60TH-SVM IRIS

December 18, 2021

[67]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.datasets import load_iris
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split

iris = load_iris()
iris_df = pd.DataFrame(iris['data'] )
#Now add the column names.
iris_df.columns = iris['feature_names']
iris_df['species']= iris['target']
iris_df.head()
```

[67]:

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	\
0	5.1	3.5	1.4	0.2	
1	4.9	3.0	1.4	0.2	
2	4.7	3.2	1.3	0.2	
3	4.6	3.1	1.5	0.2	
4	5.0	3.6	1.4	0.2	

	species
0	0
1	0
2	0
3	0
4	0

[69]:
```
# Creating Train and Test datasets
# Extracting Attributes / Features
X = iris_df.iloc[:,0:4]
# Extracting Target / Class Labels
y = iris_df['species']
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state = 50,␣
 ↪test_size = 0.3)
print(y_test)
print(X_test)
```

```
88     1
72     1
20     0
16     0
147    2
140    2
113    2
23     0
12     0
68     1
39     0
130    2
34     0
112    2
55     1
25     0
82     1
48     0
81     1
77     1
100    2
80     1
14     0
131    2
86     1
118    2
56     1
54     1
97     1
143    2
125    2
98     1
73     1
144    2
21     0
3      0
59     1
119    2
84     1
7      0
41     0
57     1
```

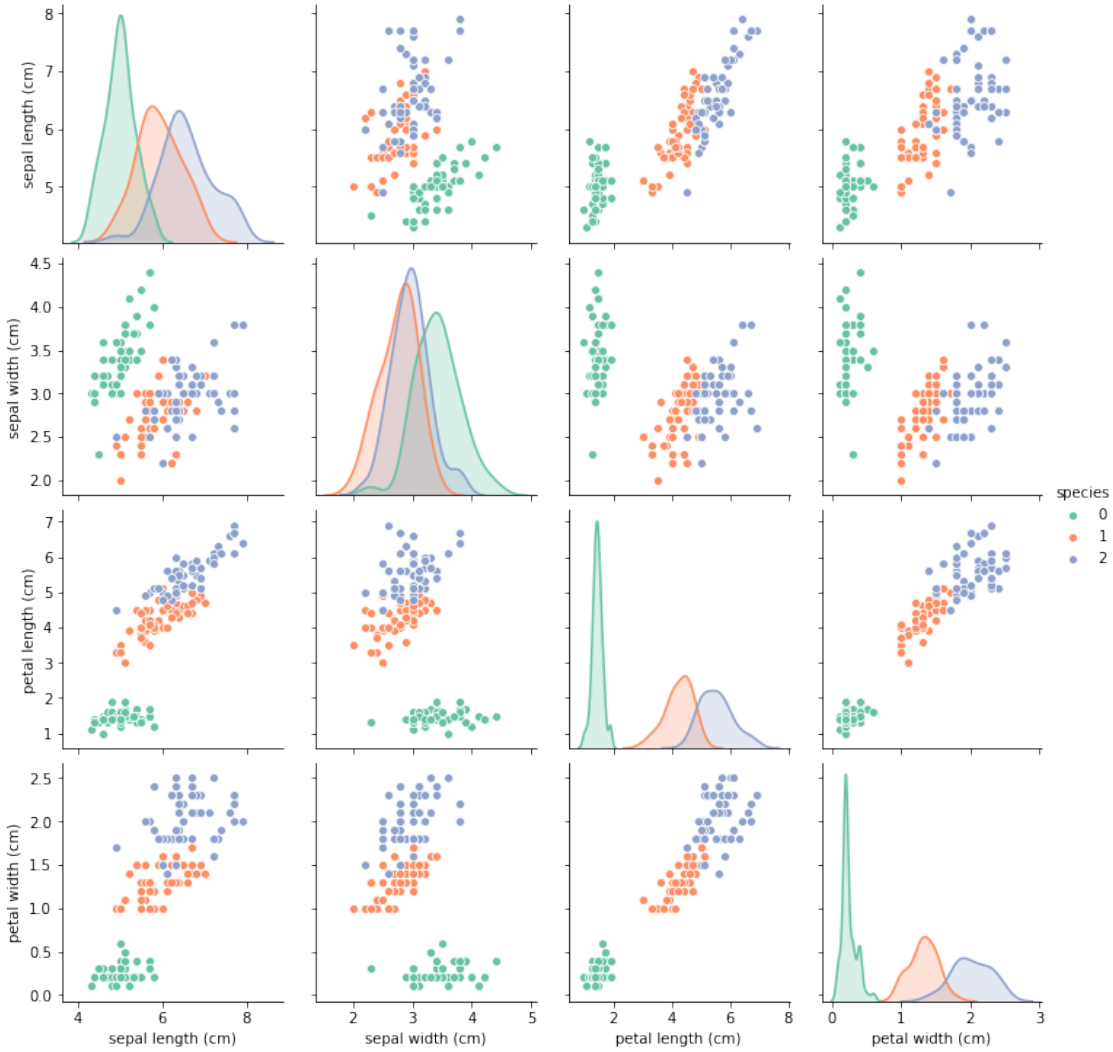
```
104    2
8      0
102    2
Name: species, dtype: int64
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
88	5.6	3.0	4.1	1.3
72	6.3	2.5	4.9	1.5
20	5.4	3.4	1.7	0.2
16	5.4	3.9	1.3	0.4
147	6.5	3.0	5.2	2.0
140	6.7	3.1	5.6	2.4
113	5.7	2.5	5.0	2.0
23	5.1	3.3	1.7	0.5
12	4.8	3.0	1.4	0.1
68	6.2	2.2	4.5	1.5
39	5.1	3.4	1.5	0.2
130	7.4	2.8	6.1	1.9
34	4.9	3.1	1.5	0.2
112	6.8	3.0	5.5	2.1
55	5.7	2.8	4.5	1.3
25	5.0	3.0	1.6	0.2
82	5.8	2.7	3.9	1.2
48	5.3	3.7	1.5	0.2
81	5.5	2.4	3.7	1.0
77	6.7	3.0	5.0	1.7
100	6.3	3.3	6.0	2.5
80	5.5	2.4	3.8	1.1
14	5.8	4.0	1.2	0.2
131	7.9	3.8	6.4	2.0
86	6.7	3.1	4.7	1.5
118	7.7	2.6	6.9	2.3
56	6.3	3.3	4.7	1.6
54	6.5	2.8	4.6	1.5
97	6.2	2.9	4.3	1.3
143	6.8	3.2	5.9	2.3
125	7.2	3.2	6.0	1.8
98	5.1	2.5	3.0	1.1
73	6.1	2.8	4.7	1.2
144	6.7	3.3	5.7	2.5
21	5.1	3.7	1.5	0.4
3	4.6	3.1	1.5	0.2
59	5.2	2.7	3.9	1.4
119	6.0	2.2	5.0	1.5
84	5.4	3.0	4.5	1.5
7	5.0	3.4	1.5	0.2
41	4.5	2.3	1.3	0.3
57	4.9	2.4	3.3	1.0
104	6.5	3.0	5.8	2.2

