Training models: Learning

Jens Peter Andersen, Assistant Professor, Roskilde
Michael Claudius, Associate Professor, Roskilde, with respect and gratefullness to Jens Peter

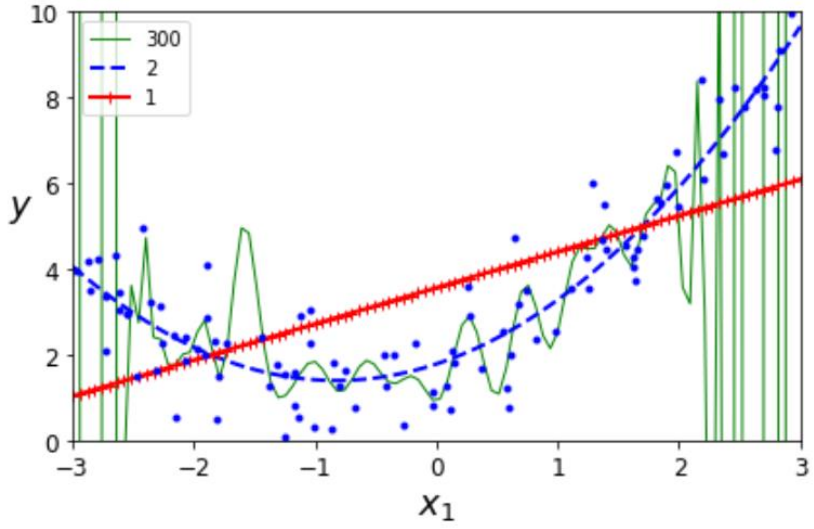
26-03-2020

Learning curves

- **Purpose:** Evaluating a model by comparing performance RMSE on both the training and validation sets
- **Focus:** Overfit and underfit situations

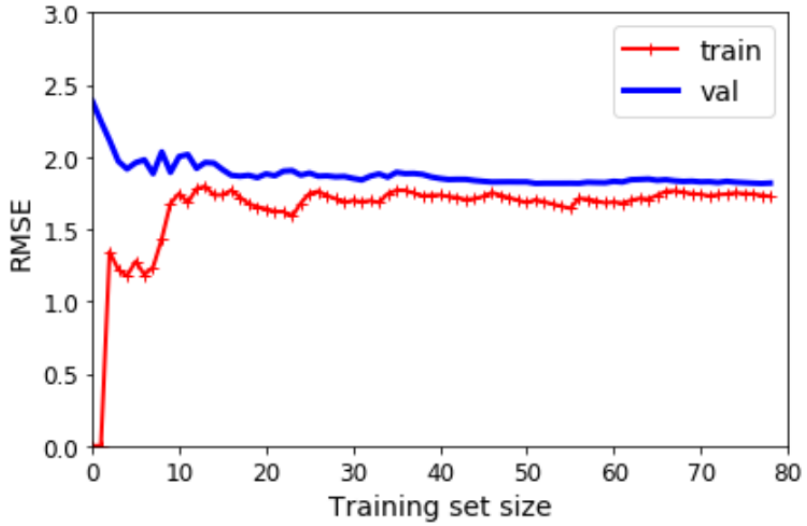
Learning curves - examples

- **Overfit:** Green curve, polynomium degree 300. Performs well on the training set. Will it also perform well on the validation set?
- **Underfit:** Red curve, straight line (polynomium degree 1). Comparable lower performance on both training and validation sets.
- **Good fit:** Blue curve. Polynomium degree 2. Good performance on both training and validation sets.



