

Start Machine learning programming in 5 simple steps

By Renjith M P

<https://www.linkedin.com/in/renjith-m-p-bbb67860/>

To start with machine learning, we need to follow five basic steps.

Steps

1. Choose a use case / problem statement :- Define your objective
2. Prepare data to train the system :- for any machine learning project, first you need to train the system with some data
3. Choose a programming language and useful libraries for machine learning :- Yes, obviously you need to choose a programming language to implement your machine learning
4. Training and prediction implementation :- Implement your solution using the programming language that you have selected
5. Evaluate the result accuracy :- validate the results (Based on accuracy results, we could accept the model or we could fine tune the model with various parameters and improve the model until we get a satisfactory result)




Warning:

Target Audience : Basic knowledge on python (execute python scripts, install packages etc) is mandatory to follow this course.

Lets get into action. We will choose a use case and implement the machine learning for the same.

1. Choose a use case / problem statement

Use case : Predict the species of iris flower based on the lengths and widths of sepals and petals .

<i>Iris setosa</i>	<i>Iris versicolor</i>	<i>Iris virginica</i>
		

2. Prepare data to train the system

We will be using iris flower data set (https://en.wikipedia.org/wiki/Iris_flower_data_set) which consist of 150 rows. Each row will have 5 columns

1. sepal length
2. sepal width
3. petal length
4. petal width
5. species of iris plant

out of 150 rows, only 120 rows will be used to train the model and rest will be used to validate the accuracy of predictions.

3. Choose a programming language and libraries for machine learning

There are quite few options available however the famous once are R & Python. My choice is Python. Unlike R, Python is a complete language and platform that you can use for both research and development and to develop production systems

Ecosystem & Libraries

Machine learning needs plenty of numeric computations, data mining, algorithms and plotting.

Python offers a few ecosystems and libraries for multiple functionalities. One of the commonly used ecosystem is SciPy, which is a collection of open source software for scientific computing in Python, which has many packages or libraries.

Out of that please find the list of packages from SciPy ecosystem, that we are going to use

Package	Description
NumPy	The fundamental package for numerical computation. It defines the numerical array and matrix types and basic operations on them.
Matplotlib	a mature and popular plotting package, that provides publication-quality 2D plotting as well as rudimentary 3D plotting
SciPy Library	One of the components of the SciPy stack, providing many numerical routines
Pandas	Providing high-performance, easy to use data structures
sklearn	Simple and efficient tools for data mining and data analysis Accessible to everybody, and reusable in various contexts Built on NumPy, SciPy, and matplotlib

4. Training, Prediction and validation implementation

4.1. Import libraries (before importing make sure you install them using pip/pip3)

```
import pandas
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
```

```
# Load dataset
url =
"https://raw.githubusercontent.com/renjithmp/machinelearning/master/python/usecases/1\_irisflowers/iris.csv"
names = ['sepal-length', 'sepal-width', 'petal-length',
'petal-width', 'class']
dataset = pandas.read_csv(url, names=names)

#print important information about dataset
print(dataset.shape)
print (dataset.head(20))
print (dataset.describe())
print(dataset.groupby('class').size())

#visualize the data

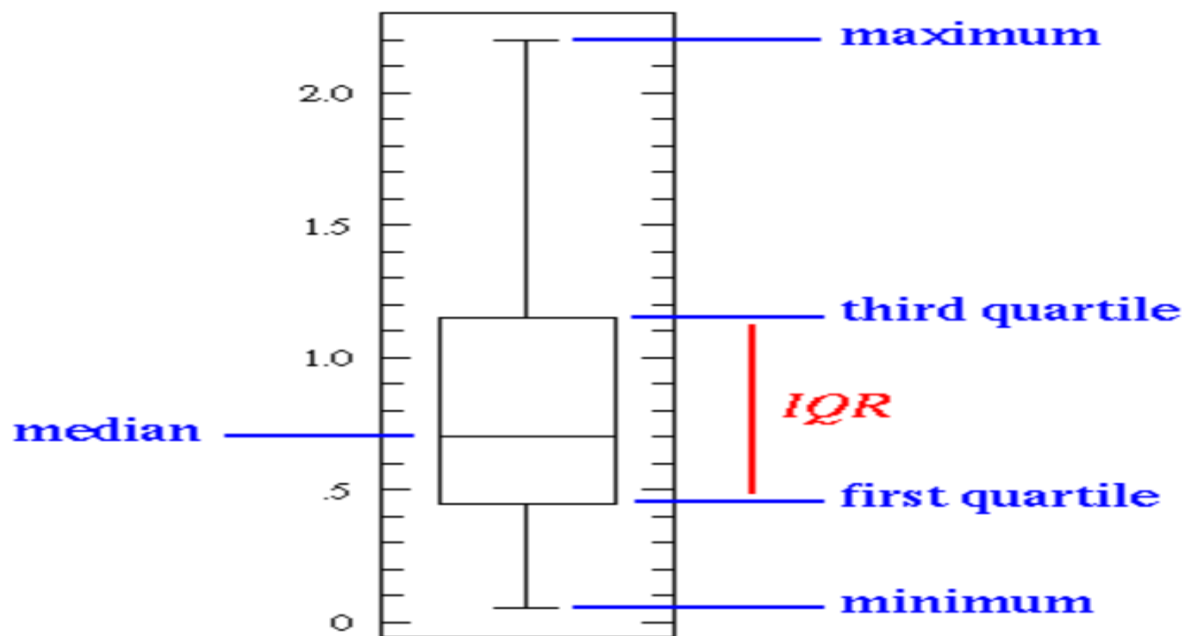
dataset.plot(kind='box', subplots=True, layout=(2,2),
sharex=False, sharey=False)

4.2. Load data to train the model
plt.show()
```

