Machine learning

Machine learning is the scientific study of algorithms and statistical models that computer systems use to perform a specific task without using explicit instructions, relying on patterns and inference instead. It is seen as a subset of artificial intelligence

**Machine learning** is an application of artificial **intelligence** (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. **Machine learning** focuses on the development of computer programs that can access data and use it learn for themselves.

**Machine learning methods**

Machine learning algorithms are often categorized as supervised or unsupervised.

* **Supervised machine learning algorithms**can apply what has been learned in the past to new data using labeled examples to predict future events.

Starting from the analysis of a known training dataset, the learning algorithm produces an inferred function to make predictions about the output values.

The system is able to provide targets for any new input after sufficient training. The learning algorithm can also compare its output with the correct, intended output and find errors in order to modify the model accordingly

* In contrast, **unsupervised machine learning algorithms**are used when the information used to train is neither classified nor labeled. Unsupervised learning studies how systems can infer a function to describe a hidden structure from unlabeled data. The system doesn’t figure out the right output, but it explores the data and can draw inferences from datasets to describe hidden structures from unlabeled data.
* **Semi-supervised machine learning algorithms** fall somewhere in between supervised and unsupervised learning, since they use both labeled and unlabeled data for training – typically a small amount of labeled data and a large amount of unlabeled data.

The systems that use this method are able to considerably improve learning accuracy. Usually, semi-supervised learning is chosen when the acquired labeled data requires skilled and relevant resources in order to train it / learn from it. Otherwise, acquiringunlabeled data generally doesn’t require additional resources.

* **Reinforcement machine learning algorithms**is a learning method that interacts with its environment by producing actions and discovers errors or rewards.

Trial and error search and delayed reward are the most relevant characteristics of reinforcement learning. This method allows machines and software agents to automatically determine the ideal behavior within a specific context in order to maximize its performance.

Simple reward feedback is required for the agent to learn which action is best; this is known as the reinforcement signal.

There are 5 key libraries that you will need to install. Below is a list of the Python SciPy libraries required for this tutorial:

* scipy
* numpy
* matplotlib
* pandas
* sklearn

# 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

# Load dataset

url = "https://raw.githubusercontent.com/jbrownlee/Datasets/master/iris.csv"

names = ['sepal-length', 'sepal-width', 'petal-length', 'petal-width', 'class']

dataset = pandas.read\_csv(url, names=names)

# shape

print(dataset.shape)

# head

print(dataset.head(20))

|  |  |
| --- | --- |
| 2 | # descriptions  print(dataset.describe()) |

# class distribution

print(dataset.groupby('class').size())

|  |  |
| --- | --- |
| 1  2  3 | # box and whisker plots  dataset.plot(kind='box', subplots=True, layout=(2,2), sharex=False, sharey=False)  plt.show() |

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, test\_size=validation\_size, random\_state=seed)

seed = 7

scoring = 'accuracy'

### Build Models

* Logistic Regression (LR)
* Linear Discriminant Analysis (LDA)
* K-Nearest Neighbors (KNN).
* Classification and Regression Trees (CART).
* Gaussian Naive Bayes (NB).
* Support Vector Machines (SVM).

|  |  |
| --- | --- |
| 5  6  7  8  9  10  11  12  13  14  15  16  17  18 | # Spot Check Algorithms  models = []  models.append(('LR', LogisticRegression(solver='liblinear', multi\_class='ovr')))  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 = model\_selection.KFold(n\_splits=10, random\_state=seed)  cv\_results = model\_selection.cross\_val\_score(model, X\_train, Y\_train, cv=kfold, scoring=scoring)  results.append(cv\_results)  names.append(name)  msg = "%s: %f (%f)" % (name, cv\_results.mean(), cv\_results.std())  print(msg) |

# Compare Algorithms

fig = plt.figure()

fig.suptitle('Algorithm Comparison')

ax = fig.add\_subplot(111)

plt.boxplot(results)

ax.set\_xticklabels(names)

plt.show()

## Make Predictions

# 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))