IIT Madras

ONLINE DEGREE

B. Sc in Programming and Data Science
Diploma Level
Dr. Ashish Tendulkar
Indian Institute of Technology – Madras

Demonstration of Sklearn Dataset API

(Video Time: 00:10)

Welcome to the next video of the machine learning practice course. In this video, we will demonstrate a SQL and dataset API. There are three API’s the first one is to load small datasets bundled with SQL on that called loaders and they start with prefix load underscore then we have features that fetch larger datasets from the internet and load them in memory they start with prefix fetch.

And we have generators that generate controls in 3d datasets and they start generating rates or make underscore prefixes. Loaders and features return a bunch object while generators return a tuple of feature matrix and label vector sometimes it also returns label matrix depending on the nature of the problem. Loaders and features can also return a tuple of feature matrix and label vector if we set return underscore x to underscore y argument to true.

Let us first look at the loaders. So, the plan here is to demonstrate loaders then fetchers, and then generators. We will not be demonstrating all the loaders will be demonstrating a couple of loaders and then we will give a few of them as an exercise for you to practice. So, for example, this load diabetes and load digit we will partially demonstrate these loaders and some parts will be left to you as an exercise.

And these are guided exercises where we have given instructions and you are required to write one-line line code based on the instructions provided in the exercise. We will follow the same pattern in features as well as in generators in generators we will have to make regression make classification make multi-level classification and make blobs which are used for generating synthetic datasets for clustering.

So, let us get started with loaders. Let us first load the iris dataset and let us close this table of content. So, that we get more space on the screen. Let us load the iris dataset. So, the iris dataset is an important toy dataset that is used to demonstrate multi-class classification problems. Iris dataset has three classes and each there are 150 examples each example comes up with four features.

So, let us explore how to load this iris dataset using dataset loaders. So, we have a load underscore iris library that helps us load underscore API from SQL under dataset library that helps us to load the iris dataset in memory. So, we can call load to underscore iris function and that returns a dataset which is a bunch object and bunch is a dictionary-like object with the following attribute it has got data and target at least these two attributes are present in the bunch object.

Data has got the feature matrix whereas a target is the label vector then we have feature names and target names they contain names of the features and classes respectively we also have the full description of the dataset in the SCR key and we have the path to the location of the data in the file name key of the bunch object. We can make sure that the return the type of the returned object is indeed bunch by invoking type on this data object and you can see that it is indeed a bunch object.

Now what we will do is we will try to see what kind of features are present. Let us examine you know the content of the bunch object the first one is the names of the features. So, we can simply write this command data, data dot feature underscore names and it returns four features of the iris dataset which is sample length sample width petal length, and petal width.

So, we can see here the names of the features. Then we can also examine the names of the labels by calling target underscore names on the data on the data object and you can see that there are three different classes setosa, versicler and virginica. The next important thing is the feature matrix itself and can access by looking at the data attribute of the bunch object.

Let us look at the first five examples in the feature matrix by invoking the data attribute of the bunch object and looking at the first five examples and you can see that there are four features for example here. So, this is the first example with four features second example with four features, and so on. Let us examine the shape of the feature matrix and by calling data dot data dot shape and we can see that there are 150 examples each with four features.

Next, we will examine the label vector and its shape. The label vector can be invoked by calling the target attribute of the punch object and you can see these are 150 you know labels corresponding to 150 examples and there are 50 examples from each of the classes there is first 50 example belongs to class 0 next 50 examples belongs to class 1 and the third and then the remaining 50 examples belong to class 2.

So, we can read additional documentation on load iris dataset by you know putting a question mark before underscore iris function and you can see that there will be documentation that will pop up here and by reading this documentation you can understand various parameters and the return types of this particular function. This particular function also you know documents what is the dataset like.

So, you can see that there are three classes 50 samples per class the total of 150 examples. Each example has four features and the features are the real positive number. So, you can get much more such information about the dataset as well as the functionality of the API by reading such documentation within your colab notebook. Let us close this documentation and move forward.

So, this is how you can load and examine different datasets. So, you know typical things to examine is feature matrix then target labels then look at the shapes of both feature matrix and labels to make sure that they have same number of examples. Then we can look at other things like the dataset description and if you are interested in knowing more about the function itself or the loader itself we can invoke its documentation.

So, we can also make this these loaders to return tuples of feature matrix and label vector and for that we have to set an argument which is written underscore x underscore y to true and when we do that the load iris will return a tuple of feature matrix and label vectors both of them and are numpy arrays and. So, here so, this is different from what we were doing earlier where we were getting a bunch object and that time we did not set this parameter to true.

