# Introduction to scikit-learn and Model Building Basics

## What is scikit-learn?

* **Open-source ML library** for Python: provides *simple, efficient tools* for data modeling and analysis[[1]](https://www.geeksforgeeks.org/machine-learning/what-is-python-scikit-library/#:~:text=Scikit,regression%2C%20clustering%2C%20and%20dimensionality%20reduction).
* **Built on NumPy/SciPy**: integrates well with the Python scientific stack for fast computation[[1]](https://www.geeksforgeeks.org/machine-learning/what-is-python-scikit-library/#:~:text=Scikit,regression%2C%20clustering%2C%20and%20dimensionality%20reduction).
* **Support for many tasks**: includes algorithms for classification, regression, clustering, and more[[2]](https://www.geeksforgeeks.org/machine-learning/what-is-python-scikit-library/#:~:text=Supervised%20Learning).
* **Supervised & unsupervised learning**: supports both labeled (e.g. classification) and unlabeled (e.g. clustering) problems[[3]](https://scikit-learn.org/stable/getting_started.html#:~:text=%60Scikit,evaluation%2C%20and%20many%20other%20utilities).
* **Consistent, user-friendly API**: all models use .fit()/.predict() syntax, making it easy to try different algorithms[[4]](https://www.geeksforgeeks.org/machine-learning/what-is-python-scikit-library/#:~:text=%2A%20Inbuilt%20datasets%3A%C2%A0Scikit,ensuring%20continuous%20development%20and%20support)[[5]](https://scikit-learn.org/stable/getting_started.html#:~:text=Once%20the%20estimator%20is%20fitted%2C,train%20the%20estimator).
* **Rich utilities**: includes tools for data preprocessing, model selection, and evaluation (e.g. train\_test\_split, scaling, metrics)[[3]](https://scikit-learn.org/stable/getting_started.html#:~:text=%60Scikit,evaluation%2C%20and%20many%20other%20utilities).

## Machine Learning Workflow (scikit-learn)

* **Data Preprocessing**: clean data, handle missing values, encode categorical features, scale/normalize numeric features (e.g. using LabelEncoder, StandardScaler)[[6]](https://www.geeksforgeeks.org/machine-learning/learning-model-building-scikit-learn-python-machine-learning-library/#:~:text=Machine%20learning%20algorithms%20require%20numerical,techniques%20which%20are%20as%20follows).
* **Train/Test Split**: split dataset into a training set and a separate test (or validation) set (e.g. train\_test\_split), to evaluate performance on unseen data.
* **Model Training**: choose an estimator and call model.fit(X\_train, y\_train) to learn from the training data[[7]](https://www.geeksforgeeks.org/machine-learning/learning-model-building-scikit-learn-python-machine-learning-library/#:~:text=Step%204%3A%20Training%20the%20Model). scikit-learn models have a consistent interface for fitting.
* **Prediction**: use model.predict(X\_test) to generate predictions on new data[[8]](https://www.geeksforgeeks.org/machine-learning/learning-model-building-scikit-learn-python-machine-learning-library/#:~:text=Python%20)[[5]](https://scikit-learn.org/stable/getting_started.html#:~:text=Once%20the%20estimator%20is%20fitted%2C,train%20the%20estimator). No additional training is needed after fitting.
* **Evaluation**: compare y\_pred with true labels y\_test using metrics like accuracy, precision, recall, F1-score, or confusion matrix[[9]](https://www.geeksforgeeks.org/machine-learning/learning-model-building-scikit-learn-python-machine-learning-library/#:~:text=Step%206%3A%20Evaluating%20Model%20Accuracy). For example, accuracy\_score(y\_test, y\_pred) computes overall accuracy.