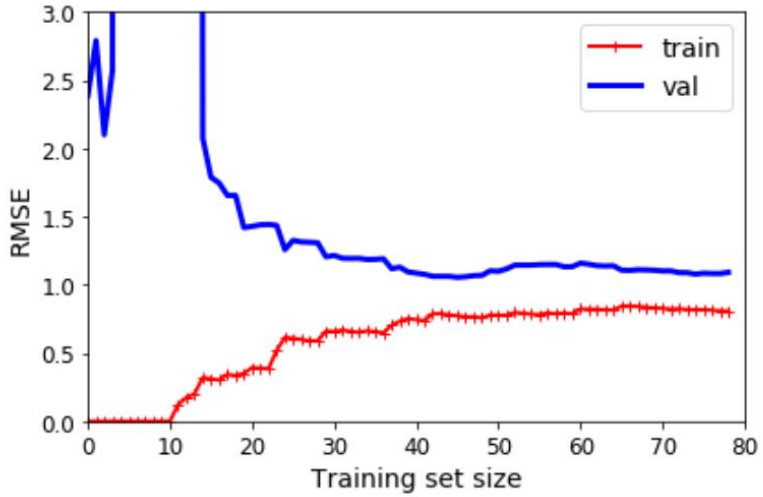
Learning curves – recognizing underfit

- Relatively poor performance RMSE on both validation and training sets
- Performance RMSE on both validation and training sets are compareable



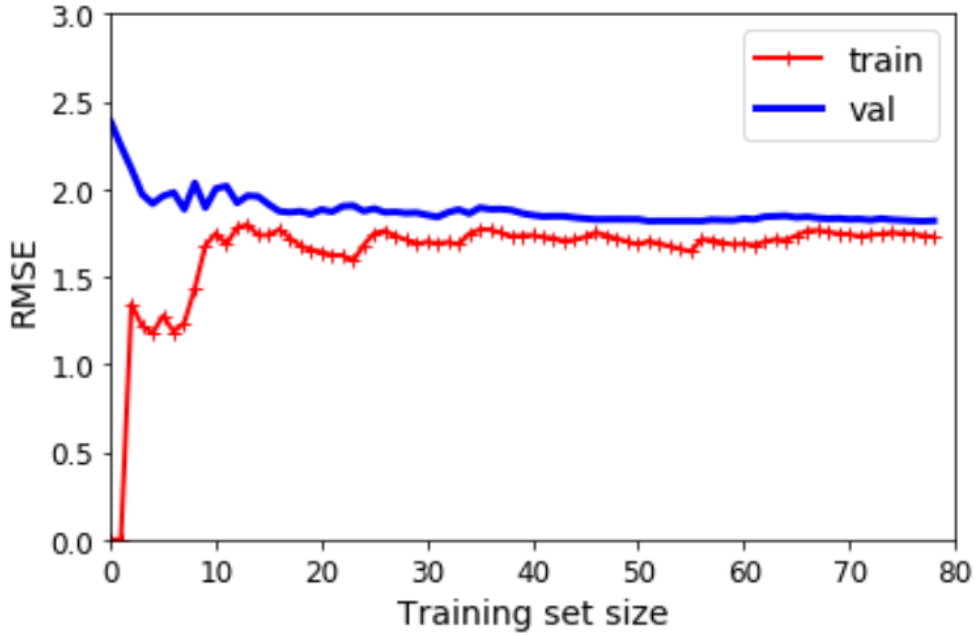
Learning curves – recognizing overfit

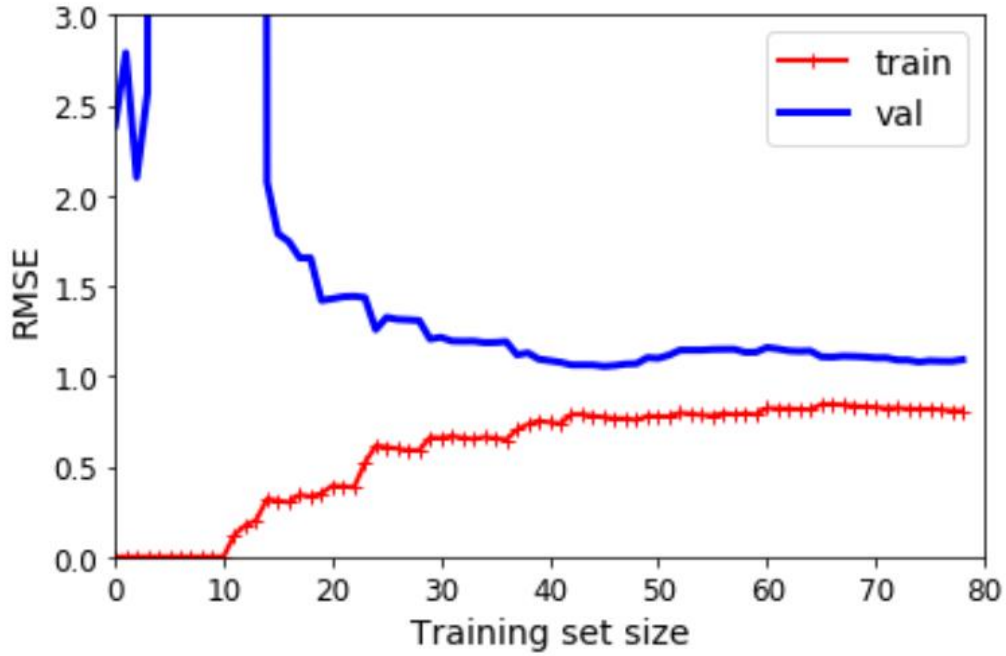
- Relatively good performance RMSE on the training set and a lot worse both the validation set
- Performance RMSE on both validation and training sets are less compareable



