797.0	1797.000000	1797.000000	1797.000000	1797.000000	1797.000000	1797.000000	1797.000000	1797.000000	1797.000000	1797.000000	1797.000000	1797.000000	1797.000000	1797.00
mean	0.0	0.303840	5.204786	11.835838	11.848080	5.781859	1.362270	0.129661	0.005565	1.993879	10.382304	11.979410	10.279354	8.175849	1.84
std	0.0	0.907192	4.754826	4.248842	4.287388	5.666418	3.325775	1.037383	0.094222	3.196160	5.421456	3.977543	4.782681	6.052960	3.58
min	0.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00
25%	0.0	0.000000	1.000000	10.000000	10.000000	0.000000	0.000000	0.000000	0.000000	0.000000	6.000000	9.000000	7.000000	2.000000	0.00
50%	0.0	0.000000	4.000000	13.000000	13.000000	4.000000	0.000000	0.000000	0.000000	0.000000	12.000000	13.000000	11.000000	9.000000	0.00
75%	0.0	0.000000	9.000000	15.000000	15.000000	11.000000	0.000000	0.000000	0.000000	3.000000	15.000000	16.000000	15.000000	14.000000	2.00
max	0.0	8.000000	16.000000	16.000000	16.000000	16.000000	16.000000	15.000000	2.000000	16.000000	16.000000	16.000000	16.000000	16.000000	16.00

```
df['target']=digits.target
df.head()
```

	digits	target
0	0	0
1	1	1
2	2	2
3	3	3
4	4	4

```
df.target.value_counts()
```

```
3    183
5    182
1    182
6    181
4    181
9    180
7    179
0    178
2    177
8    174
Name: target, dtype: int64
```

```
X=df.drop(['target'],axis='columns')
Y=df.target
```

```
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.2,random_state=5)
```

```
[ ] from sklearn.svm import SVC
```

```
[ ] mymodel = SVC()
```

```
[ ] mymodel.fit(X_train,Y_train)
```

```
SVC(C=1.0, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='scale', kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
```

```
[ ] mymodel.score(X_test,Y_test)
```

```
0.9777777777777777
```

```
[ ] pred=mymodel.predict(X_test)
```

```
[ ] cm=np.array(confusion_matrix(Y_test,pred))
    cm
```

```
array([[35,  0,  0,  0,  0,  0,  0,  0,  0,  0],
       [ 0, 35,  0,  0,  0,  0,  0,  0,  0,  0],
       [ 0,  0, 36,  0,  0,  0,  0,  0,  0,  0],
       [ 0,  0,  0, 35,  0,  1,  0,  0,  1,  0],
       [ 0,  0,  0,  0, 32,  0,  0,  0,  0,  0],
       [ 0,  0,  0,  0,  0, 45,  0,  0,  0,  1],
       [ 0,  0,  0,  0,  0,  0, 30,  0,  0,  0],
       [ 0,  0,  0,  0,  0,  0,  0, 41,  0,  1],
       [ 0,  1,  0,  0,  0,  0,  0,  0, 36,  1],
       [ 0,  0,  0,  0,  0,  1,  0,  0,  1, 27]])
```

Tuning

```
[ ] model_C=SVC(C=1)
    model_C.fit(X_train,Y_train)
    model_C.score(X_test,Y_test)
```

```
0.9777777777777777
```

```
[ ] model_C=SVC(C=2)
    model_C.fit(X_train,Y_train)
    model_C.score(X_test,Y_test)
```

```
0.9833333333333333
```

```
[ ] model_C=SVC(C=3)
    model_C.fit(X_train,Y_train)
    model_C.score(X_test,Y_test)
```

```
0.9861111111111112
```

```
[ ] model_kernel=SVC(C=3,kernel='linear')
    model_kernel.fit(X_train,Y_train)
    model_kernel.score(X_test,Y_test)
```

```
0.975
```

```
[ ] model_kernel=SVC(C=3,kernel='rbf')
    model_kernel.fit(X_train,Y_train)
    model_kernel.score(X_test,Y_test)
```

```
0.9861111111111112
```

```
0.9861111111111112
```

```
[ ] model_g=SVC(C=3,kernel='poly',degree=5, gamma='auto')
    model_g.fit(X_train,Y_train)
    model_g.score(X_test,Y_test)
```

```
0.9833333333333333
```

```
[ ] finalModel=SVC(C=3)
    finalModel.fit(X_train,Y_train)
    finalModel.score(X_test,Y_test)
```

```
0.9861111111111112
```

```
[ ] pred=finalModel.predict(X_test)
    cm=np.array(confusion_matrix(Y_test,pred))
    cm
```

```
array([[35,  0,  0,  0,  0,  0,  0,  0,  0,  0],
       [ 0, 35,  0,  0,  0,  0,  0,  0,  0,  0],
       [ 0,  0, 36,  0,  0,  0,  0,  0,  0,  0],
       [ 0,  0,  1, 35,  0,  1,  0,  0,  0,  0],
       [ 0,  0,  0,  0, 32,  0,  0,  0,  0,  0],
       [ 0,  0,  0,  0,  0, 45,  0,  0,  0,  1],
       [ 0,  0,  0,  0,  0,  0, 30,  0,  0,  0],
       [ 0,  0,  0,  0,  0,  0,  0, 41,  0,  1],
       [ 0,  0,  0,  0,  0,  0,  0,  0, 38,  0],
       [ 0,  0,  0,  0,  0,  1,  0,  0,  0, 28]])
```