Lecture24

May 2, 2024

1 Data Classification

[6]:
```
import pandas as pd
df=pd.read_csv('datafiles/iris.csv')
# df.species.unique()
df.head()
```

[6]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

1.0.1 Split the Data

[9]:
```
from sklearn.model_selection import train_test_split
X=df.iloc[:,:-1]
Y=df.iloc[:,-1]
X_train, X_test, Y_train, Y_test =train_test_split(X,Y, test_size=.2)
```

1.0.2 Build the classifier

[26]:
```
from sklearn.neighbors import KNeighborsClassifier
my_classifier = KNeighborsClassifier(n_neighbors = 10) # n_neighbors= sqrt(n)
my_classifier.fit(X_train, Y_train)
```

[26]:
```
KNeighborsClassifier(n_neighbors=10)
```

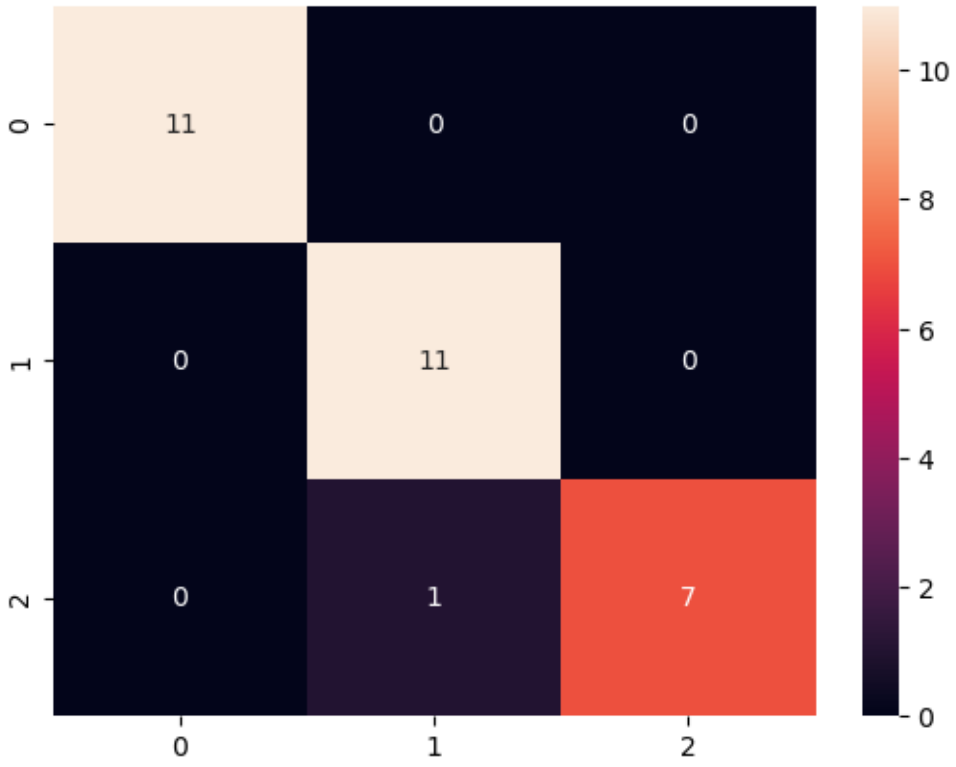
1.0.3 Evaluate the Classifier

[27]:
```
# pip install seaborn
```

[28]:
```
import sklearn.metrics as m
Y_hat = my_classifier.predict(X_test)
cm=m.confusion_matrix(Y_test, Y_hat)
```

[29]:
```
import seaborn as sns
sns.heatmap(cm, annot=True)
```

[29]:
```
<Axes: >
```



[30]:
```
m.accuracy_score(Y_test, Y_hat)
```

[30]:
```
0.9666666666666667
```

2 Regression

[33]:
```
df=pd.read_csv('datafiles/deliveryTime.csv', index_col=0)
df.head()
```

[33]:

	milesTraveled	numDeliveries	gasPrice	travelTime
0	89.0	4.0	3.74	7.0
1	43.9	1.0	3.47	4.8
2	88.8	4.0	4.04	7.0
3	76.9	2.9	3.67	6.4
4	77.0	3.0	3.67	6.4

2.0.1 Evaluate Correlation and Determine X and Y

[36]:
```
from scipy import stats
stats.pearsonr(df.gasPrice, df.travelTime)
stats.pearsonr(df.milesTraveled, df.travelTime)
stats.pearsonr(df.numDeliveries, df.travelTime)
```

