Internet of Things

Programming

WEEK-10 Tutorial

Learning from data

Swinburne University of Technology

May 2023

Background

- In this tutorial, you learn how to create a python environment for machine learning using a very popular numerical dataset from UCI.
- Load the data.
- Summarize the data.
- Visualize the data.
- Models.
- Make Predictions(if time permits)

Create a python environment

- Run following commands on Debian to install following libraries:
  - **scipy** : sudo apt-get install python3-scipy
  - **numpy**: sudo apt-get install python3-numpy
  - **matplotlib**: sudo apt-get install python3-matplotlib
  - **pandas**: sudo apt-get install python3-pandas
  - **sklearn**: sudo apt-get install python3-sklearn

Note: I assume that you are using python version 3.x

Load the data

- You can use IRIS dataset from UCI repository.
- The dataset has 150 rows(instances)
- There are four attributes(four columns) of flowers.
- Fifth attribute(or column) is for class i.e. type of flowers (three in the dataset).
- You load the dataset in .csv file from URL.(Please see code script)

URL for dataset: https://raw.githubusercontent.com/jbrownlee/Datasets/master/iris.csv.

setosa

versicolor

virginica

https://en.wikipedia.org/wiki/Iris_flower_data_set

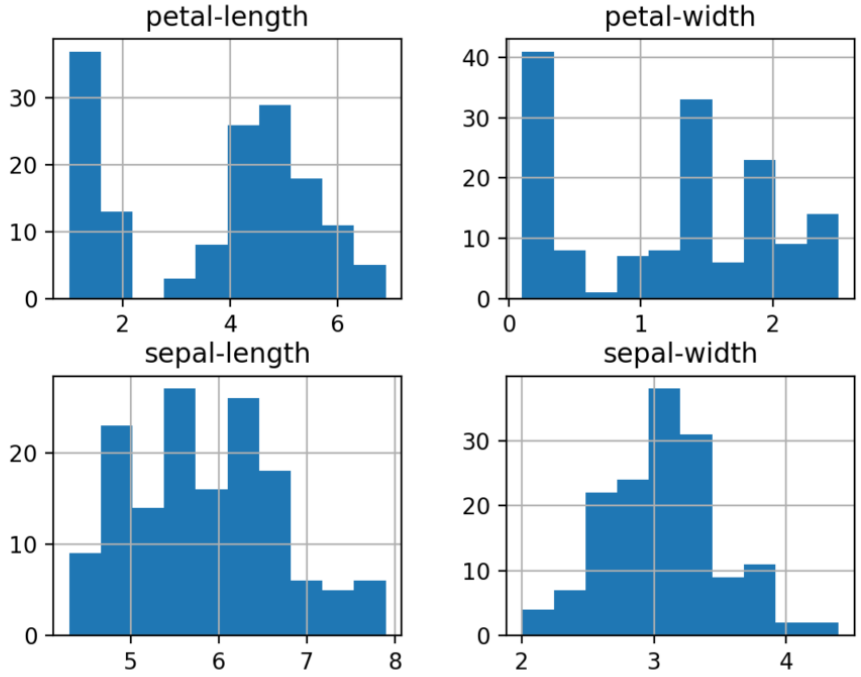
Summarise the dataset

- Let’s try to summarise the dataset:
  - Dimensions of the dataset: number of instances, attributes, classes etc.
  - Statistical summary of the dataset:
    - Mean
    - Std
    - Min
    - Max
    - percentile

Data visualization

- Univariate: To plot and understand the variation of each input attribute(variable)

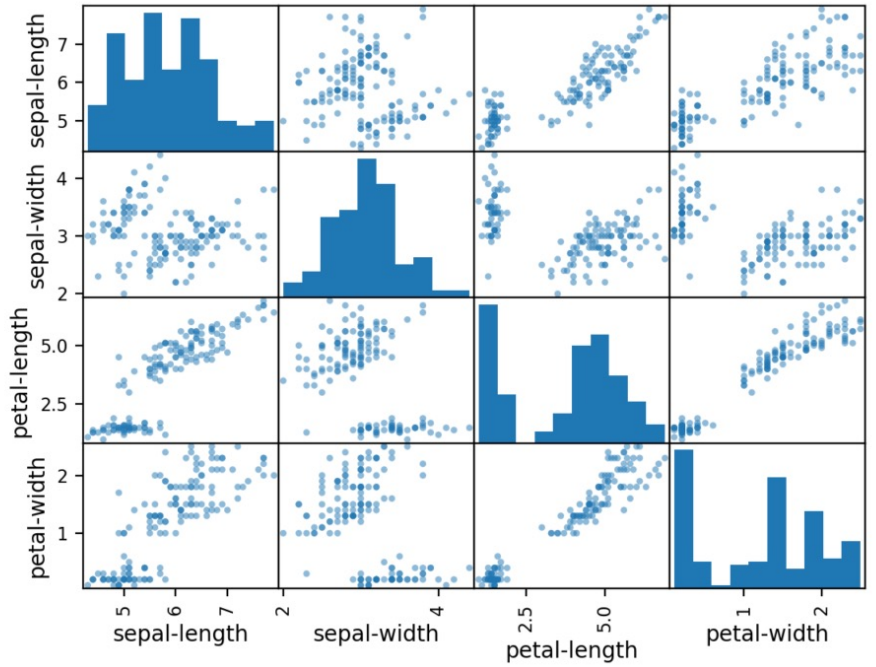
#Histogram plots
dataset.hist()
pyplot.show()



Data visualization

- Multivariate:
  - To visualize and understand the relationships between attributes:

```
#scatter plot matrix
scatter_matrix(dataset)
pyplot.show()
```



Models

- Separate out a validation dataset.
- 10-fold cross validation.
- Build models to predict the species of flowers from flower measurement.
  - Logistic Regression.(Linear model)
  - Decision Tree Classifier (non-linear model)
  - KNN Classifier(non-linear model)
  - SVM(non-linear model)
- Select the best model.

Python code

Import libraries in python script

```
# Import libraries
from pandas import read_csv
from matplotlib import pyplot
from sklearn.model_selection import train_test_split
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import StratifiedKFold
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
```

Load the dataset

```
# Load dataset
url = "https://raw.githubusercontent.com/jbrownlee/Datasets/master/iris.csv"
names = ['sepal-length', 'sepal-width', 'petal-length', 'petal-width', 'class']
dataset = read_csv(url, names=names)
```

Split out validation dataset

```
# Split-out validation dataset
array = dataset.values
X = array[:,0:4]
y= array[:,4]
X_train, X_validation, Y_train, Y_validation = train_test_split(X, y, test_size=0.20, random_state=1, shuffle=True)
```

Build models

```
# models
models = []
models.append(('LR', LogisticRegression(solver='liblinear', multi_class='ovr')))
models.append(('KNN', KNeighborsClassifier()))
models.append(('CART', DecisionTreeClassifier()))
models.append(('SVM', SVC(gamma='auto')))
```

Select the best model

```
# evaluate each model in turn
results = []
names = []
for name, model in models:
        kfold = StratifiedKFold(n_splits=10, random_state=1)
        cv_results = cross_val_score(model, X_train, Y_train, cv=kfold, scoring='accuracy')
        results.append(cv_results)
        names.append(name)
        print('%s: %f (%f)' % (name, cv_results.mean(), cv_results.std()))
# Compare Algorithms
pyplot.boxplot(results, labels=names)
pyplot.title('Algorithm Comparison')
pyplot.show()
```

References

- https://machinelearningmastery.com/machine-learning-in-python-step-by-step/