Learning curves – comparing underfit and overfit

- To the left: Underfit situation - aka high bias
- To the right: Overfit situation – aka high variance



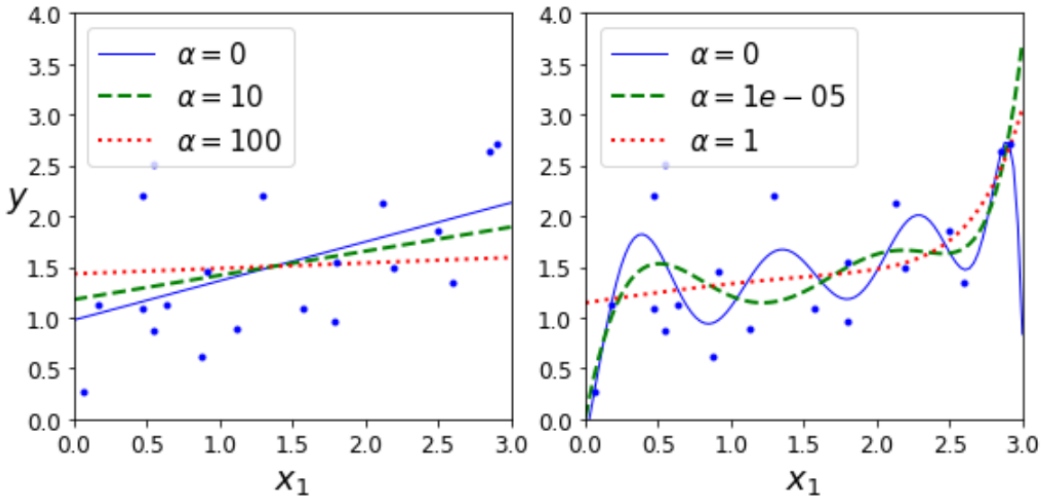


Regularized models

- **Purpose**: Avoiding the overfitting situation
- **Overfitting**: Model fits training set well, but fits the validation set badly
- **Polynomial models:** Reduce polynomial degrees
- **Linear models**: Constrain the model parameters $\theta_1,\ldots,\theta_n$ - That is reducing slope-changes

Ridge Regression

- Adding a penalty to the cost function MSE during learning only
- Keeps models weights as small as possible
- Different learning conditions depending on ‘penalty factor’ α
- Linear model to the left – Polynomial model to the right

