Assignment1

January 18, 2019

1 CS395 - Assignment 1

1.0.1 Machine Learning Project in Python Step-By-Step

https://machinelearningmastery.com/machine-learning-in-python-step-by-step/
By: Joshua Swick
Date: Jan 16, 2019

1.1 1. Downloading, Installing, and Starting Python SciPy

1.1.1 1.2 Start Python and Check Versions

```
In [3]: # Python version
        import sys
        print('Python: {}'.format(sys.version))
        # scipy
        import scipy
        print('scipy: {}'.format(scipy.__version__))
        # numpy
        import numpy
        print('numpy: {}'.format(numpy.__version__))
        # matplotlib
        import matplotlib
        print('matplotlib: {}'.format(matplotlib.__version__))
        # pandas
        import pandas
        print('pandas: {}'.format(pandas.__version__))
        # scikit-learn
        import sklearn
        print('sklearn: {}'.format(sklearn.__version__))
```

```
Python: 3.6.7 (default, Oct 22 2018, 11:32:17)
[GCC 8.2.0]
scipy: 1.2.0
numpy: 1.16.0
matplotlib: 3.0.2
pandas: 0.23.4
sklearn: 0.20.2
```

1.2 2. Load The Data

1.2.1 2.1 Import Libraries

```
In [4]: # Load libraries
        import pandas
        from pandas.plotting import scatter_matrix
        import matplotlib.pyplot as plt
        from sklearn import model_selection
        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
```

1.2.2 2.2 Load Dataset

```
In [9]: # Load dataset
        url = "iris.csv"
        names = ['sepal-length', 'sepal-width', 'petal-length', 'petal-width', 'class']
        dataset = pandas.read_csv(url, names=names)
```

1.3 3. Summarize the Dataset

1.3.1 3.1 Dimensions of Dataset

```
In [10]: # shape
         print(dataset.shape)
```

```
(150, 5)
```

1.3.2 3.2 Peek at the Data

```
In [13]: # 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
```

1.3.3 3.3 Statistical Summary

```
In [14]: # descriptions
         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
```

1.3.4 3.4 Class Distribution

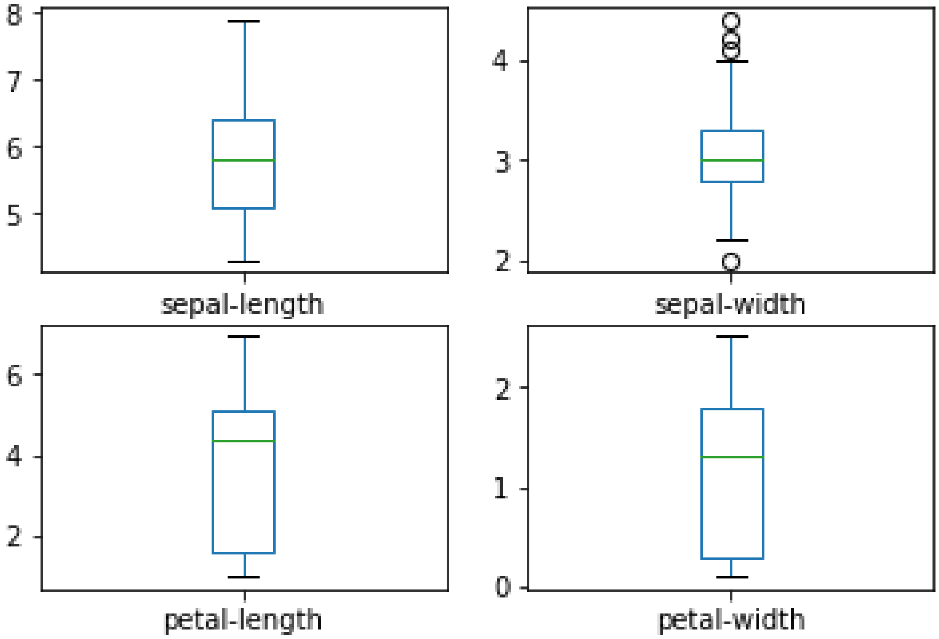
```
In [15]: # class distribution
         print(dataset.groupby('class').size())
```

