Linear Models and ML Fundamentals I

Machine Learning

Denis Litvinov

September 15, 2022

Table of Contents

Prerequisites

- Linear Algebra
- Analysis
- Probability Theory
- Statistics

Course Content

1. Linear models, ML fundamentals
2. Decision Trees, Ensembles
3. Unsupervised Learning: Clustering, Dimension reduction
4. Neural Networks, FFNN
5. Embeddings, Transfer Learning
6. k-NN

Evaluation

- HW1 - Linear Models
- HW2 - Ensembles
- HW3 - Unsupervised Learning
- HW4 - FFNN
- HW5 - Transfer Learning
- exam - in written form, 2 theoretical questions

$$Total = 0.8 * 0.2(HW1 + HW2 + HW3 + HW4 + HW5) + 0.2 * exam$$

Literature I

[1] Goodfellow I. Deep Learning. MIT Press, 2016

[2] Bishop C.M. Pattern Recognition and Machine Learning. Springer, 2006

[3] Hastie, T. The elements of statistical learning : data mining, inference, and prediction. Springer, 2009

[4] Sheldon A. Linear Algebra Done Right. Springer, 3d edition, 2015

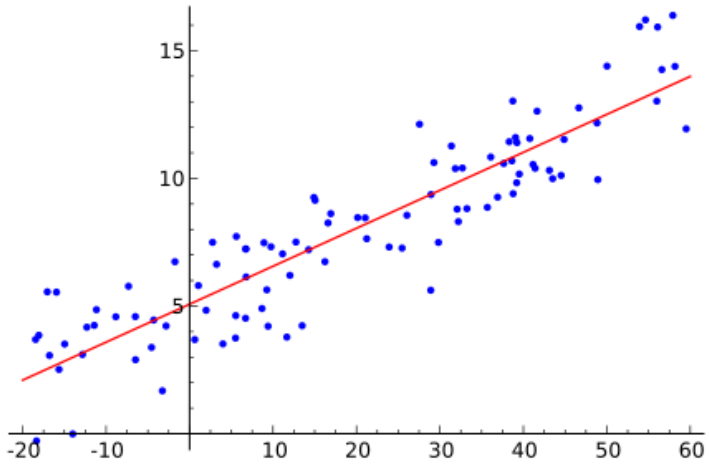
[5] Schapire. Boosting: Foundation and Algorithms. MIT Press, 2012

Taxonomy

- Supervised vs Unsupervised vs Reinforcement Learning
- Supervised Learning: Classification vs Regression
- Discriminative vs Generative Model
- Structured vs Unstructured Prediction

Task Formulation

Given dataset $\{(x_i, y_i)\}_{i=1}^{N}$ of i.i.d. objects
Or, equvalently given:
$X \in R^{NxD}$ - feature matrix, where d is dimension of feature space and N - number of objects.
$Y \in R^{N}$ - target vector.
We want to find such algorithm $h \in H$ that $h(x) = \hat{y}$ "assigns for each object the right target value".



Loss Functions

$Loss : RxR \rightarrow R$ - loss function, that evaluates how bad our prediction for particular object are.
Some loss functions:

Mean Squared Error

$$Loss(\hat{y}_i, y_i) = (\hat{y}_i - y_i)^2$$

Mean Absolute Error

$$Loss(\hat{y}_i, y_i) = |\hat{y}_i - y_i|$$

Mean Absolute Percentage Error

$$Loss(\hat{y}_i, y_i) = \frac{|\hat{y}_i - y_i|}{y_i}$$

Quality Metrics

Coefficient of Determination

$$R^2(\hat{y}, y) = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \frac{1}{N}\sum_j y_j)^2}$$