But if you desire if you wish to get tuples like this you should be setting the return underscore x underscore y to true and you can also examine the shapes of the feature matrix and label vector and you can see that you know there are 150 examples in feature matrix with four features each and label vector has 150 labels. So, let us move to the other dataset which is the diabetes dataset.

And we can load this diabetes dataset with load underscore diabetes function or class from sklearn dot dataset API. So, as usual, you can you know get additional details about this loader by accessing its documentation, and then what I have done here is I have actually left this particular thing as an exercise for you where I have given the steps about what you should be doing here.

So, here you should write functionality to load the dataset and get a bunch object, and then you need to examine the bunch object look at the description of the dataset find out the shape of the feature matrix look at the first five examples of the feature matrix find out the shape of the label matrix look at the labels of first five examples find out the names of the features names of the class labels.

So, these are typical exploration steps about a dataset and what I have done here is I have you know I have created some sort of an exercise for you to write these one-line codes in these code cells that are also included in this colab. We do the same thing with the loads rigid dataset for loading the digit dataset and you can see that there are a bunch of instructions that I have written for you to follow and work on.

So, as an exercise you know you can experiment with other data state loaders like a wine loader breast cancer loader, and network loaders and examine those and examine that dataset. So, I have actually installed one of these exercises for you for loading the wind dataset. So, you can access the documentation of this by putting the question mark before this class itself.

And you will get documentation. you can see the documentation for the wine dataset here. You can see that there are three classes and there are a total of 178 samples the first class has 59 samples second one has 71 samples and the third one has 48 samples each sample has 13 features and its features are real positive numbers. Let us close the help. Then we can actually call the load wine loader and get the punch object and examine the bunch object look at the description of the dataset by looking at the SCR attribute of the bunch object and you can see you know it is not.

So, nicely formatted but this is the description of the dataset it lists down the dataset characteristics it also leads to listing down the attributes and also has some kind of summary statistics and the class information. You can find out the shape of the feature matrix by accessing the feature matrix to the data attribute of the bunch object and then you know examining the shape of this particular feature matrix.

You can see that there are 178 examples each with 13 features and we can look at the first five examples by accessing the feature matrix and its first five examples like this. Then there are things like you know examining the shape of the label matrix looking at first five examples and so on and these are the names of the feature these are 13 features of the dataset they include alcohol malic acid, ash, alkalinity of the ash magnesium total phenol flavonoids and so on.

There are these 13 different features in this dataset and there are three classes class underscore 0 class underscore 1 and class underscore 2 are the names of the classes. So, in the same manner, you should explore this breast cancer dataset and linear root dataset that was about loaders. Let us move to features. So, we are going to use the California housing dataset for you know to demonstrate the feature transformation as well as linear regression concepts or how to perform linear regression in the sklearn library we are going to use the California housing dataset and we can fetch the California housing dataset does not come bundled with skill and library we need to fetch it from the internet and fortunately we have to fetch underscore California underscore housing class that is provided by the scale under dataset API. We can look at the documentation by you know putting the question mark before the name of the class itself. Again these features return bunch objects like loaders we can also make features to return tuples by setting return underscore x underscore y to true.

And we can fetch the California housing dataset by calling fetch underscore California underscore housing we get the data in the housing underscore data bunch object and we can examine this bunch object by looking at the description of the dataset by checking the shape of the feature matrix looking at first five examples in the feature matrix looking at the shape of the label matrix.

So, there are about 20046 examples and each example has got 8 features and you can also see that the label or the or the label matrix here in this case there is a single label single target hence it is a label vector and it has got 20046 for the example. So, we can check the shape of the first dimension of the target and feature matrix to make sure that the number of examples is the same or the first dimension is the same in both cases.

This is the basic sanity check that should be done on each dataset that you load in memory. Then we can look at the labels of the first five examples you can see that the labels are real numbers in case of the housing price dataset or housing dataset we can also look at names of the features. So, there is a mid-income house age average room size, average bedrooms, population average, average occupancy, latitude, and longitude.

These are the features and then the label is median house value. So, that was about the California housing dataset. We are also going to use fetch underscore openml API in one of our colabs and mainly to fetch the digit dataset or for loading the mnist dataset. So, a fetch underscore openml is an experimental API and is likely to change in future releases. So, we can essentially call fetch underscore openml with the name of the dataset which is amnest under underscore 784 we can also specify the version of the dataset here.

And here instead of returning the bunch object, we are interested in the tuple of feature matrix and label vector. So, we set return underscore x underscore y to true and we can examine the shape of the feature matrix as well as label vector. We can see that there are 70 000 examples each in the feature matrix and label and feature matrix each example has 784 features.

