HELP: HW1 CSC 84020 –Neural Networks and Deep Learning

Descriptive Statistics, Analysis and Classification Using Python and Python Libraries

Description of Iris flower data set

- Attributes are numeric (float values) and they are:
sepal -length
sepal - width
petal - length
petal - width
- The classes are: Iris Setosa, Iris Versicolor and Iris Virginica
- The number of instances of each class is 50

All the attribute values are in the same units (cm) and same scale as well.

Listing 1a: Load libraries

In [1]:
```
# Load libraries
from pandas import read_csv
from pandas.plotting import scatter_matrix
from matplotlib import pyplot
from sklearn.model_selection import train_test_split
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
```

In [ ]:
```
1
```

Listing 1.b: Load the Iris dataset

In [4]:
```
# Load dataset
filename = 'iris.data.csv'
names = ['sepal-length', 'sepal-width', 'petal-length', 'petal-width', 'class']
dataset = read_csv(filename, names=names, delimiter = '\t')
```

In [ ]:
```
1
```

Listing 2 – Dimensions of the dataset. Peek at the data itself. Statistical summary of all attributes. Breakdown of the data by the class variable.

a) Print the shape of the data-set.

```
In [6]:
# Summarize Data

# Descriptive statistics
# shape
print(dataset.shape)
```

```
(150, 5)
```

```
In [ ]:
```

b) Print the first few rows of the data-set.

```
In [2]:
# head
print(dataset.head(20))
```

```
    sepal-length  sepal-width  petal-length  petal-width        class
0            5.1          3.5           1.4          0.2  Iris-setosa
1            4.9          3.0           1.4          0.2  Iris-setosa
2            4.7          3.2           1.3          0.2  Iris-setosa
3            4.6          3.1           1.5          0.2  Iris-setosa
4            5.0          3.6           1.4          0.2  Iris-setosa
5            5.4          3.9           1.7          0.4  Iris-setosa
6            4.6          3.4           1.4          0.3  Iris-setosa
7            5.0          3.4           1.5          0.2  Iris-setosa
8            4.4          2.9           1.4          0.2  Iris-setosa
9            4.9          3.1           1.5          0.1  Iris-setosa
10           5.4          3.7           1.5          0.2  Iris-setosa
11           4.8          3.4           1.6          0.2  Iris-setosa
12           4.8          3.0           1.4          0.1  Iris-setosa
13           4.3          3.0           1.1          0.1  Iris-setosa
14           5.8          4.0           1.2          0.2  Iris-setosa
15           5.7          4.4           1.5          0.4  Iris-setosa
16           5.4          3.9           1.3          0.4  Iris-setosa
17           5.1          3.5           1.4          0.3  Iris-setosa
18           5.7          3.8           1.7          0.3  Iris-setosa
19           5.1          3.8           1.5          0.3  Iris-setosa
```

c) Print the statistical descriptions of the data-set.

```
In [3]:
1 print(dataset.describe())
```

```
       sepal-length  sepal-width  petal-length  petal-width
count    150.000000   150.000000    150.000000   150.000000
mean       5.843333     3.054000      3.758667     1.198667
std        0.828066     0.433594      1.764420     0.763161
min        4.300000     2.000000      1.000000     0.100000
25%        5.100000     2.800000      1.600000     0.300000
50%        5.800000     3.000000      4.350000     1.300000
75%        6.400000     3.300000      5.100000     1.800000
max        7.900000     4.400000      6.900000     2.500000
```

d) Print the class distribution in the data-set.

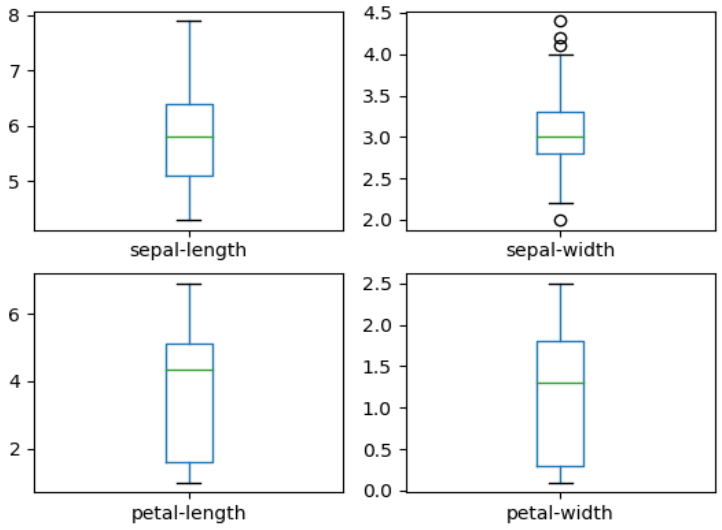
```
In [4]:
1 # class distribution
2 print(dataset.groupby('class').size())
```

```
class
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
dtype: int64
```

Listing 3. Univariate plots to better understand each attribute. Multivariate plots to better understand the relationships between attributes.