Root Mean Squared Error

$$RMSE(\hat{y}, y) = \sqrt{\frac{1}{N}\sum_i (\hat{y}_i - y_i)^2}$$

Mean Absolute Error

$$MAE(\hat{y}, y) = \frac{1}{N}\sum_i |\hat{y}_i - y_i|$$

Linear Regression

Here we explore linear regression with MSE loss

$$L_{MSE}(y, \hat{y}) = \frac{1}{N}\sum_{i=1}^{N}(y - \hat{y})^2$$

Usually we write

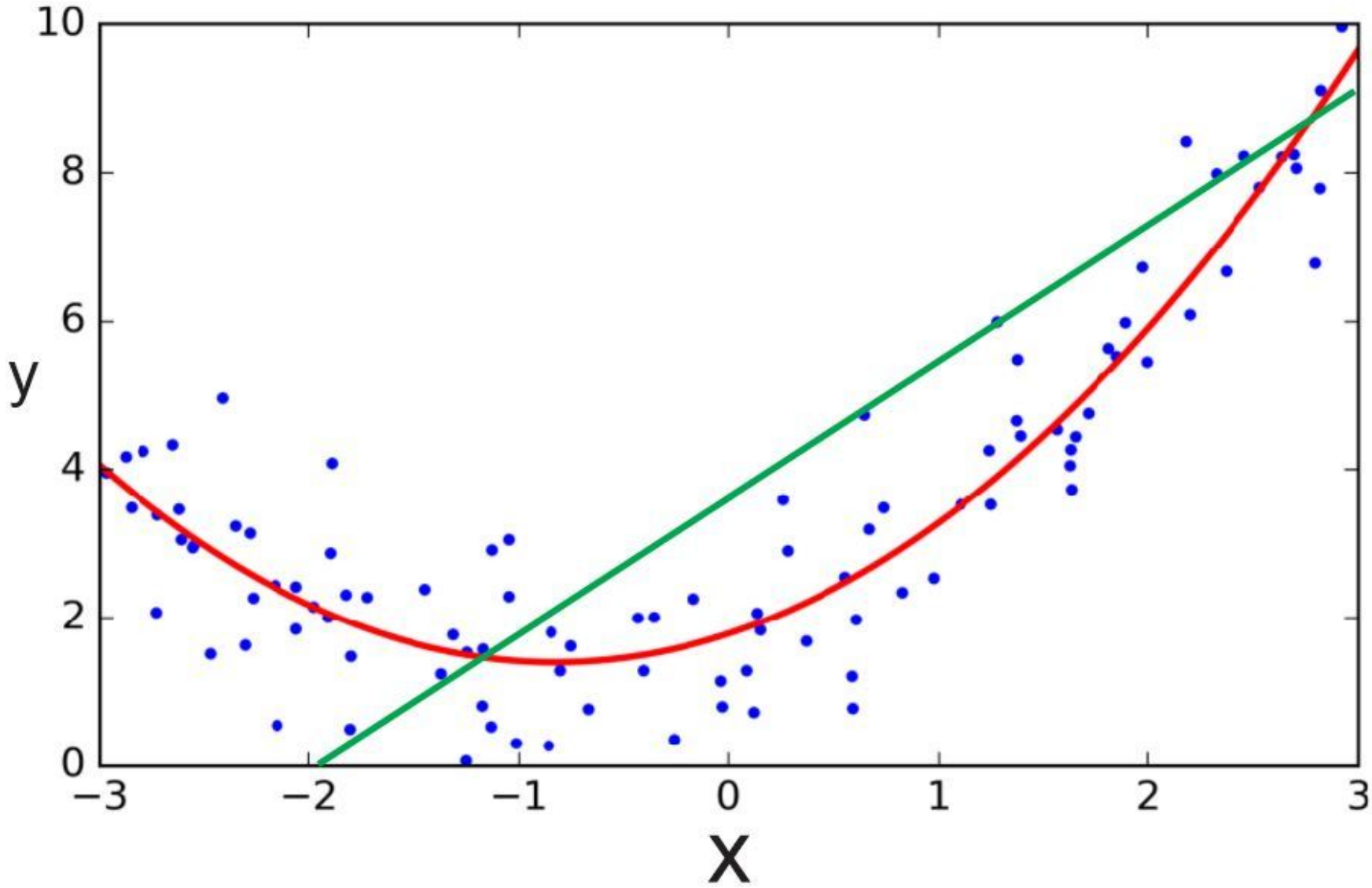
$$\hat{y} = w^T x + b$$

where w, b - model weights, and b is called *intercept*
For future convenience add b into vector x

$$(x)^T \rightarrow (1, x)^T$$

$$\hat{y} = w^T x$$

Polynomial Regression 1



Polynomial Regression 2

What if we can't approximate data with linear model? Use polynomial regression

$$\phi_n(x) = (1, x, x^2, x^3, .., x^n)^T$$

For example,

$$\phi_2(1, x_1, x_2) \rightarrow (1, x_1, x_2, x_1^2, x_1 x_2, x_2^2)$$

And this way can still use linear model

$$\hat{y} = w^T \phi_n(x)$$

However, need to choose the right polynomial power.