[36]:
```
PearsonRResult(statistic=0.9122483165073335, pvalue=0.0)
```

[37]:
```
stats.pearsonr(df.numDeliveries, df.milesTraveled)
```

[37]:
```
PearsonRResult(statistic=0.9566863689509586, pvalue=0.0)
```

[ ]:
```
# import numpy as np
# np.cov(x,y)
```

[44]:
```
X = df.iloc[:, [0]]
Y = df.iloc[:,-1]
```

2.0.2 Split the Data

[46]:
```
X_train, X_test, Y_train, Y_test= train_test_split(X,Y)
```

2.0.3 Create the Model

[49]:
```
from sklearn import linear_model
my_model=linear_model.LinearRegression()
my_model.fit(X_train, Y_train)
```

[49]:
```
LinearRegression()
```

2.0.4 Evaluate the model

[53]:
```
Y_hat = my_model.predict(X_test)
m.mean_squared_error(Y_test, Y_hat)
m.r2_score(Y_test, Y_hat)
```

[53]:
```
0.856603405836817
```

2.0.5 Overfitting vs underfitting

[ ]:
```
# Look at the learning curve (Training vs Test)
```

[84]:
```
np.linspace(.1,1,10)
```

[84]:
```
array([0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1. ])
```

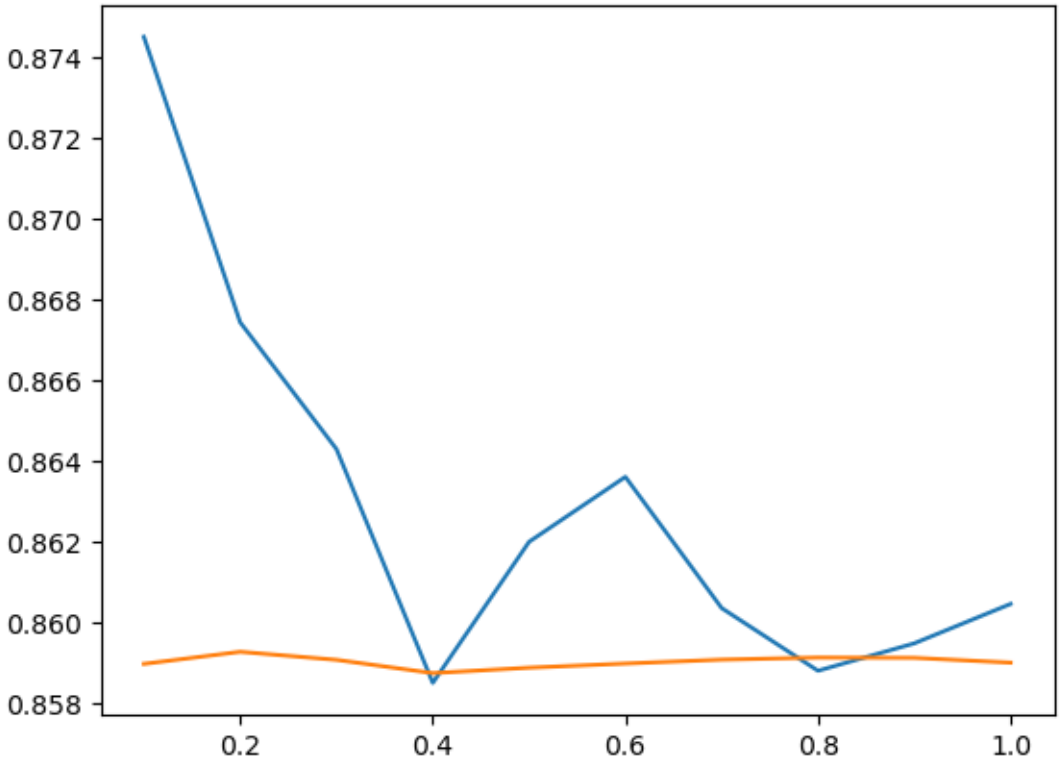
[68]:
```
import numpy as np
from sklearn.model_selection import learning_curve
train_sizes, train_scores, validation_scores = learning_curve(my_model, X, Y,␣
 ↪train_sizes= np.linspace(.1,1,10))
```

[81]:
```
train_mean_score = np.mean(train_scores, axis=1)
train_std_score = np.std(train_scores, axis=1)
validation_mean_score = np.mean(validation_scores, axis=1)
validation_std_score = np.std(validation_scores, axis=1)
```

[82]:
```
import matplotlib.pyplot as plt
```

[83]:
```
plt.plot(np.linspace(.1,1,10), train_mean_score)
plt.plot(np.linspace(.1,1,10), validation_mean_score)
```

[83]:
```
[<matplotlib.lines.Line2D at 0x7f4665ab74d0>]
```



[80]:
```
validation_mean_score
```

[80]:
```
0.8590180924601394
```

[ ]:
```

```