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

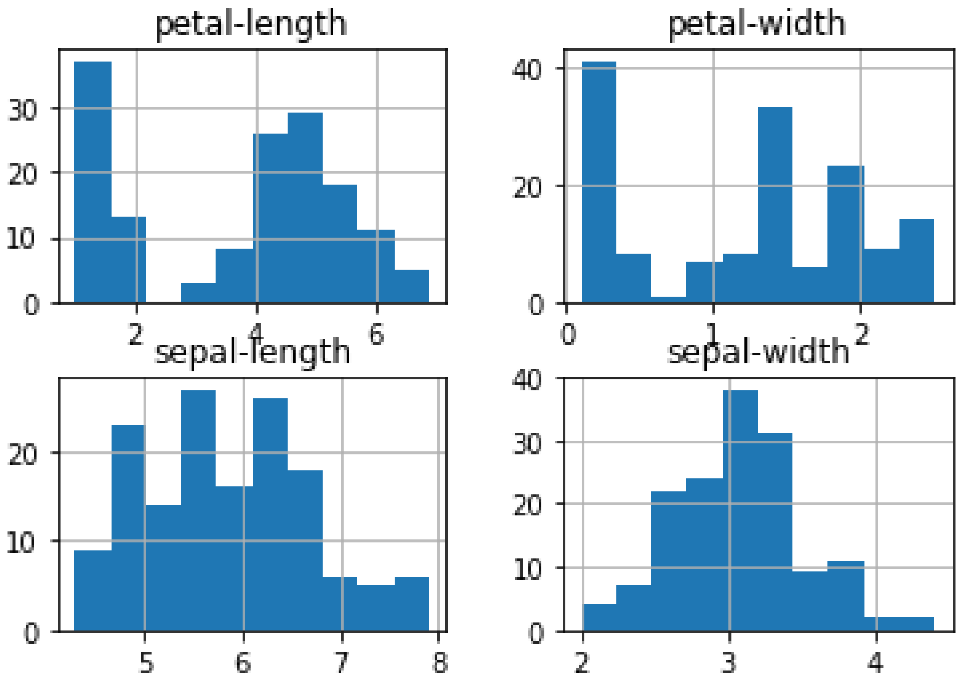
1.4 4. Data Visualization

1.4.1 4.1 Univariate Plots

```
In [17]: # box and whisker plots
         dataset.plot(kind='box', subplots=True, layout=(2,2), sharex=False, sharey=False)
         plt.show()
```

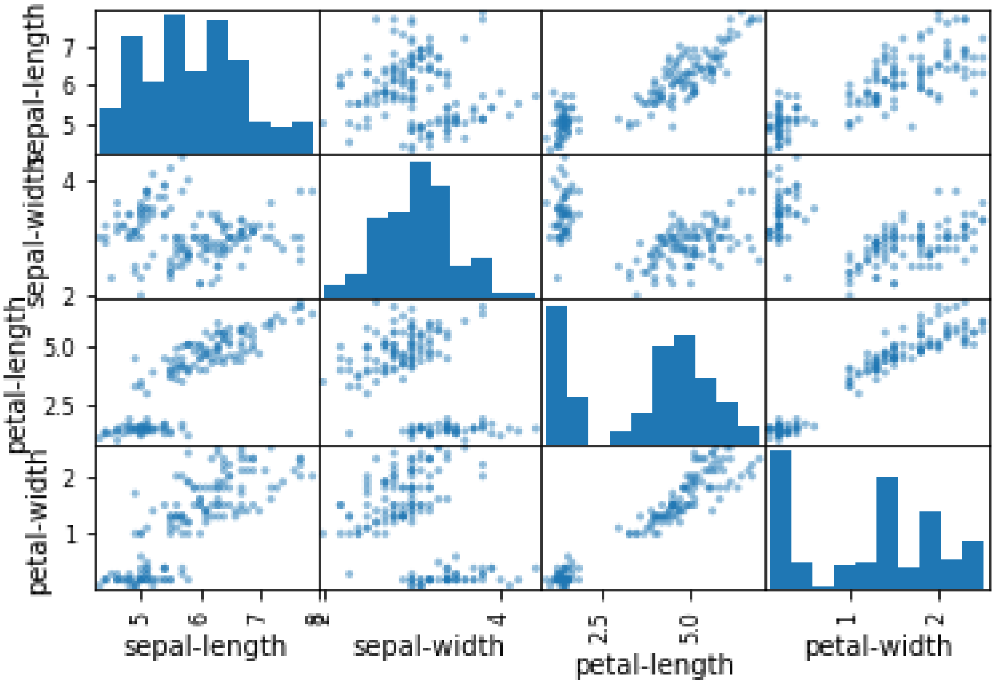


```
In [18]: # histograms
         dataset.hist()
         plt.show()
```