Analytic solution of MSE linear regression 1

$$\hat{y} = Xw$$

$$\begin{aligned} L_{MSE}(y, X) &= \frac{1}{N}(y - Xw)^T(y - Xw) \\ &= \frac{1}{N}(y^T - (Xw)^T)(y - Xw) \\ &= \frac{1}{N}(y^T y - y^T(Xw) - (Xw)^T y + (Xw)^T(Xw)) \\ &= \frac{1}{N}(y^T y - 2y^T(Xw) + (Xw)^T(Xw)) \end{aligned}$$

Analytic solution of MSE linear regression 2

$$\frac{\partial tr(BA)}{\partial A} = B^T$$
$$\frac{\partial tr(A^T B)}{\partial A} = B$$
$$\frac{\partial tr(A^T A)}{\partial A} = 2A$$
$$\frac{\partial tr(A^T BA)}{\partial A} = AB + A^T B$$
$$A, B \in R^{NxM}$$

Analytic solution of MSE linear regression 3

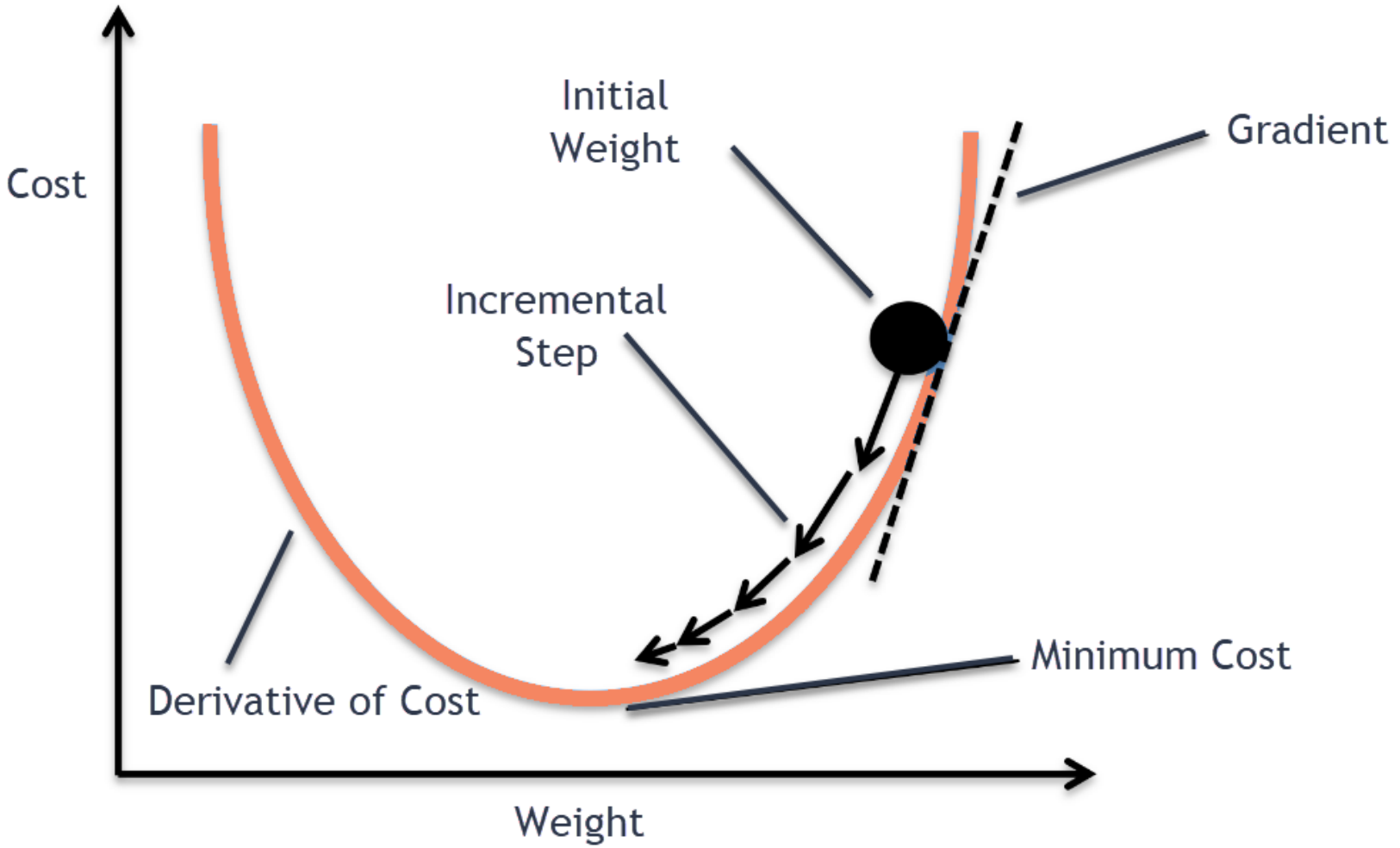
$$\frac{\partial L_{MSE}}{\partial w} = \frac{1}{N}(-2X^T y + 2X^T Xw) = 0$$

