Machine Learning with Scikit-Learn

Youssef Wally

Ulm University

March 29, 2020

Training Models

The idea is to find the least cost for the model. Meaning we want to find the value of theta that minimizes the cost function.

Training Linear Regression Models

The cost function for linear regression is its mean square root equation for theta. To find the theta that minimizes the cost function we use the normal equation:

$$\theta = (X^T X)^{-1} X^T y \tag{1}$$

Where:

1. θ is the value of θ that minimizes the cost function.
2. y is the vector of target values containing y^1 to y^m .

or SVD

The problem is that computation of the normail equation is $O(n^3)$ and $O(n^2)$ for the SVD. Therefore there are different methods.

Gradient descent

The main idea is to start with a random θ and then start changing it in order to minimize the cost function. An important thing to remember is that gradient descent works better when we scale the data. There are two types of gradient descent:

1. batch gradient descent
2. Stochastic gradient descent

Batch Gradient Descent

The batch gradient descent considers the whole dataset. You then set the learning rate and number of iterations. However with large sets of data this will be slow

Stochastic Gradient Descent

Therefore Stochastic gradient descent uses one instance of the data. This is benifical because its faster however it does not get the optiml theta but rather bounces around it.

Mini-batch Gradient Descent

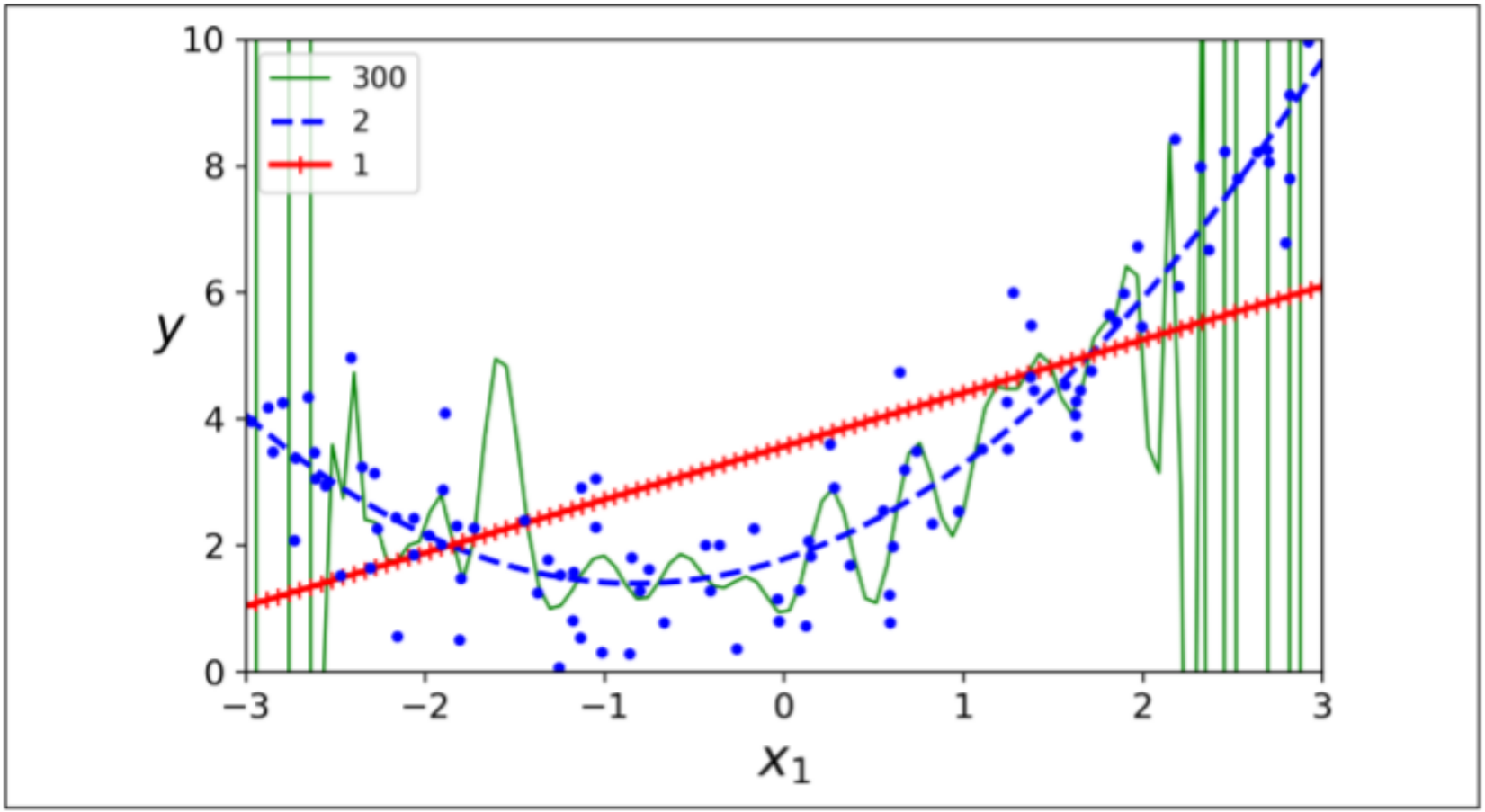
Mini-batch Gradient Descent makes a mix between both batch and Stochastic Gradient Descent by using samples of the data and not the whole set.

summary

Algorithm	Large *m*	Out-of-core support	Large *n*	Hyperparams	Scaling required	Scikit-Learn
Normal Equation	Fast	No	Slow	0	No	n/a
SVD	Fast	No	Slow	0	No	`LinearRegression`
Batch GD	Slow	No	Fast	2	Yes	`SGDRegressor`
Stochastic GD	Fast	Yes	Fast	≥2	Yes	`SGDRegressor`
Mini-batch GD	Fast	Yes	Fast	≥2	Yes	`SGDRegressor`

