

IIT Madras

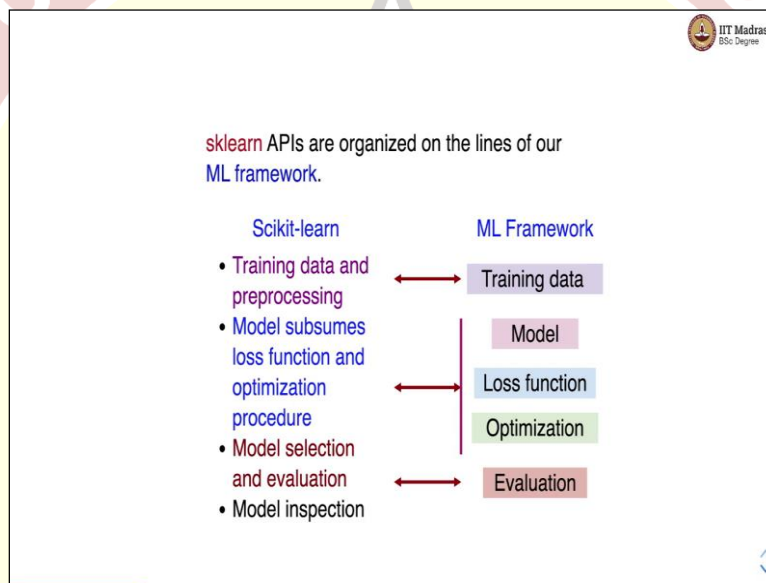
ONLINE DEGREE

**Machine Learning Practice
Online Degree Programme
B. Sc in Programming and Data Science
Diploma Level
Dr. Ashish Tendulkar
Indian Institute of Technology – Madras**

Introduction to Scikit-Learn

Namaste welcome to the next video of the machine learning practice course. In this video, I will introduce you to Scikit-learn. Scikit-learn is a python library that will be used for implementing machine learning algorithms in machine learning practice courses.

(Refer Slide Time: 00:27)



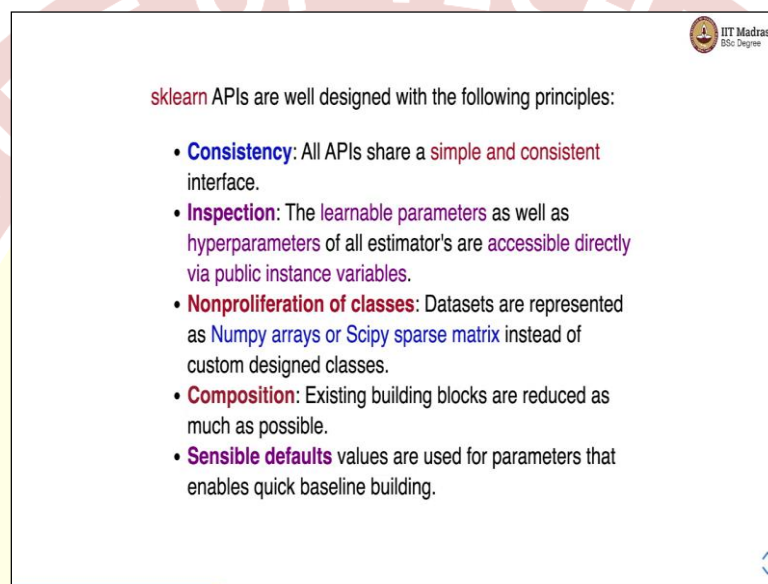
Scikit-learn API's are organized on the lines of our machine learning framework. Here what I am doing is I am going to list the components of the scikit-learn API and machine learning framework side by side. So, that we can establish the correspondences. The first component in scikit-learn API helps us in training data loading preparation and preprocessing it corresponds to the train data component in the machine learning framework.