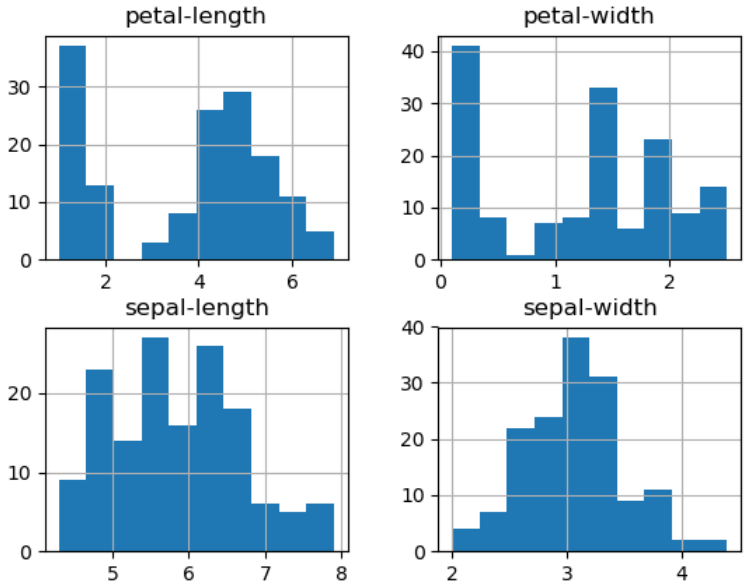
a) Univariate plot.

```
In [5]:
1 # box and whisker plots
2 dataset.plot(kind='box', subplots=True, layout=(2,2), sharex=False, sharey=False)
3 pyplot.show()
```



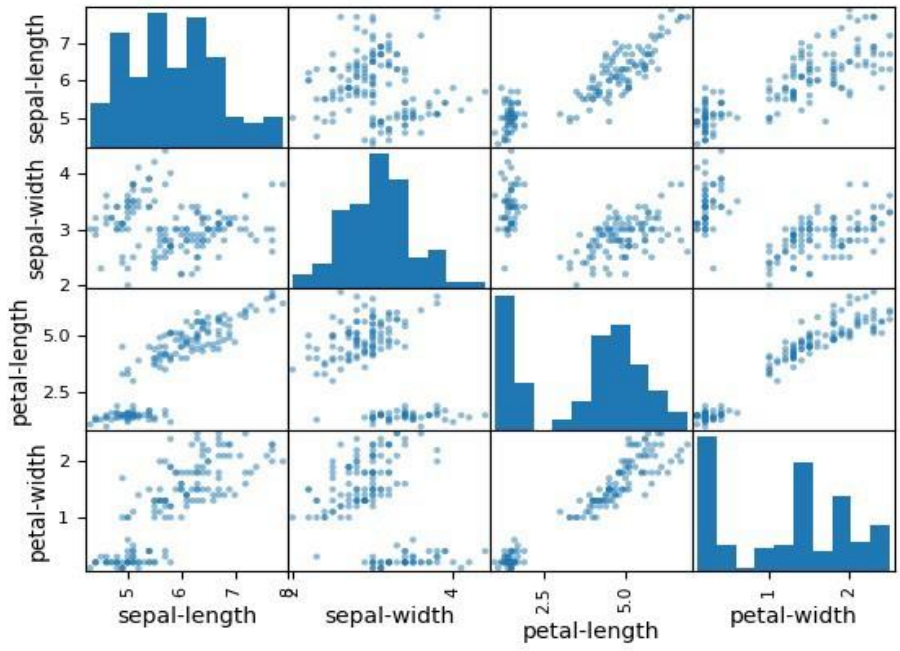
b) Visualize the data-set using histogram plots.

In [6]:
```
# histograms
dataset.hist()
pyplot.show()
```



c) Visualize the dataset using scatter plots.

