PYTHON LIBRARIES

WWW.TANTROSTAV.CO.IN

- Scikit-learn is a popular open-source machine learning library for Python that provides a range of tools for data mining, data analysis, and predictive modeling. It is built on top of NumPy, SciPy, and Matplotlib, and is designed to be easy to use and efficient for both novice and experienced machine learning practitioners.

- Scikit-learn includes a broad range of machine learning algorithms, including supervised and unsupervised learning, as well as feature selection, feature extraction, and data preprocessing techniques.
- It is designed for ease of use and scalability
- Scikit-learn is an open-source library, which means it is free to use and can be customized to suit the needs of different users.

Brief history

- Scikit-learn is a popular open-source machine learning library for Python, which was initially released in 2007 as part of the Google Summer of Code program. The project was started by David Cournapeau, who was working on a machine learning library for the French Atomic Energy Commission.

- The initial release of Scikit-learn included a few basic algorithms, such as linear regression and support vector machines (SVMs). Over time, more algorithms were added, and the library grew in popularity, thanks to its ease of use and scalability.

- Today, Scikit-learn is one of the most popular machine learning libraries in the world, with a large community of developers and contributors who actively maintain and improve the library.

Why scikit learn has an edge over other machine learning libraries

- Scikit-learn is designed to be easy to use, with a consistent API and clear documentation.
- Scikit-learn includes a wide range of machine learning algorithms, as well as feature selection, feature extraction, and data preprocessing techniques.
- Scikit-learn is designed to work well with other Python libraries, such as Pandas and Matplotlib.
- Scikit-learn stands out from other machine learning libraries because of its ease of use, range of algorithms, efficiency, interoperability, and active development and community.

Libraries for data preprocessing techniques

sklearn.preprocessing: This library provides a range of tools for data scaling, normalization, and binarization. It includes classes such as **StandardScaler** for standardization, **MinMaxScaler** for normalization, and **Binarizer** for binarization

sklearn.impute: This library provides tools for imputing missing values in datasets. It includes classes such as **SimpleImputer** for mean and median imputation, and **KNNImputer** for imputing missing values using K-nearest neighbors

sklearn.feature_extraction: This library provides tools for feature extraction and transformation. It includes classes such as **CountVectorizer** for converting text data into numerical data, and **TfidfVectorizer** for generating TF-IDF features.

Libraries for data preprocessing techniques

sklearn.feature_selection: This library provides tools for feature selection and dimensionality reduction. It includes classes such as **SelectKBest** for selecting the K best features based on statistical tests, and **PCA** for performing principal component analysis

sklearn.compose: This library provides tools for combining multiple data preprocessing techniques into a single pipeline. It includes classes such as **Pipeline** for creating a pipeline of data preprocessing steps, and **ColumnTransformer** for applying different preprocessing techniques to different columns in the dataset.

Libraries for model selection techniques

- **sklearn.model_selection**: This library provides tools for model selection and evaluation, including cross-validation, grid search, and random search. It includes classes such as **GridSearchCV** for performing grid search over a set of hyperparameters, **RandomizedSearchCV** for performing random search over a set of hyperparameters, and **cross_val_score** for performing cross-validation

- **sklearn.metrics**: This library provides tools for model evaluation and performance metrics. It includes classes such as **accuracy_score** for computing accuracy, **precision_score** for computing precision, **recall_score** for computing recall, and **roc_auc_score** for computing the area under the ROC curve

- **sklearn.pipeline**: This library provides tools for building and evaluating complex machine learning pipelines. It includes classes such as **Pipeline** for chaining multiple machine learning steps together, and **FeatureUnion** for combining the output of multiple transformers.

Libraries for model selection techniques

- **sklearn.ensemble**: This library provides tools for building ensemble models, such as random forests and gradient boosting machines. It includes classes such as **RandomForestClassifier** for building random forest models, and **GradientBoostingClassifier** for building gradient boosting models

- **sklearn.naive_bayes**: This library provides tools for building naive Bayes models, such as Gaussian naive Bayes and multinomial naive Bayes. It includes classes such as **GaussianNB** for building Gaussian naive Bayes models, and **MultinomialNB** for building multinomial naive Bayes models

Libraries for supervised machine learning

sklearn.linear_model: This library provides tools for linear regression and logistic regression. It includes classes such as **LinearRegression** for performing linear regression, **Ridge** for performing ridge regression, and **LogisticRegression** for performing logistic regression.

sklearn.tree: This library provides tools for decision tree models. It includes classes such as **DecisionTreeClassifier** for building decision tree models for classification, and **DecisionTreeRegressor** for building decision tree models for regression

sklearn.neighbors: This library provides tools for building k-nearest neighbors models. It includes classes such as **KNeighborsClassifier** for building k-nearest neighbors models for classification, and **KNeighborsRegressor** for building k-nearest neighbors models for regression.

sklearn.svm: This library provides tools for building support vector machine models. It includes classes such as **SVC** for building support vector machine models for classification, and **SVR** for building support vector machine models for regression.

