Data^{X}

About Me:

Underfitting / Overfitting, Polynomial Regression & Regularization

Alexander Fred-Ojala

Alexander Fred-Ojala
Research Director, Data Lab, SCET, UC Berkeley
afo@berkeley.edu

Underfitting

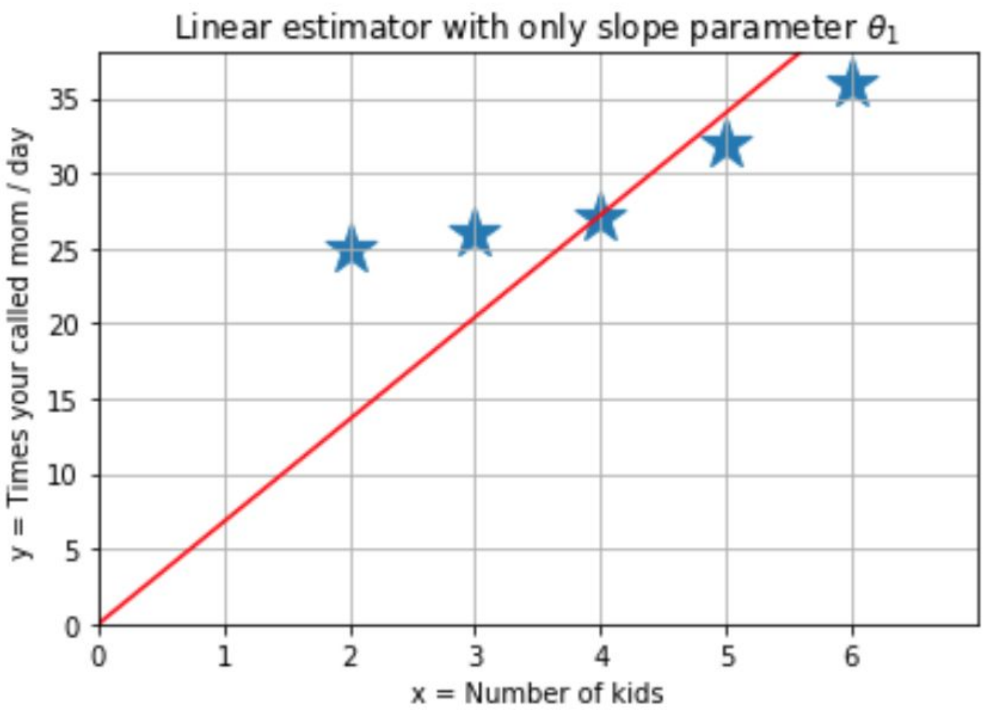
Underfitting / High bias

Characteristics of underfitting:

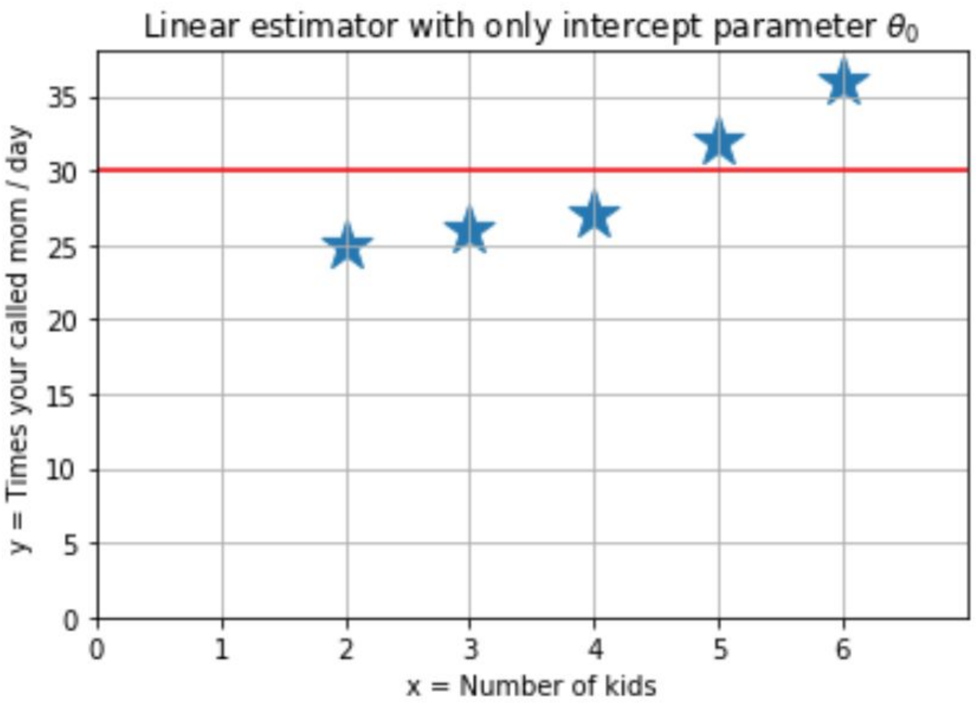
- The model is too simple (few degrees of freedoom)
- Low variance, but high bias
- Strong preconception about model parameters