$$X^T Xw = X^T y$$

$$w = (X^T X)^{-1} X^T y$$

Properties of analytic solution depends on $(X^T X)^{-1}$
Remember, that if $det(A) \to 0$ then A^{-1} is numerically unstable

Iterative solution 1



Iterative solution 2

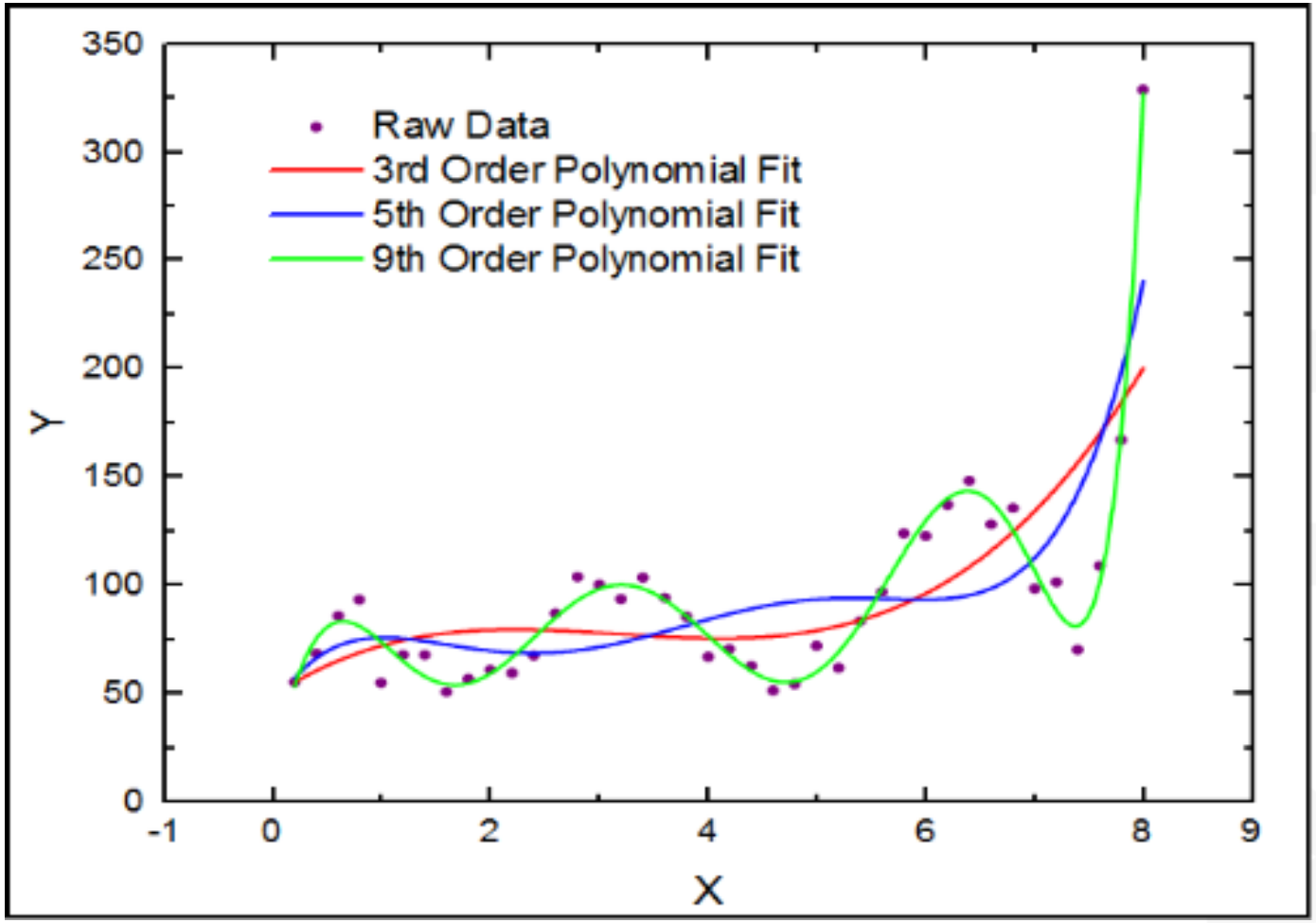
As we observed previously

$$\nabla_w L_{MSE} = \frac{1}{N} X^T (Xw - y)$$

1. $w^{(0)}$ = random init
2. at time t $w^{(t)} = w^{(t-1)} - \alpha \nabla_w L_{MSE}(w^{(t-1)})$
3. until convergence $||\nabla_w L_{MSE}(w^{(t-1)})|| < \epsilon$ OR $||w^{(t)} - w^{(t-1)}|| < \epsilon$ OR number of iterations exceeds predefined maximum