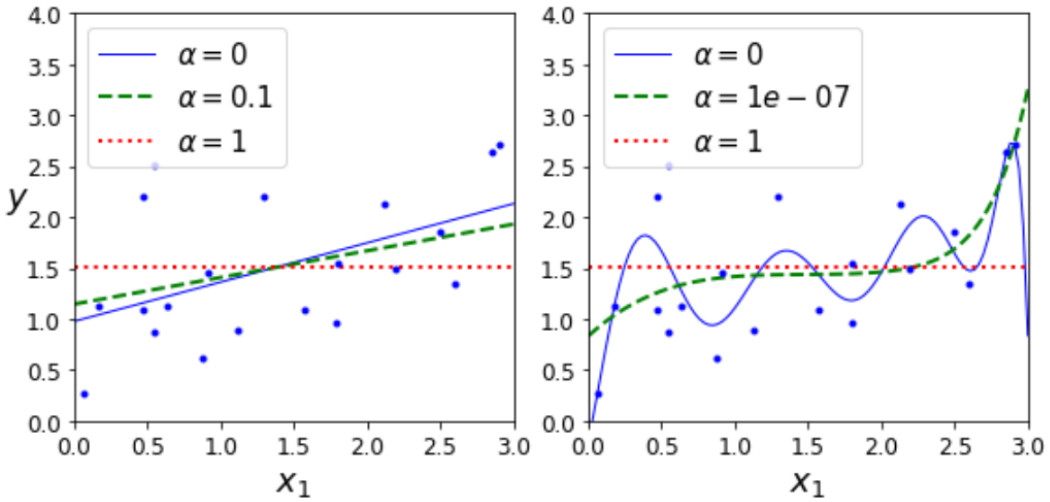


Equation 4-8. Ridge Regression cost function

$$J(\theta) = \text{MSE}(\theta) + \alpha \frac{1}{2} \Sigma_{i=1}^{n} \theta_i^2$$

Lasso Regression

- Adding a penalty to the cost function MSE during learning only
- Eliminates the least important features
- Different learning conditions depending ‘penalty factor’ α
- Linear model to the left – Polynomial model to the right



Equation 4-10. Lasso Regression cost function

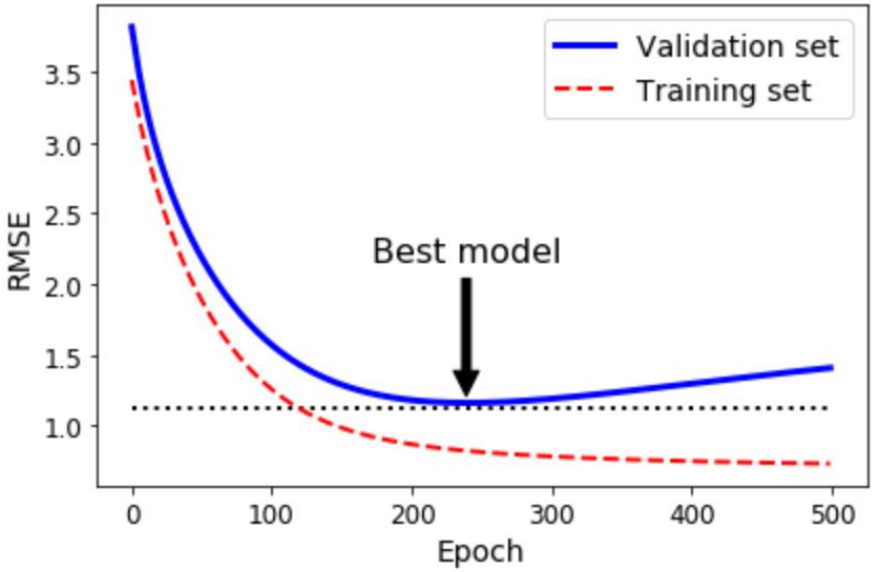
$$J(\theta) = \mathrm{MSE}(\theta) + \alpha \sum_{i=1}^{n} |\theta_i|$$

Elastic Net

- Is a combination of the Ridge and Lasso regression

Early Stopping – stop learning when validation is best

- Error RMSE when predicting on training set approaches zero
- Error RMSE when predicting on the validation set reaches the minimum
- The model is best at this minimum
- If proceeding further, we will recognize the overfit situation



Early stopping – SGDRegressor example

Available parameters:

- **early_stopping : bool**, default=False
  Whether to use early stopping to terminate training when validation score is not improving. If set to True, it will automatically set aside a fraction of training data as validation and terminate training when validation score is not improving by at least the value of tol (i.e. tolerance) for n_iter_no_change consecutive epochs.
- **n_iter_no_change** : int, default=5
  Number of iterations with no improvement to wait before early stopping.
- **validation_fraction : float**, default=0.1
  The proportion of training data to set aside as validation set for early stopping. Must be between 0 and 1. Only used if early_stopping is True.

***Source*: scikit-learn.org**

Learning code

- **Time to take another look at the code**