Scikit-learn is an open-source Python library for all kinds of predictive data analysis. You can perform classification, regression, clustering, dimensionality reduction, model tuning, and data preprocessing tasks.

### Loading the Data

**Classification**
- from sklearn import datasets
- X, y = datasets.load_wine(return_X_y=True)

**Regression**
- from sklearn import datasets
diabetes = datasets.load_diabetes()
- X, y = diabetes.data, diabetes.target

### Training And Test Data

**Classification**
- from sklearn.model_selection import train_test_split
- X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=0)

**Regression**

### Preprocessing the Data

**Standardization**
- from sklearn.preprocessing import StandardScaler
- scaler = StandardScaler()
- scaled_X_train = scaler.fit_transform(X_train)
- scaled_X_test = scaler.transform(X_test)

**Normalization**
- from sklearn.preprocessing import Normalizer
- norm = Normalizer()
- norm_X_train = norm.fit_transform(X_train)
- norm_X_test = norm.transform(X_test)

**Binarization**
- from sklearn.preprocessing import Binarizer
- binary = Binarizer(threshold=0.0)
- binary_X_train = binary.fit_transform(X_train)

**Encoding Categorical Features**
- from sklearn.preprocessing import LabelEncoder
- lab_enc = LabelEncoder()
- y = lab_enc.fit_transform(y)

**Imputer**
- from sklearn.impute import SimpleImputer
- imp_mean = SimpleImputer(missing_values=0, strategy='mean')
- imp_mean.fit_transform(X_train)

### Supervised Learning Model

**Linear Regression**
- from sklearn.linear_model import LogisticRegression
- lr = LogisticRegression()

**Support Vector Machines**
- from sklearn.svm import SVC
- svm_svc = SVC(kernel='linear')

**Naive Bayes**
- from sklearn.naive_bayes import GaussianNB
- gnb = GaussianNB()

### Unsupervised Learning Model

**Principal Component Analysis**
- from sklearn.decomposition import PCA
- pca = PCA(n_components=2)

**K Means**
- from sklearn.cluster import KMeans
- kmeans = KMeans(n_clusters=5, random_state=0)

### Model Fitting

**Supervised Learning**
- lr.fit(X_train, y_train)
- svm_svc.fit(X_train, y_train)

**Unsupervised Learning**
- model = pca.fit_transform(X_train)
- kmeans.fit(X_train)

### Prediction

**Supervised Learning**
- y_pred = lr.predict_proba(X_test)
- y_pred = svm_svc.predict(X_test)

**Unsupervised Learning**
- y_pred = kmeans.predict(X_test)

### Evaluation

**Accuracy Score**
- from sklearn.metrics import accuracy_score
- accuracy_score(y_test, y_pred)

**Classification Report**
- from sklearn.metrics import classification_report
- print(classification_report(y_test, y_pred))

**Mean Squared Error**
- from sklearn.metrics import mean_squared_error
- mean_squared_error(y_test, y_pred)

**R2 Score**
- from sklearn.metrics import r2_score
- r2_score(y_test, y_pred)

**Adjusted Rand Index**
- from sklearn.metrics import adjusted_rand_score
- adjusted_rand_score(y_test, y_pred)

### Cross-Validation

**from sklearn.model_selection import cross_val_score**

**from sklearn.model_selection import GridSearchCV**

**Model Tuning**
- parameters = {'kernel':('linear', 'rbf'), 'C':[1, 10]}
- model = GridSearchCV(svm_svc, parameters)
- model.fit(X_train, y_train)
- print(model.best_score_)
- print(model.best_estimator_)

### Subscribe to KDnuggets News

Visit KDnuggets.com for more cheatsheets and additional learning resources.