# Scikit Learn:

* First you’ll import the model, the general form is:

From sklearn.family import Model

* For example:

from sklearn.linear\_model import LinearRegression

* Estimator parameters: All the parameters of an estimator can be set when it is instantiated, and have suitable default values. For example:

model = LinearRegression(normalize=True)

print(model)

LinearRegression(copy\_X=True, fit\_intercept=True, normalize=True)

* You can use Shift+Tab in jupyter to check the possible parameters.
* Once you have your model created with your parameters, it is time to fit your model on some data. But remember, we should split this data into a training and a test set.

From sklearn.cross\_validation import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3)

* After you’ve split the data, you can train/fit model on the training data. This is done through the model.fit() method:

model.fit(X\_train, y\_train)

* We get predicted values using the predict method:

predictions = model.predict(X\_test)

* We can then evaluate the model by comparing our predictions to correct values. Evaluation method depends on what sort of machine learning algorithm we are using (ex: Regression, Classification, Clustering, etc.)
* Available in all estimators:

model.fit()

For supervised learning applications model.fit(X,y)

For unsupervised learning applications model.fit(X)

* Available in supervised estimators:

model.predict() : given a trained model, predicting the label for a new set of data.

ex: model.predict(X\_test)

* Also available in supervised estimators:

model.predict\_proba() : For classification problems, some estimators provide this method which returns the probability that a new observation has each categorical label. In this case, the label with the highest probability is returned by model.predict()

* Also available in supervised estimators:

model.score() : for classification or regression problems, most estimators implement a score method. Scores are between 0 and 1, with a larger score indicating a better fit.

* Available in unsupervised estimators:

model.predict() : predict labels in clustering algorithms.

* Available in unsupervised estimators:

model.transform() : given an unsupervised model, transform new data into the new basis. This also accepts one argument X\_new, and returns the new representation of the data based on the unsupervised model.

* Also available in unsupervised estimators:

model.fit\_transform() : some estimators implement this method, which more efficiently performs a fit and a transform on the same imput data.

# Scikit Learn ML Cheat Sheet:

