Lab 8 - CCPS 844 Data Mining

Salman AlMaskati

Answer the following questions and submit a PDF file on the D2L.

Select a dataset of your choice.

- Apply Random Forest and Decision Tree
- Evaluate the algos by changing parameter "criterion" for both (entropy, gini) and "n_estimators" for Random Forest
- Compare your results (evaluation metrics)

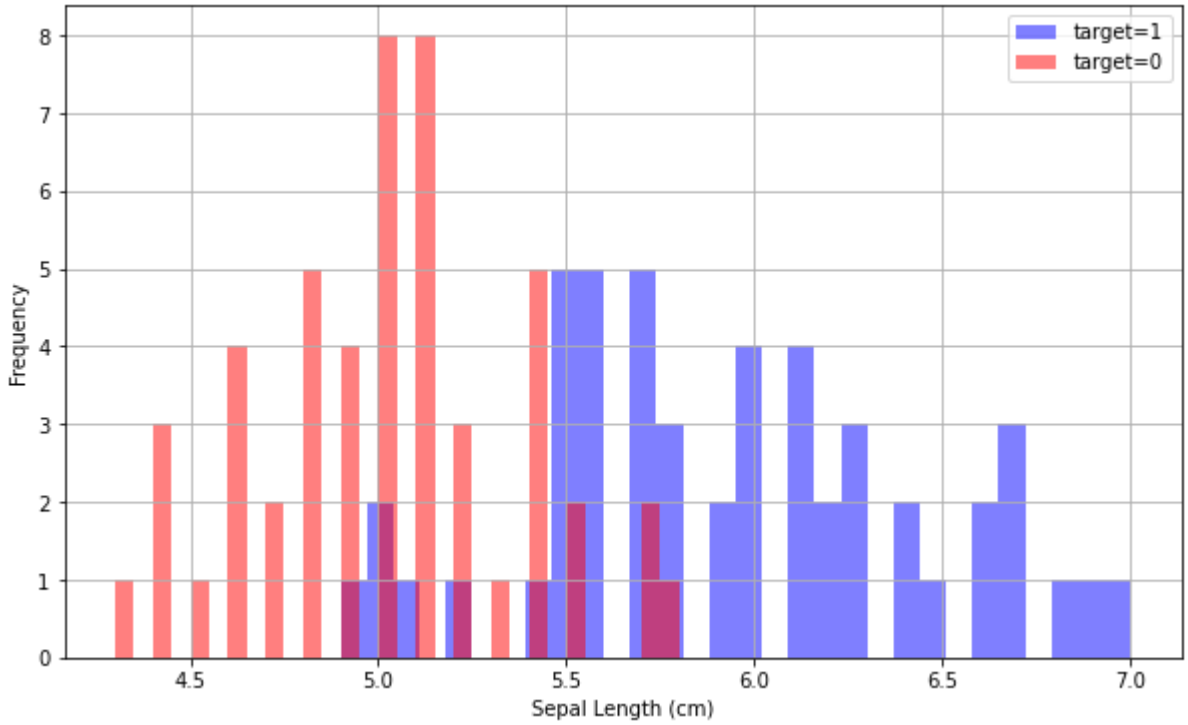
In [ ]:
```
import pandas as pd
from sklearn import datasets
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [ ]:
```
iris = datasets.load_iris()
iris_df = pd.DataFrame(data=iris.data, columns=iris.feature_names)
iris_df['target'] = iris.target
iris_df.head()
```

Out[ ]:

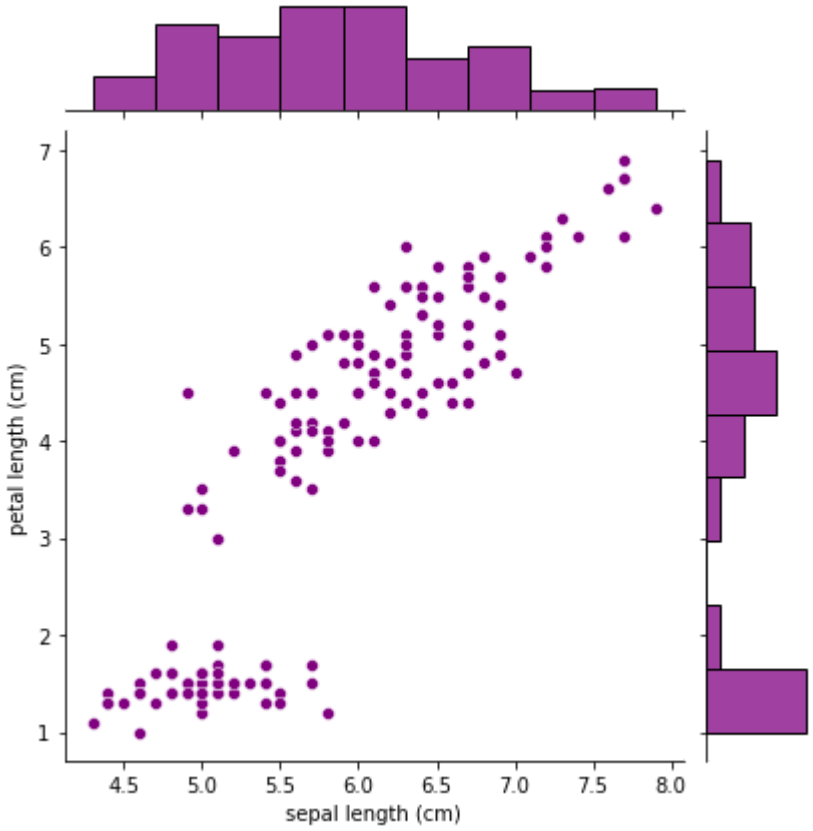
	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	target
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0
3	4.6	3.1	1.5	0.2	0
4	5.0	3.6	1.4	0.2	0

In [ ]:
```
# Plot histograms for target=1 and target=0 based on 'sepal length'
plt.figure(figsize=(10, 6))
iris_df[iris_df['target'] == 1]['sepal length (cm)'].hist(alpha=0.5, color='blue', bins
iris_df[iris_df['target'] == 0]['sepal length (cm)'].hist(alpha=0.5, color='red', bins=
plt.xlabel('Sepal Length (cm)')
plt.ylabel('Frequency')
plt.legend()
plt.show()
```



In [ ]:
```
#trend between sepal length and petal length
sns.jointplot(x='sepal length (cm)',y='petal length (cm)',data=iris_df,color='purple')
```

Out[ ]:
```
<seaborn.axisgrid.JointGrid at 0x1f3cd36af70>
```



Do not need dummmies (NO categorical features).

In [ ]:
```
from sklearn.model_selection import train_test_split
```

In [ ]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, random_state=
```