Explanation :

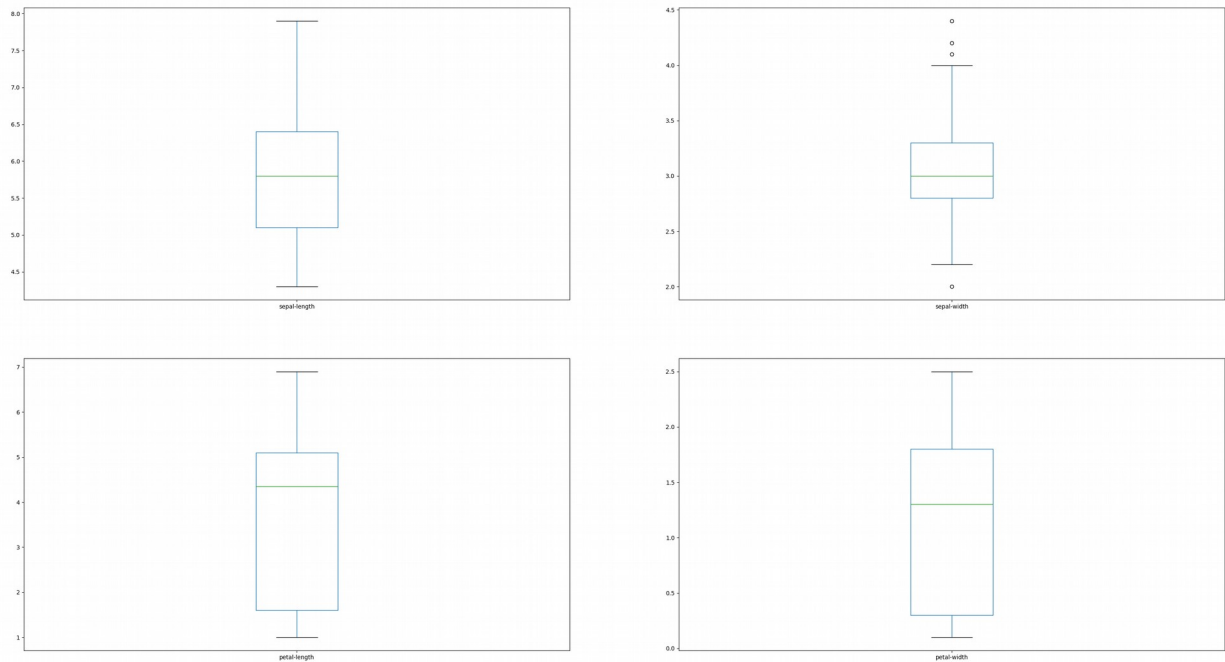
dataset.plot() function :-

The box plot (a.k.a. box and whisker diagram) is a standardized way of displaying the distribution of data based on the five number summary: minimum, first quartile, median, third quartile, and maximum. In the simplest box plot the central rectangle spans the first quartile to the third quartile (the interquartile range or IQR). A segment inside the rectangle shows the median and "whiskers" above and below the box show the locations of the minimum and maximum.



In machine learning, it is important to analyse the data using different parameters.

Visualize them using plot methods makes it much easier than analyzing data in tabular format. For our use case, we will get below plots for sepal, petal length's and widths.



4.3. split the data for training and validation

```
array=dataset.values
X=array[:,0:4]
Y=array[:,4]
validation_size=0.20
seed=7
scoring='accuracy'
X_train,X_validation,Y_train,Y_validation=model_selection.train_t
est_split(X,Y,test_size=validation_size,random_state=seed)
```

Explanation :-

X_train – training data (120 rows consist of petal ,sepal lengths and widths)

Y_train – training data (120 rows consist of class of plant)

x_validate – validation data (30 rows consist of petal,sepal lengths and widths)

Y_train -validation data(30 rows consist of class of plant)

4.4. Train few models using training data

Lets use X_train and Y_train to train few models

```
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()))
```

The explanation of algorithms can be found @ scikit-learn.org e.g

https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

I am not covering them here as it need a much deeper explanation. For now, we need to keep in mind that a model is something that has the capability to learn by it self using the training data and predict the output for future use cases

```
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
    v=kfold,scoring=scoring)
    results.append(cv_results)
    names.append(name)
    msg="%s: %f (%f)" % (name,cv_results.mean(),cv_results.std())
    print(msg)
```

Explanation :

Kfold :- it is a very useful function to divide and shuffle the data in dataset.

Here we are dividing the data in to 10 equal parts.

Cross_val_score :- This is the most important step. We are feeding the model with training data (X_train -input and Y_train -corresponding output). The method will execute the model and provide accuracy for each of the fold (remember we used 10 folds)

take the mean and std deviation of 10 fold's to see the accuracy for the entire training set.

4.5. Choose the best model which seems to be more accurate

As you can see, we have executed 5 different models for the training data (5 different algorithms) and results shows that (cv_results.mean())

KneighborsClassifier() gives the most accurate results (0.98 or 98 %)

4.6.Predict and validate the results using validation data set

```
knn=KNeighborsClassifier()
knn.fit(X_train,Y_train)
predictions=knn.predict(X_validation)
print(accuracy_score(Y_validation,predictions))
```

Lets choose KNN and find predict the output for validation data

5. Publish results

The `accuracy_score()` function can be used to see the accuracy of the prediction. In our use case we can see an accuracy of 0.90 (90%)

You can find the source code here

https://github.com/renjithmp/machinelearning/blob/master/python/usecases/1_irisflowers/flowerclassprediction.py

Reference

Jason Brownlee article

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

Scikit

<https://scikit-learn.org>