## Decision Tree Classifier

* **Concept**: A *decision tree* is a non-parametric, tree-based model that learns a set of if–then rules to split the data by feature values[[10]](https://scikit-learn.org/stable/modules/tree.html#:~:text=Decision%20Trees%20,as%20a%20piecewise%20constant%20approximation). It can handle both classification and regression.
* **Why use it**: Easy to interpret (model can be visualized), requires little data preprocessing, and works with numeric inputs[[11]](https://scikit-learn.org/stable/modules/tree.html#:~:text=Some%20advantages%20of%20decision%20trees,are)[[12]](https://scikit-learn.org/stable/modules/tree.html#:~:text=,See%20algorithms%20for%20more%20information).
* **How it works**: The tree splits data recursively on features (e.g. using Gini impurity) to maximize class purity in each branch[[10]](https://scikit-learn.org/stable/modules/tree.html#:~:text=Decision%20Trees%20,as%20a%20piecewise%20constant%20approximation).
* from sklearn.tree import DecisionTreeClassifier  
  clf = DecisionTreeClassifier(random\_state=0)  
  clf.fit(X\_train, y\_train) # Train the decision tree  
  y\_pred = clf.predict(X\_test) # Predict class labels
* *(This trains a CART decision tree and uses it to predict on test data.)*

## K-Nearest Neighbors (KNN) Classifier

* **Concept**: KNN is a *non-parametric, instance-based* method. Each new sample is classified by a majority vote of its k nearest neighbors in feature space[[13]](https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm#:~:text=Evelyn%20Fix%20%20and%20,of%20that%20single%20nearest%20neighbor). There is effectively no explicit training step (it stores the training data).
* **How it works**: For each test point, compute distances (e.g. Euclidean) to all training points, find the k closest, and assign the most common class among them[[13]](https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm#:~:text=Evelyn%20Fix%20%20and%20,of%20that%20single%20nearest%20neighbor).
* **Considerations**: Simple to implement; sensitive to feature scaling (features should be normalized). Typically choose k (e.g. 3 or 5).
* from sklearn.neighbors import KNeighborsClassifier  
  knn = KNeighborsClassifier(n\_neighbors=3)  
  knn.fit(X\_train, y\_train) # Store training data  
  y\_pred = knn.predict(X\_test) # Classify test samples by majority vote
* *(Here n\_neighbors=3 means each prediction looks at the 3 nearest training points.)*

## Logistic Regression Classifier

* **Concept**: Logistic Regression models the **probability** of the positive class by applying a logistic (sigmoid) function to a linear combination of features. It is a **linear model** used for *binary classification*[[14]](https://www.geeksforgeeks.org/machine-learning/understanding-logistic-regression/#:~:text=Logistic%20Regression%20is%20a%20supervised,regression%20and%20its%20core%20concepts).
* **How it works**: It estimates log-odds of the positive class as w·x + b, then uses the sigmoid to output a probability between 0 and 1. Predictions are made by thresholding (e.g. >0.5 → class 1, else 0)[[14]](https://www.geeksforgeeks.org/machine-learning/understanding-logistic-regression/#:~:text=Logistic%20Regression%20is%20a%20supervised,regression%20and%20its%20core%20concepts).
* **Why use it**: Performs well for linearly separable data, and outputs calibrated probabilities.
* from sklearn.linear\_model import LogisticRegression  
  lr = LogisticRegression()  
  lr.fit(X\_train, y\_train) # Train logistic regression  
  y\_pred = lr.predict(X\_test) # Predict class labels (0 or 1)
* *(The model learns weights w so that sigmoid(w·X + b) matches the training labels.)*

## Confusion Matrix (Model Evaluation)

*Example confusion matrix for a binary classifier (rows=actual class, columns=predicted class).* A confusion matrix is a table that summarizes correct versus incorrect predictions[[15]](https://pieriantraining.com/confusion-matrix-with-scikit-learn-and-python/#:~:text=A%20confusion%20matrix%20is%20a,the%20algorithm%20in%20each%20class). It shows counts of **True Positives (TP)**, **True Negatives (TN)**, **False Positives (FP)**, and **False Negatives (FN)**. From this matrix, one can compute accuracy, precision, recall, etc. For example:

from sklearn.metrics import confusion\_matrix  
cm = confusion\_matrix(y\_test, y\_pred)  
print(cm)

The output cm is a 2×2 NumPy array. In the binary case it has the form:

[[TN, FP],  
 [FN, TP]]

where the first row/column are negatives and the second are positives. Each cell shows how many samples fall into that category (correct or incorrect). This breakdown helps identify where a model is making mistakes (e.g. many false positives or false negatives)[[15]](https://pieriantraining.com/confusion-matrix-with-scikit-learn-and-python/#:~:text=A%20confusion%20matrix%20is%20a,the%20algorithm%20in%20each%20class).