In [7]:
```
# scatter plot matrix
scatter_matrix(dataset)
pyplot.show()
```



Listing 4 – Separate out a validation dataset. **Setup the test harness to use 10-fold cross-validation (not in this code, but you might want to include it.)** Build 5 different models to predict species from flower measurements. Select the best model.

a) Create validation set.

```
# Split-out validation dataset
array = dataset.values
X = array[:,0:4]
Y = array[:,4]
validation_size = 0.20
seed = 7
X_train, X_validation, Y_train, Y_validation = train_test_split(X, Y,
    test_size=validation_size, random_state=seed)
```

b) Build models **(**Logistic Regression (LR), Linear Discriminant Analysis (LDA), k-Nearest Neighbors (KNN), Classifications and Regression Trees (CART), Gaussian Naive Bayes (NB), Support Vector Machines (SVM) and select the best model.

In [14]:

```
# Spot-Check Algorithms
models = []
models.append(('LR', LogisticRegression()))
models.append(('LDA', LinearDiscriminantAnalysis()))
models.append(('KNN', KNeighborsClassifier()))
models.append(('CART', DecisionTreeClassifier()))
models.append(('NB', GaussianNB()))
models.append(('SVM', SVC(gamma='auto')))
# evaluate each model in turn
results = []
names = []
for name, model in models:
    kfold = KFold(n_splits=10, random_state=seed)
    cv_results = cross_val_score(model, X_train, Y_train, cv=kfold, scoring='accuracy')
    results.append(cv_results)
    names.append(name)
    msg = "%s: %f (%f)" % (name, cv_results.mean(), cv_results.std())
    print(msg)
```

```
LR: 0.966667 (0.040825)
LDA: 0.975000 (0.038188)
KNN: 0.983333 (0.033333)
CART: 0.983333 (0.033333)
NB: 0.975000 (0.053359)
SVM: 0.991667 (0.025000)
```

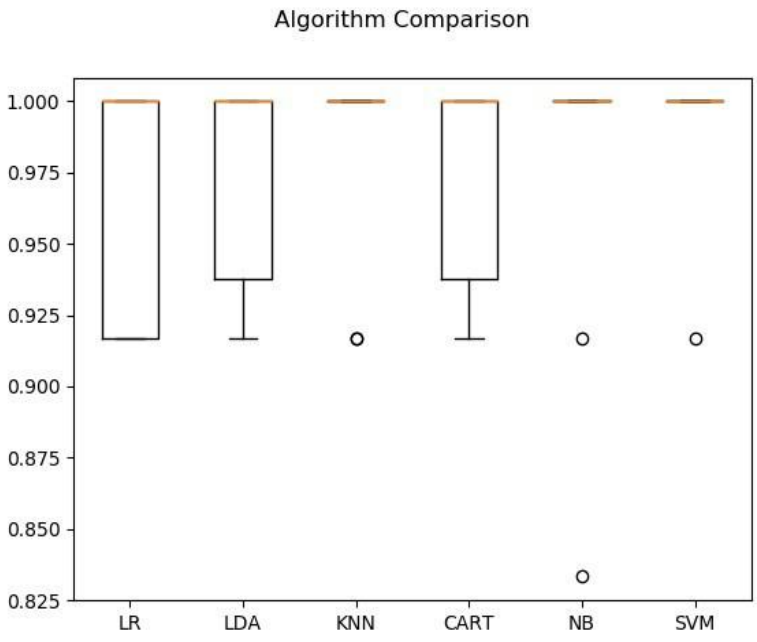
In [ ]:

```
1
```

c) Compare algorithms.

n [15]:

```
# Compare Algorithms
fig = pyplot.figure()
fig.suptitle('Algorithm Comparison')
ax = fig.add_subplot(111)
pyplot.boxplot(results)
ax.set_xticklabels(names)
pyplot.show()
```



Listing 5. Make Predictions on the Validation Data-set.

In [16]:

```
#Make predictions on validation dataset
knn = KNeighborsClassifier()
knn.fit(X train, Y train)
predictions = knn.predict(X validation)
print(accuracy score(Y validation, predictions))
print(confusion matrix(Yvalidation, predictions))
print(classification_report(Y_validation, predictions))
```

```
0.9
[[ 7  0  0]
 [ 0 11  1]
 [ 0  2  9]]
```

	precision	recall	fl-score	support
Iris-setosa	1.00	1.00	1.00	7
Iris-versicolor	0.85	0.92	0.88	12
Iris-virginica	0.90	0.82	0.86	11
micro avg	0.90	0.90	0.90	30
macro avg	0.92	0.91	0.91	30
weighted avg	0.90	0.90	0.90	30