The second component is a model and the model helps us actually train the machine learning model the model subsumes loss function and optimization procedures. It corresponds to model loss function and optimization components the of machine learning framework. The third component of the scikit-learn API helps us in model selection and

evaluation. In the model selection, we perform cross-validation and hyperparameter tuning to find out the best possible hyperparameters for the model.

And model evaluation helps us with several metrics for measuring the performance of the machine learning model. This corresponds to the evaluation component of the machine learning framework. We have an additional component for model inspection that helps us in inspecting the loan model and understanding what exactly the model has learned.

(Refer Slide Time: 02:06)



sklearn APIs are well designed with the following principles:

- **Consistency:** All APIs share a simple and consistent interface.
- **Inspection:** The learnable parameters as well as hyperparameters of all estimator's are accessible directly via public instance variables.
- **Nonproliferation of classes:** Datasets are represented as Numpy arrays or Scipy sparse matrix instead of custom designed classes.
- **Composition:** Existing building blocks are reduced as much as possible.
- **Sensible defaults** values are used for parameters that enables quick baseline building.

You might get overwhelmed with the number of components several APIs listed in the scikit-learn and package. However, psychologists are very designed with the following principles if you understand these principles it will be easier for you to intuitively figure out what kind of methods or what kind of member variables will be exposed by different sci-kit-learn APIs.

So the first principle is about consistency. All APIs share a simple and consistent interface. For example, there is an estimated API that helps us in training the machine learning model. All estimator APIs have a fit method. So, in that sense if we change from one estimator to the other probably we do not have to change any part of our code and our code will work just fine and this is possible because of this consistency principle.

All estimators have a fit method and in the same way, there are predictors which are another escape and object they have prediction score method. So, no matters what kind of creditor we are using each one of them will have implemented predict and fit method. So,

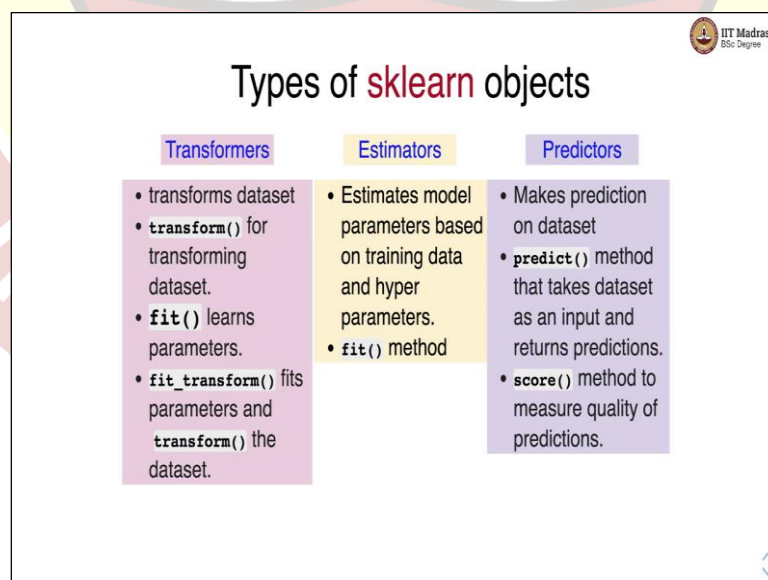
this simple and consistent interface also helps us in remembering or in understanding today what kind of methods we should be using once we figure out the class of the object.

The second is inspection once we train a machine learning model we are interested in inspecting the model we are curious about what kind of parameters what kind of weights each parameter is taking or what kind of hyperparameters have been obtained to hyperparameter search. And scikit-learn API's provide is learnable parameters as well as hyperparameters directly through public instance variables or member variables.

Scikit-learn uses standard data representations like numbers arrays or side by sparse matrix instead of custom design classes that help us not design too many classes in the API and also freeze up programmers from the burden of remembering you know more classes. Then there is the inherent principle of compulsivity existing building blocks are reduced as much as possible and we can combine the very minimum building blocks to build a complex machine learning pipeline.

