Data - Feature Generation

Module	Item	What it does	Example
sklearn.datasets	load_iris, load_wine, fetch_california_housing	Load toy or real datasets from the library (or fetch from the web)	X, y = load_iris(return_X_y=True)
	make_classification, make_regression	Programmatically create synthetic classification/regression data	X, y = make_classification(n_samples=200)
	make_blobs, make_moons	Create simple toy clusters / decision boundaries	X, y = make_blobs(n_samples=300)
	make_circles	Generates a circular decision boundary	X, y = make_circles(noise=0.05)

Preprocessing - Feature Engineering

Module	Item	What it does	Example
sklearn.preprocessing	StandardScaler, MinMaxScaler, RobustScaler	Scale/normalize numeric features	X_scaled = StandardScaler().fit_transform(X)
	Normalizer	Scale each sample to unit norm	X_norm = Normalizer().fit_transform(X)
	LabelEncoder, OneHotEncoder	Encode categorical labels (ordinal or one-hot)	y_enc = OneHotEncoder().fit_transform(y.reshape(-1,1))
	PolynomialFeatures	Generate interaction / polynomial terms	X_poly = PolynomialFeatures(degree=2).fit_transform(X)
	Binarizer	Threshold values to 0/1	X_bin = Binarizer(threshold=0.5).fit_transform(X)
	PowerTransformer (yeo_johnson, box_cox)	Stabilize variance / normalize distributions	X_pt = PowerTransformer().fit_transform(X)
	FeatureUnion / ColumnTransformer	Combine several transformers on different columns	preprocess = ColumnTransformer([('num', StandardScaler(), num_cols), ('cat', OneHotEncoder(), cat_cols)])

Feature Selection - Extraction

Module	Item	What it does	Example
sklearn.feature_selection	SelectKBest, RFE, RecursiveFeatureEliminationCV	Select top-k or recursively eliminate features based on a score	selector = SelectKBest(score_func=chi2, k=10).fit(X, y)
	VarianceThreshold	Remove features with low variance	vt = VarianceThreshold(threshold=0.01).fit(X)
	SelectFromModel	Select features based on importance from a model (e.g., tree)	sfm = SelectFromModel(RandomForestClassifier()).fit(X, y)
	PCA, IncrementalPCA (in decomposition)	Principal Component Analysis for dimensionality reduction	pca = PCA(n_components=5).fit_transform(X)
	TruncatedSVD	SVD for sparse matrices (e.g., TF-IDF)	svd = TruncatedSVD(n_components=100).fit_transform(X)
	KernelPCA	Non-linear PCA via kernels	kpca = KernelPCA(kernel='rbf').fit_transform(X)

Model Selection - Hyperparameter

Module	Item	What it does	Example
sklearn.model_selection	train_test_split	Split data into train / test (or validation) sets	X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
	cross_val_score, cross_validate	Compute cross-validated scores for a model	scores = cross_val_score(LogisticRegression(), X, y, cv=5)
	GridSearchCV, RandomizedSearchCV	Exhaustive / random search over hyper-parameter grid	grid = GridSearchCV(LogisticRegression(), param_grid, cv=5).fit(X, y)
	ShuffleSplit, StratifiedKFold	Custom split strategies for CV	cv = StratifiedKFold(n_splits=5)
	ParameterSampler (used by RandomizedSearchCV)	Sample parameters randomly from distributions	
	validation_curve	Plot training / validation scores vs. a parameter	

Pipelines - Composition

Module	Item	What it does	Example
sklearn.pipeline	Pipeline	Chain transformers + estimator into a single object	pipe = Pipeline([('scaler', StandardScaler()), ('clf', LogisticRegression())])
	FeatureUnion	Combine multiple feature extraction pipelines in parallel	union = FeatureUnion([('pca', PCA()), ('poly', PolynomialFeatures())])
	TransformedTargetRegressor	Apply a transform to the target variable (e.g., log-transform)	reg = TransformedTargetRegressor(regressor=LinearRegression(), transformer=np.log1p)
	make_pipeline, make_union	Short-hand constructors for pipelines / unions	pipe = make_pipeline(StandardScaler(), LogisticRegression())

