Scikit-Learn(Cyber-security) Cheat Sheet

by satwik dondapati (sati) via cheatography.com/121228/cs/22124/

Definition

Scikit-learn is an open source Python library that implements a range of machine learning, preprocessing, cross-validation and visualization algorithms using a unified interface

Splitting Data

```
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=7)
```

Handling Missing Data

```
from sklearn.impute import SimpleImputer
```

```
missingvalues = SimpleImputer(missing_values = np.nan, strategy = 'mean')
missingvalues = missingvalues.fit(X[:, 1:3])
X[:, 1:3]=missingvalues.transform(X[:, 1:3])
```

Linear Regression

```
from sklearn.linear_model import LinearRegression
```

```
linear_reg = LinearRegression()
linear_reg.fit( X , y )
```

Decision Tree and Random forest

```
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
```

```
regressor = DecisionTreeRegressor(random_state = 0)
regressor.fit(X,y)
```

```
regressor2 = RandomForestRegressor(n_estimators = 100,random_state=0)
regressor2.fit(X,y)
```

Cross-Validation

```
from sklearn.datasets import make_regression
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import cross_validate
```

```
X , y = make)regression(n_samples = 1000, random_state = 0)
lr = LinearRegression()
```

```
result = cross_validate(lr,X,y)
result['test_score']
```

It is used to know the effectiveness of our Models by re-sampling and applying to models in different iterations.

Pandas functions for importing Data

pd.read_csv(filename)	From a CSV file
pd.read_excel(filename)	From an Excel file
pd.read_sql(query, connection_object)	Read from a SQL table/database
pd.read_clipboard()	Takes the contents of your clipboard and passes it to read_table()

Visualization using Scikit-learn

from sklearn.metrics import plot_roc_curve	Importing "plot_roc_curve" to plot
svc_disp = plot_roc_curve(svc, X_test, y_test)	Plotting **Receiver operating characteristic** Curve
metrics.plot_confusion_matrix	Plotting **Confusion Matrix**.

Clustering metrics

Adjusted Rand Index

```
>>> from sklearn.metrics import adjusted_rand_score
>>> adjusted_rand_score(y_true, y_pred)
```

Homogeneity

```
>>> from sklearn.metrics import homogeneity_score >>> homogeneity_score(y_true, y_pred)
```

V-measure

```
>>> from sklearn.metrics import v_measure_score
>>> metrics.v_measure_score(y_true, y_pred)
```

Pandas Data Cleaning functions

pd.isnull()	Checks for null Values, Returns Boolean Arrray
pd.notnull()	Opposite of pd.isnull()
df.dropna()	Drop all rows that contain null values
df.dropna(axis=1)	Drop all columns that contain null values
df.fillna(x)	Replace all null values with x

Numpy Basic Functions

import numpy as np	importing numpy
example = [0,1,2] example = np.array(example)	array([0, 1, 2])
np.arange(1,4)	array([1,2,3])
np.zeros(2,2)	array([[0,0],[0,0]])

By **satwik dondapati** (sati)
cheatography.com/sati/

Published 22nd March, 2020.
Last updated 22nd March, 2020.

Sponsored by **Readable.com**
Measure your website readability!
https://readable.com

Scikit-Learn(Cyber-security) Cheat Sheet

by satwik dondapati (sati) via cheatography.com/121228/cs/22124/

Numpy Basic Functions (cont)

np.linspace(0,10,2)	array([0,5]), gives two evenly spaced values
np.eye(2)	array([[1,0],[0,1]), 2*2 Identity Matrix
example.reshape(3,1)	array([[0],[1],[2]])

Loading Dataset from local Machine

```
import pandas as pd
data = pd.read_csv(pathname)
```

If the file is in the local directory then we can directly use File name

Loading Data from Standard datasets

```
from sklearn import datasets
iris = datasets.load_iris()
digits = datasets.load_digits()
```

Encoding Categorical Variables

```
from sklearn.preprocessing import LabelEncoder
labelencoder_X = LabelEncoder()
X[ : , 0] = labelencoder_X.fit_transform(X[ : , 0 ])
onehotencoder = OneHotEncoder(categorical_features = [0])
X = onehotencoder.fit_transform(X).toarray()
```

Polynomial Regression

```
from sklearn.preprocessing import PolynomialFeatures
poly_reg = PolynomialFeatures(degree =2)
X_poly = poly_reg.fit_transform(X)
```

It not only checks the relation between X(independent) and y(dependent). But also checks with X^2 ..X^n. (n is degree specified by us).

Evaluation of Regression Model Performance

R^2 = 1 - SS(residuals)/SS(total)

SS(res) = SUM(Yi - $\hat{y}i$)^2

SS(Total) = SUM(yi - $yavg$)^2

```
from sklearn.metrics import r2_score
r2_score(y_true,y_pred)
```

The Greater the R^2 value the better the model is..

Converting Dataframe to Matrix

```
data = pd.read_csv("data.csv")
X = data.iloc[ : , :-1].values
y = data.iloc[ : , 3].values
```

y is Dependent parameter

Feature Scaling

```
from sklearn.preprocessing import StandardScaler
sc_X = StandardScaler()
X_train = sc_X.fit_transform(X_train)
X_test = sc_X.transform(X_test)
```

Euclidean distance is dominated by the larger numbers and to make all the values on the same scale. hence Scaling should be done. Most of the models do feature scaling by themselves.

SVR(Non-linear Regression model)

```
from sklearn.svm import SVR
regressor = SVR(kernel = 'rbf')
regressor.fit(X,y)
y_prediction = regressor.predict(values)
```

Basically, the kernel is selected based on the given problem. If the problem is Linear then **kernel='linear'**. And if problem is non-linear we can choose either 'poly' or **'rbf'(gussian)**

Some Classification Models

- Logistic Regression
- K-NN(K- nearest neighbours)
- Support Vector Machine(SVM)
- Naive Bayes
- Decision Tree Classification
- Random Forest Classification

Some Clustering Models

- K-Means Clustering
- Hierarchial Clustering
- DB-SCAN

Knowing about Data information with Pandas

df.head(n)	First n rows of the DataFrame
df.tail(n)	Last n rows of the DataFrame
df.shape	Number of rows and columns
df.info()	Index, Datatype and Memory information
df.describe()	Summary statistics for numerical columns

Sponsored by **Readable.com**
Measure your website readability!
https://readable.com