Libraries for unsupervised machine learning

- **sklearn.cluster**: This library provides tools for clustering algorithms, such as k-means, hierarchical clustering, and spectral clustering. It includes classes such as **KMeans** for performing k-means clustering, **AgglomerativeClustering** for performing hierarchical clustering, and **SpectralClustering** for performing spectral clustering

- **sklearn.decomposition**: This library provides tools for dimensionality reduction algorithms, such as principal component analysis (PCA), singular value decomposition (SVD), and non-negative matrix factorization (NMF). It includes classes such as **PCA** for performing PCA, **TruncatedSVD** for performing SVD, and **NMF** for performing NMF

- **sklearn.mixture:** This library provides tools for Gaussian mixture models (GMMs), which are probabilistic models that assume the data is generated from a mixture of Gaussian distributions. It includes classes such as GaussianMixture for fitting GMMs to data.

Libraries for unsupervised machine learning

- **sklearn.manifold**: This library provides tools for manifold learning, which is a type of dimensionality reduction that aims to preserve the local structure of the data. It includes classes such as **TSNE** for performing t-SNE, which is a popular manifold learning algorithm.

- **sklearn.ensemble**: This library also provides tools for anomaly detection, including isolation forest and one-class SVM. It includes classes such as **IsolationForest** for building isolation forest models, and **OneClassSVM** for building one-class SVM models

Libraries for evaluation techniques

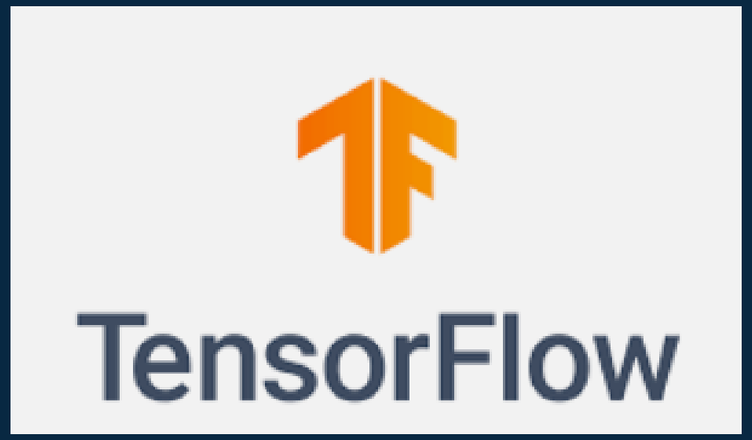
- **sklearn.metrics**: This library provides tools for computing various evaluation metrics for classification, regression, and clustering problems. It includes functions such as **accuracy_score** for computing accuracy, **precision_score** for computing precision, **recall_score** for computing recall, and **f1_score** for computing F1-score

- **sklearn.model_selection**: This library provides tools for model selection and evaluation, including cross-validation and hyperparameter tuning. It includes functions such as **cross_val_score** for computing cross-validation scores, **GridSearchCV** for performing grid search over hyperparameters, and **RandomizedSearchCV** for performing randomized search over hyperparameters

Libraries for evaluation techniques

- **sklearn.metrics.cluster**: This library provides tools for evaluating clustering algorithms, including metrics such as silhouette score and homogeneity score. It includes functions such as **silhouette_score** for computing silhouette score, and **homogeneity_score** for computing homogeneity score.

- **sklearn.pipeline**: This library provides tools for building and evaluating machine learning pipelines, which are sequences of data preprocessing and model building steps. It includes classes such as **Pipeline** for building pipelines, and **make_pipeline** for constructing pipelines from a sequence of estimators.



- TensorFlow was developed by Google and released in 2015. It is an open-source software library that provides a flexible platform for building and deploying machine learning models, including deep learning models.

- TensorFlow allows users to create complex neural network architectures using a combination of pre-built layers and custom layers, and provides tools for training these models efficiently on large datasets using distributed computing.

- TensorFlow is used for both machine learning (ML) and deep learning (DL), but it is particularly well-known for its role in the development of DL models.

Some functions

- tf.constant(): This function is used to create a constant tensor with a specified value.
- tf.Variable(): This function is used to create a variable tensor that can be updated during training.
- tf.nn.relu(): This function is used to apply the ReLU activation function to the input tensor.
- tf.nn.softmax(): This function is used to apply the softmax activation function to the input tensor.
- tf.train.GradientDescentOptimizer(): This function is used to create a gradient descent optimizer for updating the model parameters during training.
- tf.keras.layers.Dense(): This function is used to create a fully connected layer in a neural network.
- tf.keras.layers.Conv2D(): This function is used to create a 2D convolutional layer in a neural network.
- tf.keras.layers.LSTM(): This function is used to create a Long Short-Term Memory (LSTM) layer in a recurrent neural network (RNN).

- PyTorch, on the other hand, was developed by Facebook and released in 2016. It is also an open-source software library for building and training deep learning models, but it has a slightly different focus and approach compared to TensorFlow.

- PyTorch is designed to provide a more intuitive and dynamic programming interface for building neural network models. It allows users to create and modify models using Python code in a more interactive and iterative manner than TensorFlow.

Some functions

- torch.Tensor(): This function is used to create a tensor with a specified size and data type
- torch.autograd.Variable(): This function is used to create a variable tensor that can be used to compute gradients during backpropagation.

- torch.utils.data.DataLoader(): This function is used to create a data loader for loading and preprocessing data in batches during training.

- torch.cuda(): This function is used to perform tensor computations on a GPU for faster training

- torch.nn.Linear(): to create a fully connected layer
- torch.nn.Conv2d(): to create a 2D convolutional layer

- torch.optim.SGD(): This function is used to create a stochastic gradient descent optimizer for updating the model parameters during training.