Underfitting / High bias

Examples of underfitting: One degree of freedom is not enough

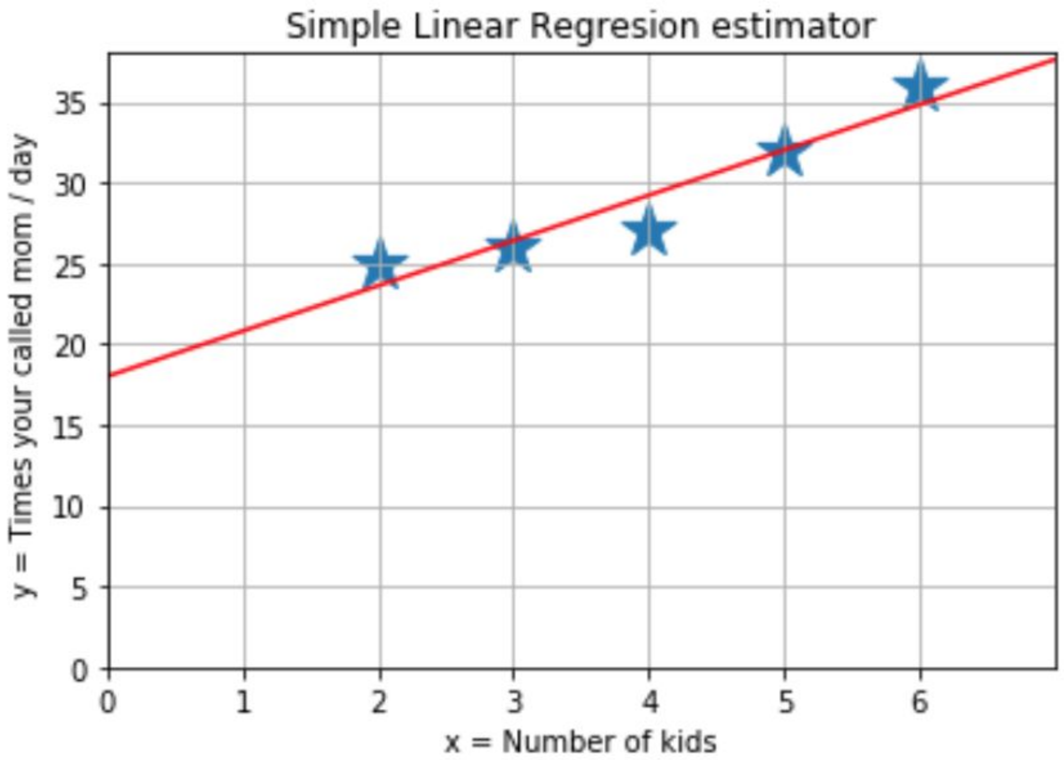


$$\hat{y} = h_\theta(x) = \theta_1 x$$



$$\hat{y} = h_\theta(x) = \theta_0$$

Good model approximation for our data



$$\hat{y} = h_\theta(x) = \theta_0 + \theta_1 x$$

Data X

Underfitting / High bias

How to prevent underfitting:

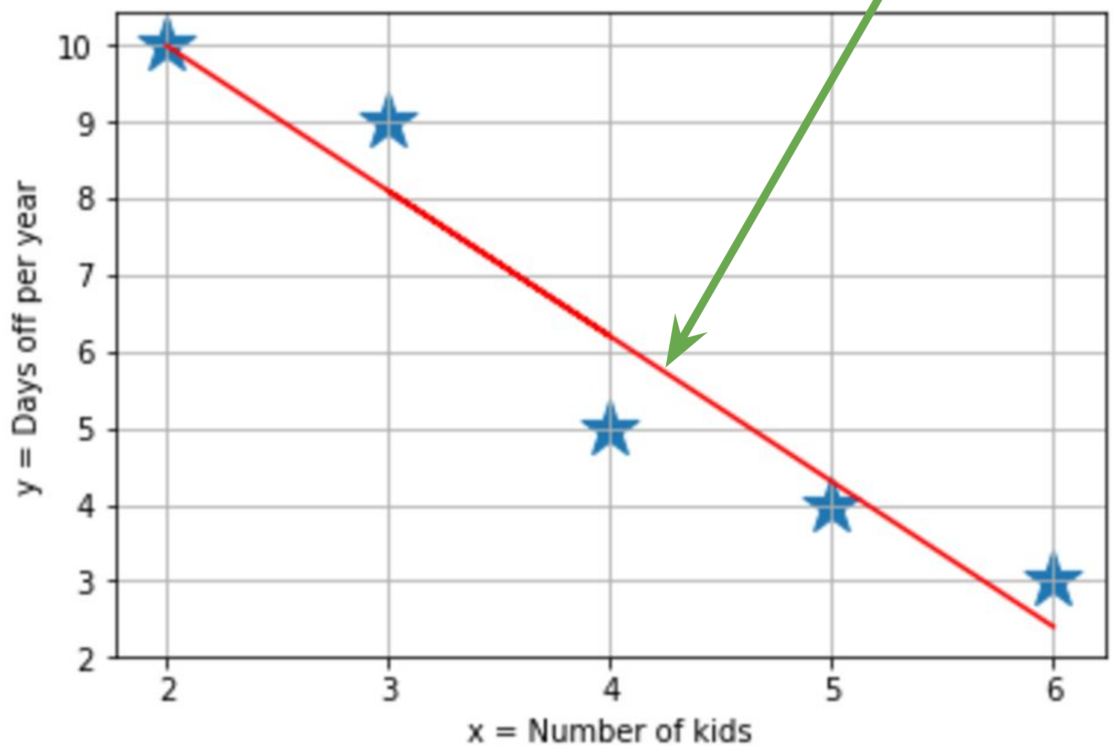
- Fit a more complex model / algorithm
- Construct features from existing ones
- Transform features to increase complexity of the model

