Hands On Machine Learning Notes

Notes based on the Book "Hands-On Machine Learning with Scikit-Learn & TensorFlow" by Aurelien Geron

Krish Katariya

Last updated: **February 08, 2024**

Contents

1. The Machine Learning Landscape 1
1.1. Types of Machine Learning systems 1
1.1.1. (Un)Supervised Learning 1
1.1.2. Semisupervised Learning 2
1.1.3. Reinforcement Learning 3
1.1.4. Batch Learning 3
1.1.5. Instance vs Model-Based Learning 3
1.2. Main Challenges of Machine Learning 3
1.2.1. Insufficient Quantity of Training Data 3
1.2.2. Non-Representative Data 4
1.2.3. Overfitting the Training Data 4
1.2.4. Underfitting the Training Data 4
1.3. Testing and Validating 4
1.3.1. Hyperparameter Turning and Model Selection 4
1.3.2. Data Mismatch 4

1. The Machine Learning Landscape

1.1. Types of Machine Learning systems

There are many different Types of ML systems that it is often useful to classify them in broad categories based on :

- Whether or not they are trained with human supervision (supervised, unsupervised, reinforcement learning)
- Whether or not they can learn incrementally on the fly (online vs batch learning)
- Whether they work by simply comparing new data points to old data points, or instead detect patterns in training data

1.1.1. (Un)Supervised Learning

- Machine learning systems can be classified according to the amount and type of supervision they get during training. There are four major categories: **supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning**

Definition 1.1.1.1

In **supervised learning**, the training data you feed to the algorithm includes the desired solution, called **labels**

A typical supervised learning task is **classification**. The spam filter is an example of this, where models are filled with lots of emails with their class(label) of spam/normal emails.

Another task can be predict a target numeric value, such as the price of a car, known as *predictors*. This task is often called **regression**.

- One example of regression is actually **logistic regression**, which is used for classification. Basically, it outputs a certain probability than an object belongs to a class.

Example 1.1.1.1

Important Supervised Learning Algorithms:

- k-Nearest Neighbors
- Linear Regression
- Logistic Regression
- Support Vector Machines (SVM)
- Decision Trees and Random Forests
- Neural Networks (*can also be unsupervised*)

Definition 1.1.1.2

In **unsupervised learning** the training data is unlabeled. In this case, the system tries to learn without having a teacher.

Example 1.1.1.2

Important Unsupervised Learning Algorithms

- Clustering
  - K Means
  - DBSCAN
  - Hierarchical Cluster Analysis (HCA)
- Anomaly detection and novelty detection
  - One class SVM
  - Isolation Forests
- Visualization and dimensionality reduction
  - Principal Component Analysis (PCA)
  - Kernel PCA
  - Local Linear Embedding
  - t-distributed Stochastic Neighbor Embedding
- Association rule learning
  - Apriori
  - Eclat

For example, say you have a lot of data about your blog's visitors. You can run an unsupervised model to group them without ever giving info about what groups they are in.

1.1.2. Semisupervised Learning

Definition 1.1.2.1

Some algorithms can deal with partially labeled training data, usually a lot of unlabeled data and a little bit of labeled data. A good example is google photos when it detects some of your pictures faces to get the rest.

1.1.3. Reinforcement Learning

Definition 1.1.3.1

Reinforcement Learning has a learning system called the *agent* that observes the environment, selects and performs actions, and gets rewards/penalties based on its actions. It then find its best strategy, called its *policy*, to get the most reward.

1.1.4. Batch Learning

Definition 1.1.4.1

In **batch learning**, the system is incapable of learning incrementally: it must be trained using all the available data. This often takes a huge amount of time and lots of computational resources, so it is often done offline, then once fully done it can be used, This is called *offline learning*.

While simple, this can take many hours of training and is not often ideal, also requiring many computing hours.

Definition 1.1.4.2

In **online learning**, you train the system incrementally by feeding it data instances sequentially, in "mini batches". This is especially useful when you have a continuous flow of new information.

These systems are especially good because they don't take much computation power since you don't need to retrain many times.

Definition 1.1.4.3

Learning rate is how fast a system should adapt to new data.

If you set it too high, it will rapidly adapt but will also tend to quickly forget the old data. If it is lower, it will learn slowly but will be more receptive to remembering things and not being affected by *noise in the data.*

1.1.5. Instance vs Model-Based Learning

Definition 1.1.5.1

Instance Based Learning means learning something by heart and then generalizing to new cases by comparing them to learned examples using a *similarity measure.*

Definition 1.1.5.2

Model Based Learning means to build a model based on examples and use those models to make predictions.

1.2. Main Challenges of Machine Learning

1.2.1. Insufficient Quantity of Training Data

Machine learning is not nearly as fast as humans at learning at a fast pace, so it takes lots of data for algorithms to work properly. Even for very simple problems, you typically need at least thousands of examples.

1.2.2. Non-Representative Data

In order to generalize well, the training data should be representative of the new cases that you want to generalize to.

1.2.3. Overfitting the Training Data

Sometimes models end up overgeneralizing instead, which is known as **overfitting**. This is especially often if the data set is noisy, because the model will tend to detect patterns in the noise itself which will not generalize to new instances.

Definition 1.2.3.1

Regularizing is the process of reducing overfitting by constraining a model by making it simpler. The amount of regularization to apply during learning can be controlled by a **hyperparameter**, which is a parameter of the learning algorithm (not the model itself), which is set prior to training and remains constant during training.

1.2.4. Underfitting the Training Data

Definition 1.2.4.1

Underfitting training data occurs when your model is too simple to learn the underlying structure of the data. This is common with linear models.

Most solutions include:

- Selecting a more powerful model, with more parameters
- Feeding better features to the learning algorithm (**feature engineering**)
- Reducing the constraints on the model (reducing the regularization hyperparameter)

1.3. Testing and Validating

The only way to know how well a model will generalize to new cases is to actually try it on new cases.

Note 1.3.1

Often, we split up our data into two sets: the training and test set. The error rate on new cases is known as the *generalization error*, and will tell you how it performs on new instances. If your training error is low but your generalization error is high, bou are overfitting the training data.

We often leave 20% of our data for testing

1.3.1. Hyperparameter Turning and Model Selection

Often we carve out part of the testing data into a **validation set**. We then try to train different models using (testing data - validation set) and see which one works best to see what we should use. If we use **cross-validation**, we can have many small validation sets and then we can then average out all evaluations of a model for a more accurate measurement.

1.3.2. Data Mismatch

In some cases, you only have a few representative data but lots of data in general. If you split this up between validation and testing, the training data isn't very representation. So what one can do is hold out part of the training data to use as a pseudo test set, called the **train-dev-set**. We can then test is on the train dev set to see if it is overfitting/is performing well. If it is not performing well, you need to try to address the data mismatch in some other way.

Need to do Data Mismatch