1.4.2 4.2 Multivariate Plots

```
In [19]: # scatter plot matrix
         scatter_matrix(dataset)
         plt.show()
```



1.5 5. Evaluate Some Algorithms

1.5.1 5.1 Create a Validation Dataset

```
In [20]: # 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 = model_selection.train_test_split(X, Y,
```

1.5.2 5.2 Test Harness

```
In [21]: # Test options and evaluation metric
         seed = 7
         scoring = 'accuracy'
```

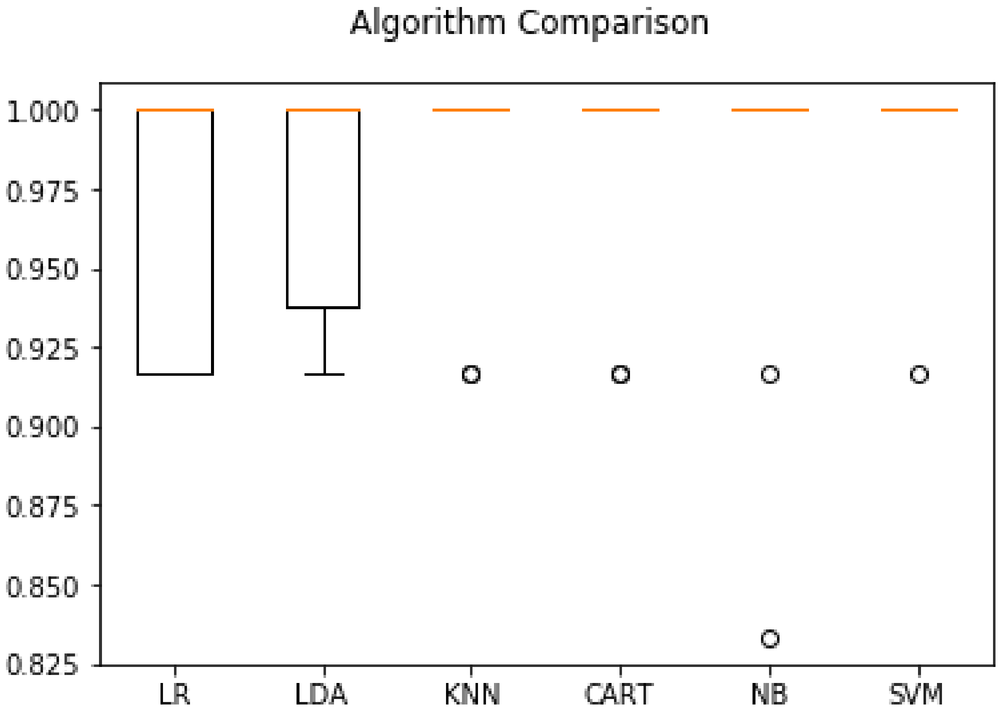
1.5.3 5.3 Build Models

```
In [33]: # Ignore warnings
         import warnings
         warnings.filterwarnings("ignore", category=FutureWarning)
         # 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()))
         # evaluate each model in turn
         results = []
         names = []
         for name, model in models:
             kfold = model_selection.KFold(n_splits=10, random_state=seed)
             cv_results = model_selection.cross_val_score(model, X_train, Y_train, cv=kfold, sc
             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)
```

1.5.4 5.4 Select Best Model

```
In [34]: # Compare Algorithms
         fig = plt.figure()
         fig.suptitle('Algorithm Comparison')
         ax = fig.add_subplot(111)
         plt.boxplot(results)
         ax.set_xticklabels(names)
         plt.show()
```



1.6 6. Make Predictions

```
In [38]: # 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(Y_validation, predictions))
         print(classification_report(Y_validation, predictions))
```

```
0.9
[[ 7  0  0]
 [ 0 11  1]
 [ 0  2  9]]
                 precision    recall  f1-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
```