And finally, there is a principle of using sensibility for twelve years for parameters and that helps us in quickly building a baseline model.

(Refer Slide Time: 05:01)



Let us look at high-level types of scikit-learn objects. There are three types of objects one is transformers second is estimators and third is predictors. Transformers help us in transforming the dataset as the name suggest. Transformers implementary methods


one is transformed second is fit and third is fit transform. Transform method helps us in transforming the dataset.

Before applying transformation we need to learn the parameters of the transformation that is done through the fit method and fit_transform does the job of both the methods fit and transform it first fits the model and then transforms the dataset using the trained model. Estimators on the other hand help us in training the machine learning model based on the given training data and hyperparameter. All estimators have a fit method when we call the fit method on the estimator it helps us enter the machine learning model.

Then there are predictors help us in making predictions on data. Predict method takes datasets dataset as an input and returns predictions as an output. They also provide another method which is the scores method and its score method helps to measure the quality of predictions. Transformers correspond to data processing step in the machine learning pipeline.

Estimators correspond to the training step and predictors correspond to the inference step. So, you can see that these three learn objects are associated with specific steps in the machine learning pipeline, and based on what kind of component of the machine learning pipeline we are implementing we can figure out what kind of scikit-learn object we should be using for that particular purpose.

(Refer Slide Time: 07:02)

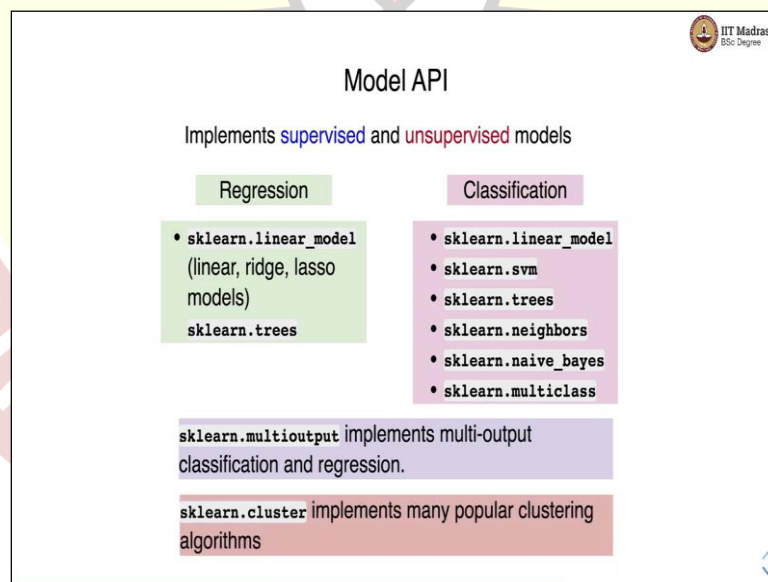
 IIT Madras BSc Degree	
Data API	
Provides functionality for loading, generating and preprocessing the training and test data.	
Module	Functionality
<code>sklearn.datasets</code>	Loading datasets - custom as well as popular reference dataset.
<code>sklearn.preprocessing</code>	Scaling, centering, normalization and binarization methods
<code>sklearn.impute</code>	Filling missing values
<code>sklearn.feature_selection</code>	Implements feature selection algorithms
<code>sklearn.feature_extraction</code>	Implements feature extraction from raw data.

Let us look at some of the concrete scikit-learn API's. First look at the data API. The data API helps us for loading generating and reprocess the training and test set. So there are different modules here the first model is learn the dataset. The sklearn under datasets module helps us ensure the data could be the custom data as well as some of the popular reference datasets.

They come bundled with a sklearn or scikit-learn helps us to face the datasets on the second is a sklearn under preprocessing. The preprocessing module helps us in scaling centering, normalization, binarisation of the features. Then we have the third model which is a sklearn dot impute that helps us in feeding the missing values there could be missing values in the dataset due to various data captured errors.