[[1]](https://www.geeksforgeeks.org/machine-learning/what-is-python-scikit-library/#:~:text=Scikit,regression%2C%20clustering%2C%20and%20dimensionality%20reduction) [[2]](https://www.geeksforgeeks.org/machine-learning/what-is-python-scikit-library/#:~:text=Supervised%20Learning) [[4]](https://www.geeksforgeeks.org/machine-learning/what-is-python-scikit-library/#:~:text=%2A%20Inbuilt%20datasets%3A%C2%A0Scikit,ensuring%20continuous%20development%20and%20support) What is python scikit library? - GeeksforGeeks

<https://www.geeksforgeeks.org/machine-learning/what-is-python-scikit-library/>

[[3]](https://scikit-learn.org/stable/getting_started.html#:~:text=%60Scikit,evaluation%2C%20and%20many%20other%20utilities) [[5]](https://scikit-learn.org/stable/getting_started.html#:~:text=Once%20the%20estimator%20is%20fitted%2C,train%20the%20estimator) Getting Started — scikit-learn 1.7.0 documentation

<https://scikit-learn.org/stable/getting_started.html>

[[6]](https://www.geeksforgeeks.org/machine-learning/learning-model-building-scikit-learn-python-machine-learning-library/#:~:text=Machine%20learning%20algorithms%20require%20numerical,techniques%20which%20are%20as%20follows) [[7]](https://www.geeksforgeeks.org/machine-learning/learning-model-building-scikit-learn-python-machine-learning-library/#:~:text=Step%204%3A%20Training%20the%20Model) [[8]](https://www.geeksforgeeks.org/machine-learning/learning-model-building-scikit-learn-python-machine-learning-library/#:~:text=Python%20) [[9]](https://www.geeksforgeeks.org/machine-learning/learning-model-building-scikit-learn-python-machine-learning-library/#:~:text=Step%206%3A%20Evaluating%20Model%20Accuracy) Learning Model Building in Scikit-learn - GeeksforGeeks

<https://www.geeksforgeeks.org/machine-learning/learning-model-building-scikit-learn-python-machine-learning-library/>

[[10]](https://scikit-learn.org/stable/modules/tree.html#:~:text=Decision%20Trees%20,as%20a%20piecewise%20constant%20approximation) [[11]](https://scikit-learn.org/stable/modules/tree.html#:~:text=Some%20advantages%20of%20decision%20trees,are) [[12]](https://scikit-learn.org/stable/modules/tree.html#:~:text=,See%20algorithms%20for%20more%20information) 1.10. Decision Trees — scikit-learn 1.7.0 documentation

<https://scikit-learn.org/stable/modules/tree.html>

[[13]](https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm#:~:text=Evelyn%20Fix%20%20and%20,of%20that%20single%20nearest%20neighbor) k-nearest neighbors algorithm - Wikipedia

<https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm>

[[14]](https://www.geeksforgeeks.org/machine-learning/understanding-logistic-regression/#:~:text=Logistic%20Regression%20is%20a%20supervised,regression%20and%20its%20core%20concepts) Logistic Regression in Machine Learning - GeeksforGeeks

<https://www.geeksforgeeks.org/machine-learning/understanding-logistic-regression/>

[[15]](https://pieriantraining.com/confusion-matrix-with-scikit-learn-and-python/#:~:text=A%20confusion%20matrix%20is%20a,the%20algorithm%20in%20each%20class) Confusion Matrix with Scikit-Learn and Python - Pierian Training

<https://pieriantraining.com/confusion-matrix-with-scikit-learn-and-python/>