Subject Train SVM classifier using sklearn digits dataset (i.e. from sklearn.datasets import load_digits) and then:

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

```
import pandas as pd
from sklearn.svm import SVC
from sklearn.model_selection import train_test_split
from sklearn.datasets import load_digits
```

```
digits = load_digits()
dir(digits)
```

```
     ['DESCR', 'data', 'images', 'target', 'target_names']
```

```
digits.target_names
```

```
     array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

```
digits.target
```

```
     array([0, 1, 2, ..., 8, 9, 8])
```

```
digits.data[0]
```

```
     array([ 0.,  0.,  5., 13.,  9.,  1.,  0.,  0.,  0.,  0., 13., 15., 10.,
            15.,  5.,  0.,  0.,  3., 15.,  2.,  0., 11.,  8.,  0.,  0.,  4.,
            12.,  0.,  0.,  8.,  8.,  0.,  0.,  5.,  8.,  0.,  0.,  9.,  8.,
             0.,  0.,  4., 11.,  0.,  1., 12.,  7.,  0.,  0.,  2., 14.,  5.,
            10., 12.,  0.,  0.,  0.,  0.,  6., 13., 10.,  0.,  0.,  0.])
```

```
df = pd.DataFrame(digits.data,digits.target)
df.head()
```

```
        0    1    2    3    4    5    6    7    8    9    10    11    12    13   14   15   16   1
```

```
df['target'] = digits.target
df.head(15)
```

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
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
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
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
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
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
5	0.0	0.0	12.0	10.0	0.0	0.0	0.0	0.0	0.0	0.0	14.0	16.0	16.0	14.0	0.0	0.0	0.0
6	0.0	0.0	0.0	12.0	13.0	0.0	0.0	0.0	0.0	0.0	5.0	16.0	8.0	0.0	0.0	0.0	0.0
7	0.0	0.0	7.0	8.0	13.0	16.0	15.0	1.0	0.0	0.0	7.0	7.0	4.0	11.0	12.0	0.0	0.0
8	0.0	0.0	9.0	14.0	8.0	1.0	0.0	0.0	0.0	0.0	12.0	14.0	14.0	12.0	0.0	0.0	0.0
9	0.0	0.0	11.0	12.0	0.0	0.0	0.0	0.0	0.0	2.0	16.0	16.0	16.0	13.0	0.0	0.0	0.0
0	0.0	0.0	1.0	9.0	15.0	11.0	0.0	0.0	0.0	0.0	11.0	16.0	8.0	14.0	6.0	0.0	0.0
1	0.0	0.0	0.0	0.0	14.0	13.0	1.0	0.0	0.0	0.0	0.0	5.0	16.0	16.0	2.0	0.0	0.0
2	0.0	0.0	5.0	12.0	1.0	0.0	0.0	0.0	0.0	0.0	15.0	14.0	7.0	0.0	0.0	0.0	0.0
3	0.0	2.0	9.0	15.0	14.0	9.0	3.0	0.0	0.0	4.0	13.0	8.0	9.0	16.0	8.0	0.0	0.0
4	0.0	0.0	0.0	8.0	15.0	1.0	0.0	0.0	0.0	0.0	1.0	14.0	13.0	1.0	1.0	0.0	0.0

```
X_train, X_test, y_train, y_test = train_test_split(df.drop('target',axis='columns'),df.targe
```

```
rbf_model = SVC(kernel='rbf',gamma=0.002)
rbf_model.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=0.002, kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
```

```
rbf_model.score(X_test,y_test)
```

```
0.9944444444444445
```

```
linear_model = SVC(kernel='linear',C=0.001)
linear_model.fit(X_train,y_train)
```

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

```
linear_model.score(X_test,y_test)
```

```
0.9833333333333333
```

✓ 0s completed at 19:20 ● ×