Polynomial Regression

(Simple) Linear Regression

$h_\theta(x)$

$h_\theta(x)$



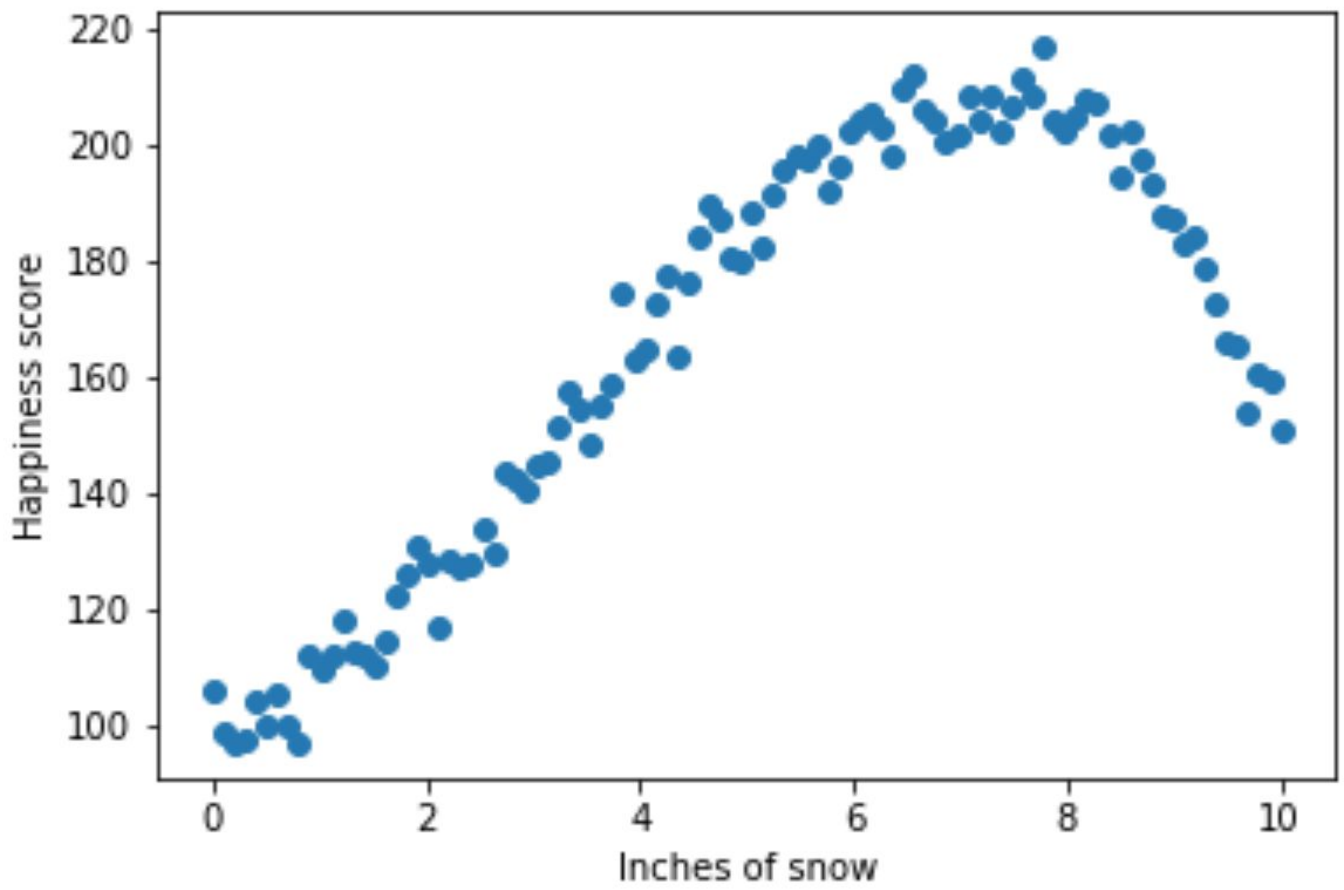
Works when have a

Linear relationship between the dependent and the independent variables

$$\hat{y} = f(x, \theta) = h_\theta(x) = \theta^T x = \theta_0 + \theta_1 x_1$$

Modeling Non-linear relationships

What if we want to model this relation?