So, there are a couple of exercises that I have included for you to try basically fetch 20 newsgroups as well as the KDD cup dataset. Kddcup dataset is one of the largest datasets that are available with the features and newsgroup 20 newsgroup dataset is a very famous dataset used in the text classification for checking or for training the test classification text classification algorithm.

So, that was about loaders and features finally. Let us look at the generators. And generators basically help us to generate synthetic data datasets with specified statistical properties. So, there are generators for regression for classification single level as well as multi-label classification and for clustering. So, making an underscore regression class helps us to generate datasets for regression.

So, let us generate 100 samples with five features for a single-label regression problem. So, you can do that by calling make underscore regression with any underscore samples set to 100 this generates a hundred samples with features equal to five number of targets equal to one and we said shuffle to true and it is important to set the random state. Because when we set the random state we get to see the repeatability in the experiment.

And when we run it we get x, y tuple x is the feature matrix and y is the wiser label vector and we can examine the shapes of both of these NumPy arrays. So, you can see that x has 100 examples each with 5 attributes and the label vector also has 100 labels. Let us generate a dataset for a multi-output regression problem. And let us say we want to generate a dataset for multiple regression with 5 outputs.

So, in this case, we just set n targets to 5 by keeping all other arguments the same. So, if you compare these two statements they differ only in this n targets here it is one whereas here it is five. We can examine the shapes and make sure that the basic sanity check is passed by the loader or by the generator here. In the same manner, we can use make classification functionality from the scale and data set API for generating random n class classification problems.

Here we are going to generate a classification problem with 100 training examples each with 10 features and a number of classes equal to 2. So, it is a binary classification problem, and a number of clusters per class we are setting it to one. So, this is a simple classification problem that we are trying to you know set up and again we are setting seed. So, here if you recall my classification, first of all, generates clusters and then assigns those clusters of points to the classes.

And here we are saying that we want to assign points from a single cluster to one class. So, in that sense, it is a simple classification setup that will be generated by this particular statement. We can examine the shapes look at the first few examples and their labels and if you want to generate a multi-class classification setup we can simply change the number of classes to three.

So, earlier this was set to two for binary classification setup but now we want you to know the classification set up with three classes. So, we simply modify these n classes to three, and then we perform the usual checks on the loaded data. So, that was about a single label classification problem with binary and multi-class setups. We can also generate multi-label classification problems by making underscore multi-label underscore classification functionality.

So, here again, we have to set and samples and features and classes. So, in classes tells you how many labels are there and this n underscore labels are something which is new which was not there earlier in the classification setup and these n labels provide an average number of labels for example. So, the average number of labels here is two and you can see that the label is no more a vector.

But it is a matrix because we now have a multi-label classification problem where each example can take more than more than one label. Now let us examine a few rows of feature matrix and label matrix. So, the feature matrix is similar to what we saw in regression and in the earlier classification setup but what is interesting is this label matrix earlier used to be a label vector in a single and multi-class classification setting.

Now we have a label matrix and you can see that there is each example has more than one label. So, whenever the label is present it is represented by one and if it is absent it is represented by zero. So, in that case in the case of the first example there are three labels that are present label one three, and four then in this fifth example, we have labels two-four, and five that are present.

So, you can this is different from the earlier setup and finally, we have to made underscore blobs to generate a dataset for clustering. And here we can specify you know the number of samples the number of centers number of features and number of centers basically help us to generate a dataset from those many clusters. So, since we are providing three centers we will be generating a dataset such that there are three clusters.

And you can see you can find the cluster membership of each point through the label vector and you can see that the first point belongs to cluster two-second one belongs to cluster two-third one belongs to cluster one and so on. So, this was about loaders features and generators the three main kinds of API's or the three main kinds of functionalities or interfaces provided by the SQL under dataset API.

Apart from that, there are other functionalities that are provided to load training data in other formats may be in lib SVM sparse format or in audio-video as well as image formats. Also, the training dataset could be present in formats like CSV's or in binary formats mostly in scientific computing. And we looked at you know functions or interfaces to load that kind of data.

And I urge you to you know to try out those you know those interfaces on your own and get yourself familiarized with how to load training data in different formats with a sklearn dataset interface or sklearn under dataset API. So, I hope this gives you clarity on you know the theoretical as well as practical aspects of the dataset API and now you have a good idea of how to load a trained dataset in memory.

And you know perform the some of the learning based on the algorithms that you are already familiar with. So, in the next video, we will study or will discuss dataset pre-processing and something like dataset transformation you know feature extraction feature reduction etcetera, so, until then goodbye and namaste.

(Video End: 28:15)