8	4.4	2.9	1.4	0.2
102	7.1	3.0	5.9	2.1

[21]:
```
# Creating a pairplot to visualize the similarities and especially difference␣
↪between the species
sns.pairplot(data=iris_df, hue='species', palette='Set2')
```

[21]: <seaborn.axisgrid.PairGrid at 0x7fd46c753ed0>



[77]:
```
from sklearn.svm import SVC
#model=SVC(kernel='linear')
#model=SVC(C=10, kernel='poly', degree=4)
model=SVC(C=1, kernel='rbf', gamma=0.1)

model.fit(X_train, y_train)
```

[77]:
```
SVC(C=1, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma=0.1, kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
```

[78]:
```
# Predict Accuracy Score
y_pred = model.predict(X_test)
```

[79]:
```
# Importing the classification report and confusion matrix
from sklearn.metrics import classification_report, confusion_matrix

print("Test data accuracy:", accuracy_score(y_true = y_test, y_pred=y_pred))
print(confusion_matrix(y_test, y_pred))
```

```
Test data accuracy: 0.9555555555555556
[[14  0  0]
 [ 0 15  2]
 [ 0  0 14]]
```

[73]:
```
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        14
           1       1.00      0.88      0.94        17
           2       0.88      1.00      0.93        14

    accuracy                           0.96        45
   macro avg       0.96      0.96      0.96        45
weighted avg       0.96      0.96      0.96        45
```

[36]:
```
#Random Forests
from sklearn.ensemble import RandomForestClassifier
rfc = RandomForestClassifier(n_estimators=500)
rfc.fit(X_train, y_train)
```

[36]:
```
RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                       criterion='gini', max_depth=None, max_features='auto',
                       max_leaf_nodes=None, max_samples=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, n_estimators=500,
                       n_jobs=None, oob_score=False, random_state=None,
                       verbose=0, warm_start=False)
```

[37]:
```
rfc_pred = rfc.predict(X_test)
print("Test data accuracy:",accuracy_score(y_true = y_test, y_pred=rfc_pred))
print(confusion_matrix(y_test, rfc_pred))
```

```
Test data accuracy: 0.9555555555555556
[[14  0  0]
 [ 0 16  1]
 [ 0  1 13]]
```

[30]:
```
print(classification_report(y_test,rfc_pred))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        14
           1       0.94      0.94      0.94        17
           2       0.93      0.93      0.93        14

    accuracy                           0.96        45
   macro avg       0.96      0.96      0.96        45
weighted avg       0.96      0.96      0.96        45
```

[ ]:
```

```