RESUMEN

Use Scikit learn when you need:

* To transform data before processing
* Especially for small amount of data
* Small and médium-size projects
* For prototypes
* Doesn’t require GPU

On the contrary, use TensorFlow when

* Not need to transform data
* To process high amount of data
* For big-size and complex projects
* When looking for a high accuracy
* Requires GPU-accelerated operations

For beginners, start with Scikit and move to Keras and Tensorflow.

Combination

* sklearn is responsible for basic data cleaning tasks
* keras are used for small-scale experiments to verify ideas
* TF is used for serious parameter adjustment (alchemy) tasks on complete data.

Traditional methods are generally more interpretable, which is also very helpful for checking the debug model.

|  |  |
| --- | --- |
| **Sckikit learn** | **Tensorflow** |
| General purpose ML | Deep learning |
| Needs to refine data | Users don’t need to refine data |
| Users select features | Users don’t select features |
| Structure steps | Higher degree of freedom, user decides how many layers, steps, and parameters, more specific process |
| Especially for a small amout of data | High amount of data |
| Requires manually process the data | Doesn’t require to manually process the data |
| Choose model (decisión tree, regression, bayes) | No need to choose a model |
| Low hardware requirement | Sometimes requests high hardware |
| Small and médium-size projects | Big projects |
| Can often be completed on the CPU | Requires GPU-accelerated operations |
|  | Deep learning methods generally require a large number of GPU machines |
| Acceptable accuracy | Looking for high accuracy |
| For beginners | For experts |
| Not for neural networks | Neural networks |
| Prototypes | Final , for serious parameter adjustment tasks |
| More interpretable, for checking the debug model | Test several parameters |
| For simple projects, regression, clustering | Usually for image speech recognition, NLP |

1. Main difference

**Scikit-learn (sklearn) as a general-purpose ML library** ,

while **TensorFlow (tf) is as a deep learning library** .

2. Obvious difference

TF does not provide the powerful feature engineering of sklearn, such as dimensional compression, feature selection, etc. The root cause, I think, is because of two different ways of processing data with ML models:

* **Traditional ML: use feature engineering to artificially refine and clean the data**
* **Deep learning: using representation learning, the ML model itself refines the data**

**Sklearn** users process data, such as selecting features, compressing dimensions, and transforming formats. It is a traditional ML library.

**TF** deep learning library will automatically extract valid features from the data, and does not need to do this manually.

3. Different degrees of freedom

**The modules in scikit-learn are highly abstract. All classifiers can basically be completed in 3–5 lines. All converters (such as scaler and transformer) also have a fixed format** . This abstraction limits the user’s freedom, but increases the efficiency of the model and reduces the difficulty of batching and standardization (through the use of pipelines).

**TF is different. Although it is a deep learning library, it has a high degree of freedom** . You can still use it to do what traditional machine learning does, at the cost of implementing algorithms yourself. Therefore, it is not suitable to use TF analogy with scikit-learn. Keras encapsulated in tool libraries such as TF is more like scikit-learn in the deep learning world.

From the perspective of degrees of freedom, TF is higher; from the perspective of abstraction and encapsulation, sklearn is higher; from the perspective of ease of use, sklearn is higher.

3. Different groups and projects

**sklearn for**

* small and medium-sized projects,
* especially for a small amount of data
* and require to manually process the data
* choose the appropriate model.
* projects that can often be completed on the CPU
* has low hardware requirements .

**TF for projects that**

* have clearly understood the need for deep learning
* have low data processing requirements.
* have a large amount of data
* require higher accuracy
* require GPU-accelerated operations .

For “learning” of deep learning, you can use keras for quick experiments on small sampled data sets.

You can understand why keras is comparable to sklearn on deep learning.

model = Sequential() # define

model.add(Dense(units=64, activation='relu', input\_dim=100)) #define network structure

model.add(Dense(units=10, activation='softmax')) # define network structure

model.compile(loss='categorical\_crossentropy', # define lossfunction, optimization method, evaluation criteria

optimizer='sgd',

metrics=['accuracy'])

model.fit(x\_train, y\_train, epochs=5, batch\_size=32) # training model

loss\_and\_metrics = model.evaluate(x\_test, y\_test, batch\_size=128) # evaluation model

classes = model.predict(x\_test, batch\_size=128) # use the trained data for prediction

There are also neural network modules in sklearn, BUT it is impossible to rely on sklearn for serious and large-scale deep learning.

Although TF can also be used for traditional ML, including cleaning data, it is often more effective.

4.Scikit-learn & tensorflow combined use

* sklearn is responsible for basic data cleaning tasks
* keras are used for small-scale experiments to verify ideas
* TF is used for serious parameter adjustment (alchemy) tasks on complete data.

If you take sklearn out and look at it alone, its documentation is particularly good.

Beginners will probably have a basic understanding of many aspects of ML when they follow the features supported by sklearn.

As a simple example, sklearn often summarizes individual knowledge points, such as simple anomaly detection. Therefore, sklearn is not just a simple tool library, its documentation is more like a simple beginner’s guide.

Therefore, traditional ML libraries represented by sklearn (universal but highly abstract like the Swiss Army Knife) and free and flexible more targeted deep learning libraries represented by TF (highly free but cumbersome to use like Lego) are both It is a tool that machine learners must understand.

But sklearn is still necessary to learn

5. Other Reasons

**Deep learning technology is also a component of ML** .

Other traditional ML methods is very helpful for deep understanding of deep learning technology.

Conditions of the model’s convexity can better understand the non-convexity of neural networks. Advantages of Traditional models can better understand that deep learning is not a panacea.

There are scenarios when using deep learning will encounter bottlenecks and problems that require traditional methods to solve.

In practice, deep learning methods generally require a large number of GPU machines. Even large companies in the industry have limited GPU resources.

Generally, deep learning methods are only considered if they have far better results than traditional methods. Using deep learning methods, such as speech recognition, image recognition and other tasks are now more used in deep learning methods. In addition to machine translation in the NLP field, most other tasks still use traditional methods more often.

Traditional methods are generally more interpretable, which is also very helpful for checking the debug model.

The industry generally likes to recruit people who can solve problems, rather than those who have mastered the hottest technologies. Therefore, while learning about deep learning techniques, it is beneficial to learn about traditional methods.

Deep learning is popular, many times you still have to solve problems with traditional machine learning methods.

First of all, not everyone has a sturdy computer / server, and secondly, most problems really don’t require a deep network. Finally, programmers who only call the toolkit are not good machine learners.