Supervised Learning - Classific

Module	Item	What it does	Example
sklearn.linear_model	LogisticRegression, SGDClassifier	Linear models for classification (log-reg, perceptron)	clf = LogisticRegression(max_iter=200).fit(X, y)
	LinearDiscriminantAnalysis	LDA for dimensionality reduction + classification	
	QuadraticDiscriminantAnalysis	QDA (quadratic decision boundaries)	
sklearn.svm	SVC, NuSVC, LinearSVC	Support Vector Machines (kernelized or linear)	svc = SVC(kernel='rbf', C=1.0).fit(X, y)
sklearn.tree	DecisionTreeClassifier	CART decision tree	
sklearn.ensemble	RandomForestClassifier, GradientBoostingClassifier, AdaBoostClassifier, BaggingClassifier	Ensemble of trees / boosting	
	VotingClassifier, StackingClassifier	Meta-learning by combining models	
sklearn.neighbors	KNeighborsClassifier	Instance-based nearest-neighbor	
sklearn.naive_bayes	GaussianNB, MultinomialNB, BernoulliNB	Naïve Bayes variants	
sklearn.neural_network	MLPClassifier	Feed-forward neural network	
sklearn.gaussian_process	GaussianProcessClassifier	Probabilistic classification with GPs	

Supervised Learning - Regression

Module	Item	What it does	Example
sklearn.linear_model	LinearRegression, Ridge, Lasso, ElasticNet, BayesianRidge	Linear models with various regularization	
	PolynomialFeatures (combined with linear model)	Fit polynomial regression	
sklearn.svm	SVR, NuSVR, LinearSVR	Support Vector Regression	
sklearn.tree	DecisionTreeRegressor	CART for regression	
sklearn.ensemble	RandomForestRegressor, GradientBoostingRegressor, AdaBoostRegressor, BaggingRegressor	Ensemble tree regressors	
	VotingRegressor, StackingRegressor	Combine multiple regressors	
sklearn.neighbors	KNeighborsRegressor	Instance-based regression	
sklearn.neural_network	MLPRegressor	Neural network regressor	
sklearn.gaussian_process	GaussianProcessRegressor	Probabilistic regression with GPs	

Clustering - Density Estimation

Module	Item	What it does	Example
sklearn.cluster	KMeans, MiniBatchKMeans	Partition data into k clusters	kmeans = KMeans(n_clusters=3).fit(X)
	AgglomerativeClustering	Hierarchical clustering (linkage)	
	DBSCAN, OPTICS	Density-based clustering	
	MeanShift, Birch	Other clustering algorithms	
sklearn.neighbors	KernelDensity	Estimate probability density via kernels	
	NearestNeighbors, RadiusNeighborsClassifier	Nearest-neighbor queries (used in DBSCAN, etc.)	

Dimensionality Reduction

Module	Item	What it does	Example
sklearn.decomposition	PCA, IncrementalPCA, TruncatedSVD, FastICA	Linear & non-linear reduction techniques	
	NMF, SparsePCA	Non-negative / sparse factorization	
sklearn.manifold	TSNE, UMAP (via third-party)	Non-linear embedding for visualization	
sklearn.feature_extraction	CountVectorizer, TfidfVectorizer	Convert text to bag-of-words / TF-IDF matrices	

Evaluation - Metrics

Module	Item	What it does	Example
sklearn.metrics	Classification: accuracy_score, precision_score, recall_score, f1_score, roc_auc_score, confusion_matrix, classification_report	Compute common classification metrics	
	Regression: mean_absolute_error, mean_squared_error, r2_score	Compute regression scores	
	Clustering: adjusted_rand_score, silhouette_score	Cluster evaluation metrics	
	Model selection helpers: confusion_matrix, roc_curve, precision_recall_curve	Produce arrays for plots	
sklearn.model_selection (again)	cross_val_predict, validation_curve, learning_curve	Generate predictions / plots across CV folds	