Modeling Non-linear relationships

The best Simple Linear Regression Model

Obtained by solving the Normal Equation

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

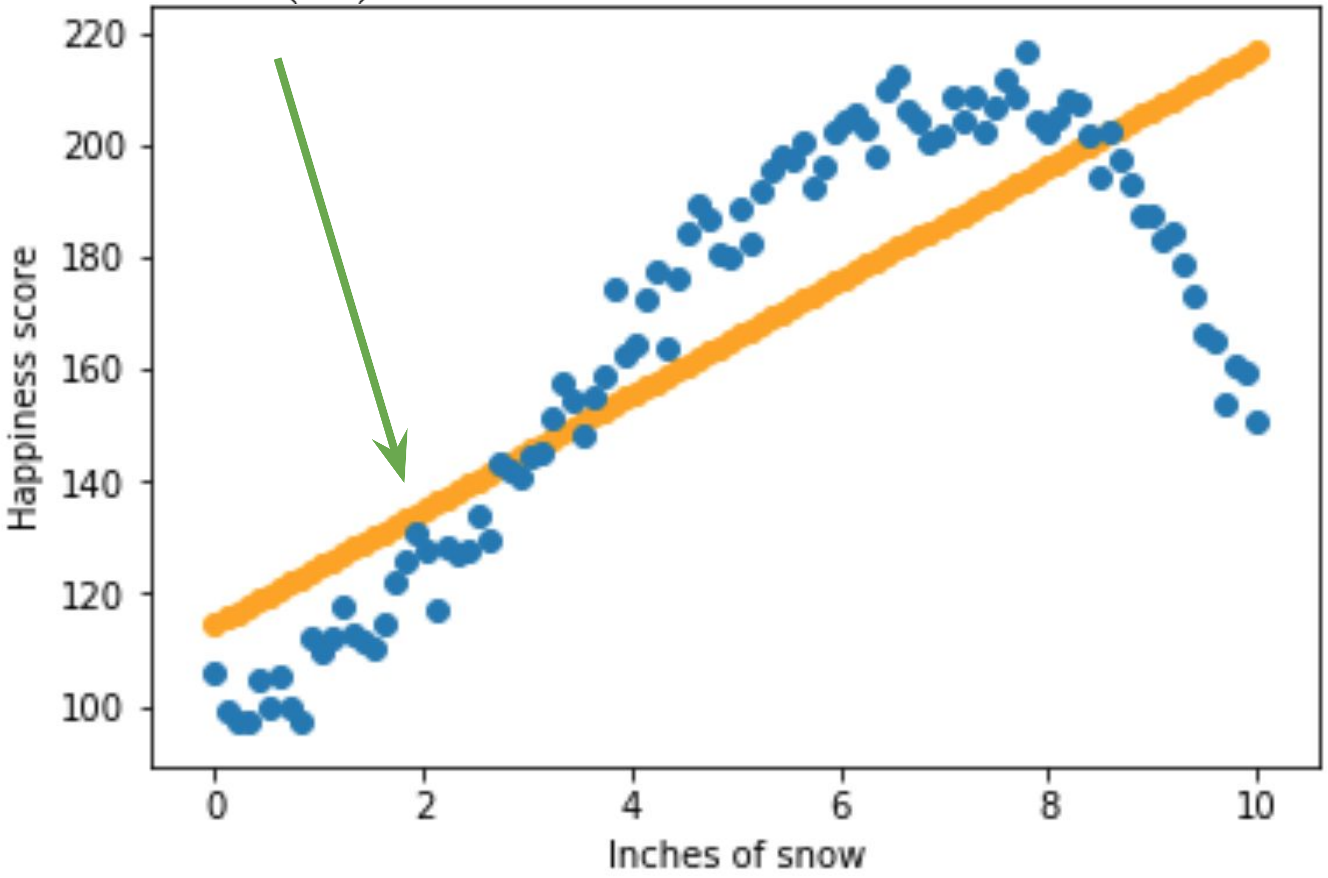
gives us:

$$\hat{y} = h_\theta(x) = \theta_0 + \theta_1 x_1$$
$$\approx 117 + 10x_1$$

We are clearly underfitting!

(Our model has high bias)

$h_\theta(x)$



Polynomial Regression

$$\hat{y} = h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \ldots + \theta_n x^n$$

It is still a Linear Regression model, we have just transformed some of the predictors.

Rewrite the predictors, as:

$$\begin{aligned} x &\rightarrow x_1 \\ x^2 &\rightarrow x_2 \\ &\vdots \\ x^n &\rightarrow x_n \end{aligned}$$

to see that it's still a Linear function for the parameters.

Exponential / Logarithmic / Square root ... Regression

$$\hat{y} = h_\theta(x) = \theta_0 + \theta_1\sqrt{x} + \theta_2 \log(x) + \theta_3 e^x$$

You can also transform the logarithmic, exponential, the square root of predictors etc.

$$\sqrt{x} \rightarrow x_1$$
$$\log(x) \rightarrow x_2$$
$$e^x \rightarrow x_3$$
$$\vdots$$

Since it still can be cast as a multiple linear regression problem we can find the optimal parameters by using the Normal Equations or Gradient Descent!

