Lecture 06: *"Regression"*

Book Chapter

Please read the *Chapter 04* **"Training Models"** till p. **134**, from Chapter 06 "Decision Trees" the section "Regression" and from Chapter 07 "Ensemble Learning" the section "Gradient Boosting". Answer the following Questions.

Keep in mind: If you answer these questions and write a detailed summary, you won't need to read these chapters again while preparing for the exam

Video Nugget

For a deeper understanding of Gradient Descent you can watch this video.

Questions

Classic regression - Linear Regression and Polynomial Regression (p. 111-117, 128-131)

1. What different ways are there to train a Linear Regression model? Which method provides the best results?
2. What is the common performance measure for regression?
3. Which is more computationally intensive for the closed form calculation, doubling the number of instances or doubling the number of features?
4. How does Polynomial Regression work?

Regression with Machine Learning Algorithms - Decision Tree and Random Forest (p. 183-184, 203-207)

1. Describe the difference between a decision tree for classification and for regression.
2. What types of functions can a decision tree regressor approximate well?
3. What is the plot of the function of a decision tree regressor for one input variable?
4. Is there a random forest regressor or can you just use the gradient tree boosting regressor for the regression?
5. How does the gradient tree boosting regressor work?

Gradient Descent Algorithms (p. 118-128, 130-134) and Early stopping (p. 141-142)

1. Please summarize the steps done using *Gradient Descent*.
2. What can be a problem with setting the learning rate to high or to low?
3. What is a problem with *Batch Gradient Descent* (or better "*Full Gradient Descent*")?
4. What is a problem with *Stochastic Gradient Descent*?
5. What is the intuition behind a *learning schedule*?
6. What is the intuition behind *Mini-batch Gradient Descent*?
7. How can you tell from the learning curves if a model underfits or overfits?
8. How does early stopping work?

Homework Assignment

Code the task given in Regression Task.

Answers

Classic regression - Linear Regression and Polynomial Regression

1. Closed-form equation & Gradient Descent. The closed form equation gives directly the optimal values for the linear regression function, the gradient Descent needs several steps to approximate them.
2. It is the RMSE or the MSE.
3. Doubling the number of features, the effort is quadratic. For doubling the number of instances the effort is just linear.
4. For each feature the quadratic values are added (and if necessary also up to degree 3, 4,...)

Regression with Machine Learning Algorithms - Decision Tree and Random Forest

1. In regression the split is set to the smallest possible mse. In the leaf a numerical value is predicted instead of a class.
2. No special function type. It can approximate all types.
3. It is a stepfunction with different stepsizes and heights.
4. There exists also a general Random Forest Regressor. It is not used in the book.
5. It fits a new predictor to the residuals of the first tree and so on.

Gradient Descent Algorithms and Early stopping

1. Gradient Descent: for a starting point the direction of the steepest slope is computed, the step goes in this direction (the length of the steps depend on the chosen learning rate). This procedure is done in every step.
2. Too high: The steps are too long, the algorithm skips over the minimum. Too low: The algorithm needs too many steps.

3. The gradient is calculated from the performance measure, in regression this is the mse. To do this, one has to compute the difference between the actual and the predicted value for each instance. So all the dataset data must be used for calculation at each gradient descent step
4. Picking just one dataset instance -> bouncing around
5. Slowing down the step size resp. The learning rate
6. using mini-batches, this is a small random subset of instances, instead of whole batch or just one sample
7. Underfit: plateau, not getting better, Overfit: train loss much lower than val loss
8. The performance measure for the training and the validation set is plotted in dependence of the number of epochs. When the validation curves reaches its minimum you know that the training set starts to overfit. Therefore the epochs are stopped.

General Points:

Linear and polynomial Regression fits the curve to a given form of the curve (e.g. a straight line or a parabel). DecisionTreeRegressor and RandomForestRegressor don't have a given form, they are more flexible.

Linear and polynomial regression are thus used to calculate general trends or when it is known that this is the underlying logic e.g. from physics. If a scatter plot gives a different impression, use DecisionTreeRegressor or RandomForestRegressor.

Regularized Linear Models (Ridge, Lasso und Elastic Net) don't fit better then Linear or Polynomial Models. They are used to regularize the model, that means to get an easier model or a model with smaller parameters. It is paid with a less accuracy.

Logistic regression is - contrary to its name - a classifier. It uses the idea of regression to perform.

Gradient descent is an optimization algorithm. It works in the background of many algorithms. It does not belong to regression.

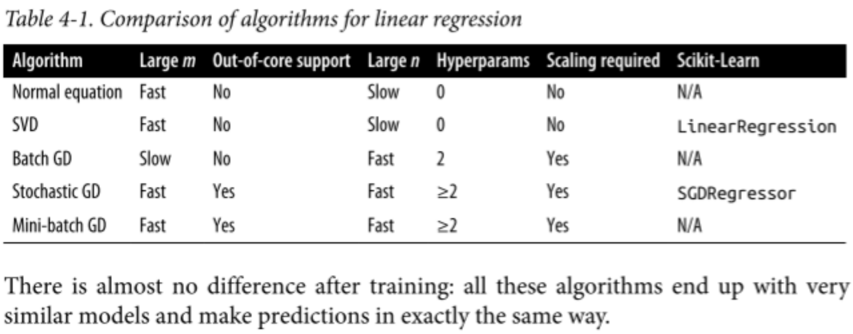

Table 4-1. Comparison of algorithms for linear regression

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	N/A
Stochastic GD	Fast	Yes	Fast	≥2	Yes	`SGDRegressor`
Mini-batch GD	Fast	Yes	Fast	≥2	Yes	N/A

There is almost no difference after training: all these algorithms end up with very similar models and make predictions in exactly the same way.

Geron, p. 128

M = number of training instances

N = number of features