Other - Commonnets

Module	Class / Function	What it does (short explanation)	Typical usage pattern
sklearn.feature_extraction.text	CountVectorizer	Converts a collection of text documents to a raw term-frequency matrix. Each row = document, each column = word (or n-gram).	CountVectorizer(ngram_range=(1,2), stop_words='english')
	TfidfVectorizer	Same as CountVectorizer but applies TF-IDF weighting (term frequency * inverse document frequency). Useful for most ML pipelines.	TfidfVectorizer(max_features=5000)
	HashingVectorizer	Same idea as CountVectorizer but uses a hash trick to avoid storing the dictionary – great for very large corpora.	HashingVectorizer(n_features=2**20)
	StopWords (via stop_words="english")	Built-in list of common English stop words.	-
sklearn.preprocessing	Normalizer	Scales feature vectors to unit norm (useful for cosine-similarity based models).	Normalizer(norm='l2')
	PolynomialFeatures (rarely used in NLP)	Creates interaction terms – occasionally used for character-n-gram models.	-
sklearn.pipeline	Pipeline	Chains together vectorizer → transformer(s) → estimator. Keeps code clean and cross-validation simple.	Pipeline([('vec', TfidfVectorizer()), ('clf', LogisticRegression())])
	FeatureUnion / make_union	Concatenates the outputs of several feature extraction pipelines (e.g., word-level + character-level).	make_union(word_vec, char_vec)
sklearn.decomposition	TruncatedSVD (aka LSA)	Performs latent semantic analysis on sparse TF-IDF matrices; reduces dimensionality while preserving variance.	TruncatedSVD(n_components=100)
	NMF	Non-negative matrix factorisation – useful for topic modelling with TF-IDF.	NMF(n_components=20)
sklearn.manifold	TSNE	Non-linear dimensionality reduction for visualising high-dimensional text embeddings.	TSNE(n_components=2)
sklearn.metrics	classification_report, confusion_matrix, accuracy_score, f1_score, precision_recall_fscore_support	Standard evaluation metrics for text classifiers.	-
sklearn.model_selection	train_test_split, GridSearchCV, RandomizedSearchCV, cross_val_score	Split data, tune hyper-parameters, and evaluate via cross-validation.	-
sklearn.linear_model	LogisticRegression, SGDClassifier (with hinge or log loss)	Fast linear classifiers for text; often the baseline.	LogisticRegression(max_iter=1000)
	PassiveAggressiveClassifier	Online, large-scale linear model that works well on sparse text.	-
sklearn.naive_bayes	MultinomialNB, BernoulliNB	Classic Naïve Bayes models for word-count or binary features.	MultinomialNB(alpha=0.1)
sklearn.svm	LinearSVC, SVC(kernel='linear')	Linear support-vector machines – good for high-dimensional sparse text.	LinearSVC(C=1)
sklearn.tree	DecisionTreeClassifier, RandomForestClassifier	Tree-based ensembles that can handle sparse features.	-
sklearn.ensemble	GradientBoostingClassifier, AdaBoostClassifier	Boosted trees – often outperform linear models on medium-size text corpora.	-
sklearn.neighbors	KNeighborsClassifier (rarely used)	Instance-based classification; works with cosine distance if you normalise vectors.	-
sklearn.neural_network	MLPClassifier	Multi-layer perceptron – can be applied to TF-IDF vectors.	-
sklearn.pipeline (advanced)	FeatureUnion, make_union	Combine multiple vectorizers – e.g., word-level + character-n-gram.	make_union(word_vec, char_vec)
sklearn.preprocessing (advanced)	FunctionTransformer	Wrap custom preprocessing functions inside a pipeline.	-
Utilities	text_dataset (user-defined)	Often you'll create a simple CSV/JSON loader that returns X, y for the above pipelines.	-