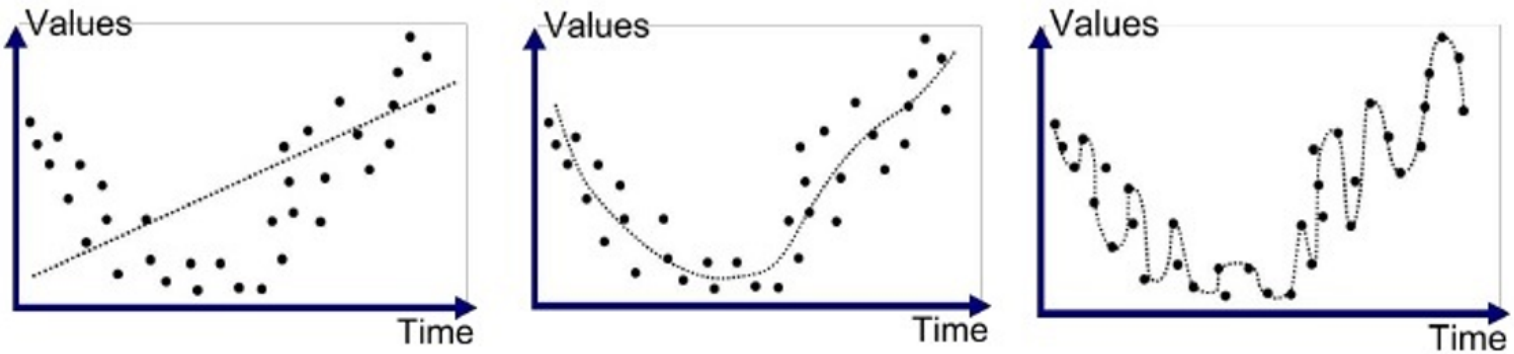
No issues of $(X^T X)^{-1}$ numeric stability!

What curve fits the data the best?



Overfitting

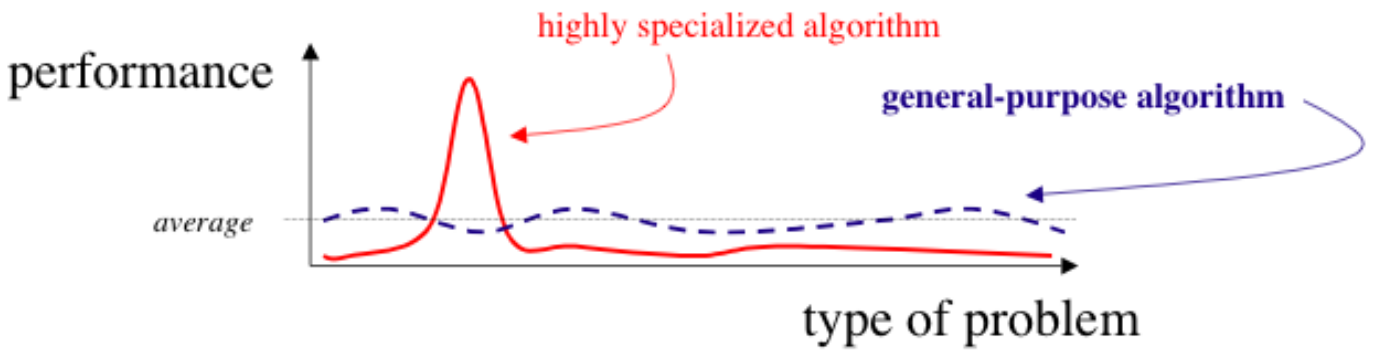
Overfitting is a situation, when a model fitted on a train dataset shows worse perfomance on a test dataset.
It corresponds to the fact, that model learns the given dataset but do not generalize to unseen data from the same distribution.
Every model does overfit!