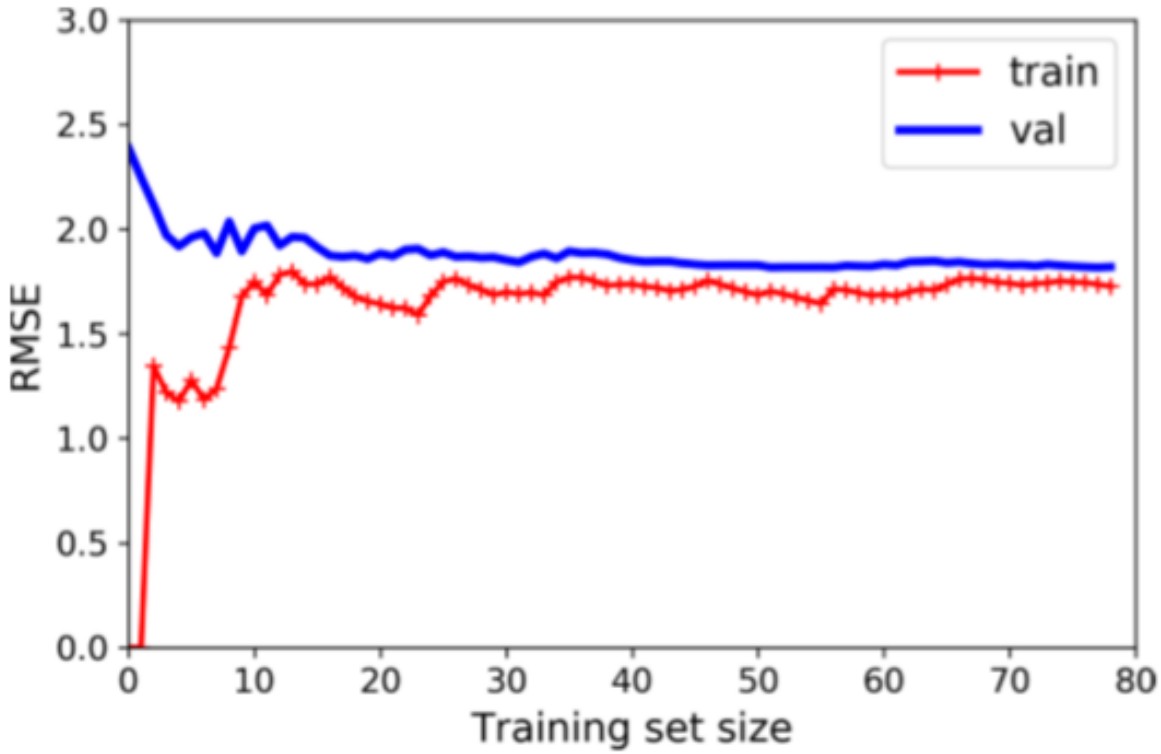
Polynomial Regression

What if the data is non-linear? Then we can still use linear regression but add a polynomial feature and as its degree gets higher it fits better but it gets so high it will over fit.



Learning Curve

A method to see the correctness of the model is the Learning curve.



Bias

This part of the generalization error is due to wrong assumptions, such as assuming that the data is linear when it is actually quadratic. A high-bias model is most likely to underfit the training data.10

Variance

This part is due to the model's excessive sensitivity to small variations in the training data. A model with many degrees of freedom (such as a high-degree polynomial model) is likely to have high variance, and thus to overfit the training data.

Regularized Linear Models

Regularized models are linear regression but with small weight to allow small cost, For example:

1. Ridge Regression (searches for minimum weight)
2. Lasso Regression (tends to eliminate the weight of the least important feature)
3. Elastic Net (Mix of both depending on r)

Other methods for regularizing

1. Early stopping; the method of just stopping when you have reached a minimum after awhile at it stayed the same for awhile
2. Logistic Regression; a binary classifier if it is predeicted prob. is less than 50 percent then its true if not false, however there is no closed form for its gradient discent
3. softmax regression; using logistic regression to predict many classes but only mutually exclusive classes

SVM

SVM can be used in linear/non linear classification, regression or detection

1. SVM is very feature sensitive

Linear SVM Classification

Basically the separation of data by a straight line.

1. large margin classification is the fact of seperation the data with a line which is the furthest from the classified data to not be biased only on training data
2. however as the street goes wider the outliers increase the idea is to find the balance between hard and soft margin classification which is indicated with a number with the variable C

Nonlinear SVM Classification

It is the same case as adding polynomial feature as we disscussed before or by a similarity function that takes many features and makes a landmark. However, it is slow as the degree increases. Thats why we use Polynomial Kernel or Gaussian RBF Kernel.

1. gamma makes bell shape narrower as it gets higher in Gaussian RBF Kernel

Summary

Class	Time complexity	Out-of-core support	Scaling required	Kernel trick
`LinearSVC`	$O(m \times n)$	No	Yes	No
`SGDClassifier`	$O(m \times n)$	Yes	Yes	No
`SVC`	$O(m^2 \times n)$ to $O(m^3 \times n)$	No	Yes	Yes

SVM Regression

The idea is to fit many data as possible on the line this time

1. epsilon determines the width of the 'street' (increase increase)

and as always there is a kernal trick for the non linear regression.

Decision Tress

1. They are orthoganal and therefore rotation of data can cause problems, inaddition they are very sensitive so the same data can make two diffrent figures unless you set the random state (random forests takes average of trees and minimizes this)
2. Decision tress dont require scaling or centering
3. Samples: how many training sampeles are classified in that node
4. value: how much of the sample is in which class
5. gini: the purity of the class (classification)
6. MSE: mean square error, splits to minimize it (regression)