Then we have a module for feature selection which helps us in selecting certain features based on different algorithms and we have also a module for feature extraction which is sklearn feature underscore extraction that helps us to extract features from the data like text and images.

(Refer Slide Time: 08:17)



Then we have model API, model API helps us to implement supervised as well as unsupervised. So, first is the regression model and second is a classification model. For regression we have a sklearn linear underscore model. Sklearn dot linear underscore module helps us to implement linear ridge and lasso modules. We can also perform regression with it the sklearn the trees.

We can perform classification to various methods like sklearn under cleaning underscore module there are methods like logistics regulation which are implemented in this module then sklearn dot SVM, sklearn dot trees, sklearn dot neighbor, sklearn dot naive_bayes, sklearn dot multiply, or sklearn dot multiclass. So SVM as name suggest implements an SVM algorithm, trees implement decision trees neighbors module helps us in implementing naive kind of classifiers.

Naïve underscore base module helps us in implementing a naïve-based classifier. Naïve based classifier is already implemented and we essentially called the fit method and for that, we have to import the scikit-learn dot naïve underscore base module. And finally, a sklearn learned multi-class module helps us in implementing the multi-class classification setups.

A sklearn and multi-output module help us in implementing multi-output classification and regression setup. So, for example, if you have multi-label classification we should be using multi-output module or if you have multiple regression or multi-output regression we should be using multi-output module from scikit-learn API. And we have a sklearn under the cluster module that implements many popular clustering algorithms.

So, if you want to perform clustering you have to look out for methods in the scikit-learn cluster module. So roughly what you can do is sklearn provides very nice documentation and if you type in this module in the sklearn documentation or the help pages you will find out different classes or different machine learning models that are implemented in that module and that would help you to select the right kind of model and module for you know implementing your machine learning algorithms.

(Refer Slide Time: 11:10)

Model inspection API

`sklearn.model_inspection` includes tools for model inspection.

The model evaluation API implements different metrics for model evaluation. The metrics for classification regression and clustering are implemented by the model evaluations APIs and are available to us in the sklearn library. Then we also have a model selection API that implements various models election strategies like cross-validation hyper-parameter evening and plotting the learning curves.

And we have modeled inspection API that has tools for model inspection where we can figure out you know what are different weights for all the different ways that are learned by the model and this kind of inspection also helps us to get an intuitive feeling about what model has learned.

(Refer Slide Time: 12:06)

Practical advice

- It is not possible to remember each and every sklearn API.
- Remember high level modules and API design principles.
- Use documentation for more information as follows:

```
1 import sklearn.linear_model import LogisticRegression
2 ?LogisticRegression
```

- Keep the following links handy:
 - [API reference](#)
 - [sklearn user guide](#)
 - [Worked examples](#) for reference implementations

Finally some practical advice about using sklearn API. So, remember that it is not possible to remember every scikit-learn API. I recommend that you remember high-level modules and API design principles. Use documentation for more information and how can you use that? this is small porcelain you can import the necessary class here we are importing logistics precaution class from a sklearn underscore linear underscore module.

And then you can you know put a question mark followed by the name of the class to access documentation through its educational notebooks or through google colab and keep the following colab links handy. So, there is a ready reference API guy that points you to scikit-learn a preference page. Sklearn also has an extensive user guide that you can use and it also has worked examples for reference implementations.

All this will help you know to effectively use different sklearn modules for solving the machine learning problem. I hope this has provided this video has provided a good overview of sklearn. Now you are aware of different sklearn modules and the correspondence to components in the machine learning framework. So, after this video now you are ready to implement a machine learning algorithm with scikit API.

And we will be using these modules again and again in machine learning practice courses and various practical implementations. So, I suggest you go through these links and get a feel for a sklearn api and be ready for the rest of the course, thank you and Namaste.