Underfitted | Good Fit/Robust | Overfitted

No Free Lunch Theorem

The No Free Lunch Theorems state that any one algorithm that searches for an optimal cost or fitness solution is not universally superior to any other algorithm.
"If an algorithm performs better than random search on some class of problems then in must perform worse than random search on the remaining problems." (No Free Lunch Theorems for Optimisation)
How that affects machine learning? **Every machine learning algorithm explicitly or implicitly implies some assumptions made about observed data**. So by contradicting these assumptions for every algorithm we create such dataset, where it achieves bad performance.



Regularization

General Form

$$L_{reg}(y, X, w) = L(y, X) + \lambda R(w)$$

where
$R(w)$ - regularization term
λ - coef of regularization (regularization strength)
In linear modes we usually use L_p norm regularization:

$$R(w) = ||w||_p^p$$

For MSE with L_2 regularization

$$L_{MSE} = \frac{1}{N}||y - Xw||_2^2 + \frac{\lambda}{2}||w||_2^2$$

$$\nabla_w L_{MSE} = \frac{1}{N}X^T(Xw - y) + \lambda w$$

L_2 Regularization as stabilization of matrix inverse

From analytic solution we have

$$\nabla_w L_{MSE} = \frac{1}{N} X^T (Xw - y) + \lambda w = 0$$

Up to scaling factor λ

$$\nabla_w L_{MSE} = X^T (Xw - y) + \lambda w = 0$$

$$(X^T X + \lambda I) w = X^T y$$

$$w = (X^T X + \lambda I)^{-1} X^T y$$

Thus we have stabilization of matrix inverse

$$(X^T X)^{-1} \to (X^T X + \lambda I)^{-1}$$

Also called Tikhonov regularization.

L_2 as Gaussian prior on weights 1

Bayesian view.
We have samples $\{(x_i, y_i)\}_{i=1}^N$ from some distribution $P(x, y)$
Suppose

$$y_i = w^T x_i + \epsilon$$

, where

$$\epsilon \sim N(0, \sigma^2)$$

Thus we can construct likelihood function (remember Maximum Likelihood Estimation)

$$p(y_1, .., y_N | x_1, .., x_N) = \prod_{i=1}^{N} N(y_i | w^T x_i, \sigma^2)$$

where
$N(y_i | w^T x_i, \sigma^2)$ is a Gaussian distribution with mean $w^T x_i$ and variance σ^2

L_2 as Gaussian prior on weights 2

$$N(y_i|w^T x_i, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{(y_i - w^T x_i)^2}{2\sigma^2}}$$

Now imposing Gaussian prior on weights $w \sim N(0, \lambda^{-1})$

$$p(y_1, .., y_N|x_1, .., x_N) = \prod_{i=1}^{N} N(y_i|w^T x_i, \sigma^2) N(w|0, \lambda^{-1})$$

by MLE we would like to maximize

$$\log \prod_{i=1}^{N} N(y_i|w^T x_i, \sigma^2) N(w|0, \lambda^{-1}) \rightarrow \max_w$$

L_2 as Gaussian prior on weights 3

$$\sum_{i=1}^{N} \log N(y_i|w^T x_i, \sigma^2) + N * \log N(w|0, \lambda^{-1}) \to \max_w$$

$$-\sum_{i=1}^{N} \log N(y_i|w^T x_i, \sigma^2) - N * \log N(w|0, \lambda^{-1}) \to \min_w$$

$$-\sum_{i=1}^{N}(-\frac{1}{2\sigma^2}(y_i - w^T x_i)^2) - N * (-\frac{1}{2\lambda^{-1}} w^T w) \to \min_w$$

$$\frac{1}{N\sigma^2}\sum_{i=1}^{N}(y_i - w^T x_i)^2 + \frac{\lambda}{2} w^T w \to \min_w$$

Up to scaling factor σ^2 we have familiar MSE loss with L_2 regularization

$$\frac{1}{N\sigma^2}(y - Xw)^T(y - Xw) + \frac{\lambda}{2}||w||_2^2 \to \min_w$$

L_2 as Gaussian prior on weights 4