Cheatography

Scikit-learn Cheat Sheet

by lkakashv via cheatography.com/172895/cs/36333/

Linear Regression

```
# Import and create the model:
from sklear n.l ine ar_ model import Linear Reg -
ression

your_model = Linear Reg res sion()
# Fit:
your_m ode l.f it( x_t rai nin g_data, y_trai nin -
g_data)
# .coef_: contains the coeffi cients
# .inter cept_: contains the intercept
# Predict:
predic tions = your_m ode l.p red ict (yo ur_ -
x_data)
#.score(): returns the coeffi cient of determ -
ination R²
```

Naive Bayes

```
# Import and create the model:
from sklear n.n aiv e_bayes import Multin omialNB

your_model = Multin omi alNB()
# Fit:
your_m ode l.f it( x_t rai nin g_data, y_trai nin -
g_data)
# Predict:
# Returns a list of predicted classes - one
prediction for every data point
predic tions = your_m ode l.p red ict (yo ur_ -
x_data)

# For every data point, returns a list of probab -
ilities of each class
probab ilities = your_m ode l.p red ict _pr oba -
(yo ur_ x_data)
```

K-Nearest Neighbors

K-Means

```
# Import and create the model:
from sklear n.c luster import KMeans

your_model = KMeans (n_ clu ste rs=4, init=' ran -
dom')
n_clus ters: number of clusters to form and number
of centroids to generate
init: method for initia liz ation
k-means++: K-Means++ [default]
random: K-Means
random _state: the seed used by the random number
generator [optional]
# Fit:
your_m ode l.f it( x_t rai nin g_data)
# Predict:
predic tions = your_m ode l.p red ict (yo ur_ -
x_data)
```

Validating the Model

```
# Import and print accuracy, recall, precision,
and F1 score:
from sklear n.m etrics import accura cy_ score,
recall _score, precis ion _score, f1_score

print( acc ura cy_ sco re( tru e_l abels, guesses))
print( rec all _sc ore (tr ue_ labels, guesses))
print( pre cis ion _sc ore (tr ue_ labels,
guesses))
print( f1_ sco re( tru e_l abels, guesses))
# Import and print the confusion matrix:
from sklear n.m etrics import confus ion _matrix

print( con fus ion _ma tri x(t rue _la bels,
guesses))
```

Training Sets and Test Sets

```
from sklearn.model_selection import
train_test_split

x_train, x_test, y_train, y_test = train_ tes -
t_s plit(x, y, train_ siz e=0.8, test_s ize =0.2)
# train_ size: the proportion of the dataset to
include in the train split
# test_size: the proportion of the dataset to
include in the test split
# random _state: the seed used by the random
number generator [optional]
```

```
# Import and create the model:
from sklear n.n eigbors import KNeigh bor sCl ass -
ifier

your_model = KNeigh bor sCl ass ifier()
# Fit:
your_m ode l.f it( x_t rai nin g_data, y_trai nin -
g_data)
# Predict:
# Returns a list of predicted classes - one
prediction for every data point
predic tions = your_m ode l.p red ict (yo ur_ -
x_data)

# For every data point, returns a list of probab -
ilities of each class
probab ilities = your_m ode l.p red ict _pr oba -
(yo ur_ x_data)
```