Scikit Learn Notes

Stephen CUI

October, 25, 2022

Contents

Part I

User Guide

Chapter 1

Supervised learning

1.1 Linear Models

The following are a set of methods intended for regression in which the target value is expected to be a linear combination of the features. In mathematical notation, if $\hat{y}$ is the predicted value.

$$\hat{y}(w,x) = w_0 + w_1 x_1 + ... + w_p x_p \tag{1.1}$$

Across the module, we designate the vector $w = (w_1, ..., w_p)$ as `coef_` and w_0 as `intercept_`.
To perform classification with generalized linear models, see Logistic regression.

1.1.1 Ordinary Least Squares

LinearRegression fits a linear model with coefficients $w = (w_1, ..., w_p)$ to minimize the residual sum of squares between the observed targets in the dataset, and the targets predicted by the linear approximation. Mathematically it solves a problem of the form:

$$\min_w ||Xw - y||_2^2 \tag{1.2}$$

LinearRegression will take in its `fit` method arrays X, y and will store the coefficients w of the linear model in its `coef_` member:

```
from sklearn import linear_model
reg = linear_model.LinearRegression()
reg.fit([[0, 0], [1, 1], [2, 2]], [0, 1, 2])
reg.coef_
```

The coefficient estimates for Ordinary Least Squares **rely on the independence of the features**. When features are correlated and the columns of the design matrix X have an approximately linear dependence, the design matrix becomes close to singular and as a result, the least-squares estimate becomes highly sensitive to random errors in the observed target, producing a large variance. This situation of multicollinearity can arise, for example, when data are collected without an experimental design.

Examples:

- Linear Regression Example

Non-Negative Least Squares

It is possible to constrain all the coefficients to be non-negative, which may be useful when they represent some physical or naturally non-negative quantities (e.g., frequency counts or prices of goods). LinearRegression accepts a boolean `positive` parameter: when set to `True` Non-Negative Least Squares are then applied. **Examples:**

- **??**

Ordinary Least Squares Complexity

1.1.2 Ridge regression and classification

1.1.3 Lasso

1.1.4 Multi-task Lasso

1.1.5 Logistic regression

Chapter 2

Model selection and evaluation

2.1 Cross-validation: evaluating estimator performance

Learning the parameters of a prediction function and testing it on the same data is a methodological mistake: a model that would just repeat the labels of the samples that it has just seen would have a perfect score but would fail to predict anything useful on yet-unseen data. This situation is called **overfitting**. To avoid it, it is common practice when performing a (supervised) machine learning experiment to hold out part of the available data as a **test set** `X_test`, `y_test`. Note that the word "experiment" is not intended to denote academic use only, because even in commercial settings machine learning usually starts out experimentally. Here is a flowchart of typical cross validation workflow in model training. The best parameters can be determined by grid search techniques in Tuning the hyper-parameters of an estimator.

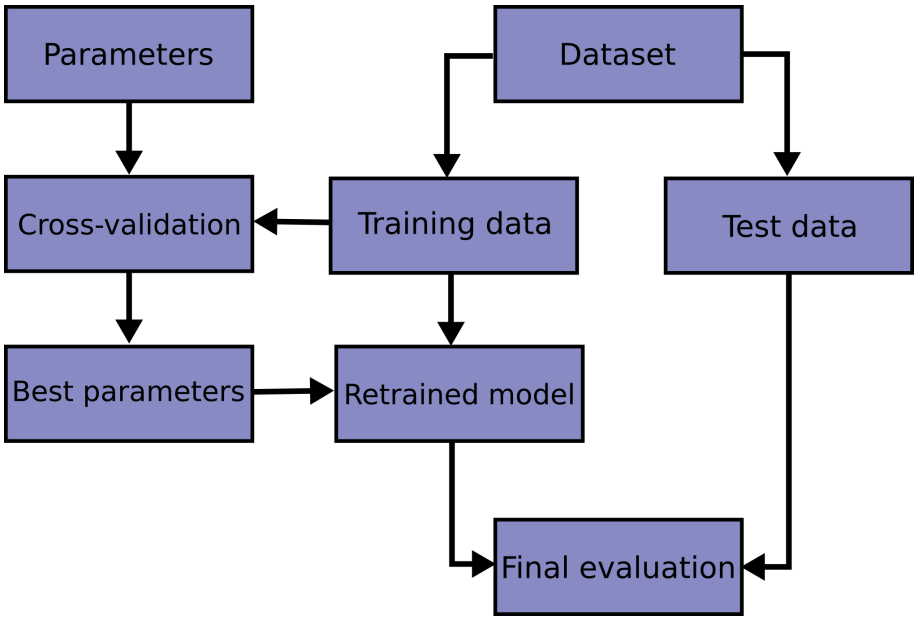

Figure 2.1: https://scikit-learn.org/stable/_images/grid_search_workflow.png

2.2 Tuning the hyper-parameters of an estimator

Part II

Example

Chapter 3

Generalized Linear Models

3.1 Linear Regression Example

3.2 Non-negative least squares