In [ ]:
```
from sklearn.tree import DecisionTreeClassifier
```

In [ ]:
```
dtree = DecisionTreeClassifier()
dtree.fit(X_train,y_train)
```

Out[ ]:
```
DecisionTreeClassifier()
```

In [ ]:
```
predictions = dtree.predict(X_test)
```

In [ ]:
```
from sklearn.metrics import classification_report,confusion_matrix
print(classification_report(y_test,predictions))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        13
           1       0.95      0.95      0.95        20
           2       0.92      0.92      0.92        12

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

In [ ]:
```
print(confusion_matrix(y_test,predictions))
```

```
[[13  0  0]
 [ 0 19  1]
 [ 0  1 11]]
```

In [ ]:
```
from sklearn.ensemble import RandomForestClassifier
```

In [ ]:
```
rfc = RandomForestClassifier(n_estimators=700)
rfc.fit(X_train,y_train)
```

Out[ ]:
```
RandomForestClassifier(n_estimators=700)
```

In [ ]:
```
predictions2 = rfc.predict(X_test)
```

In [ ]:
```
from sklearn.metrics import classification_report,confusion_matrix
```

In [ ]:
```
print(classification_report(y_test,predictions2))
```

```
                   precision    recall  f1-score   support

               0       1.00      1.00      1.00        13
               1       0.95      0.95      0.95        20
               2       0.92      0.92      0.92        12

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

In [ ]:
```
print(confusion_matrix(y_test,predictions2))
```

```
[[13  0  0]
 [ 0 19  1]
 [ 0  1 11]]
```

In [ ]:
```

```