Polynomial Regression

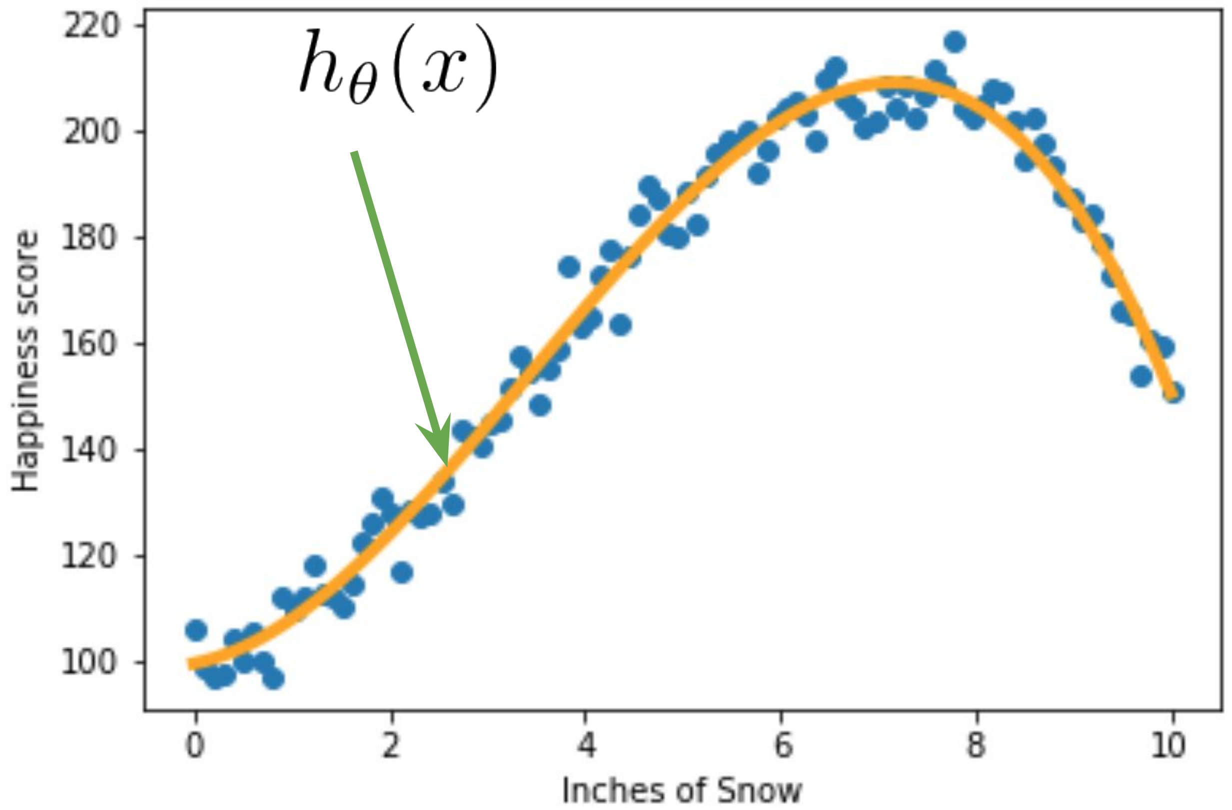
Find the best polynomial function (of degree 3)

Obtained by solving the Normal Equation

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

is given by:

$$\hat{y} = h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3$$
$$\approx 98 + 7x + 4.6x^2 - 0.5x^3$$



This model is a much better fit to our data!

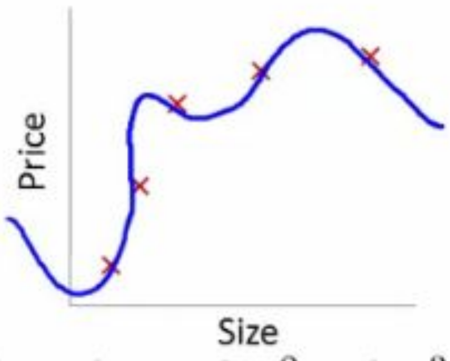
Overfitting

Overfitting

Why don't we fit polynomial functions of very high degrees that always fit our data perfectly so that we get an error that approaches zero?

$$\hat{y} = h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \ldots + \theta_\infty x^\infty \qquad J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2 \rightarrow \mathbf{0}$$

It leads to **overfitting** (we won't predict well on new data that our model hasn't seen)



$$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$$

High variance
(overfit)

Overfitting / High variance

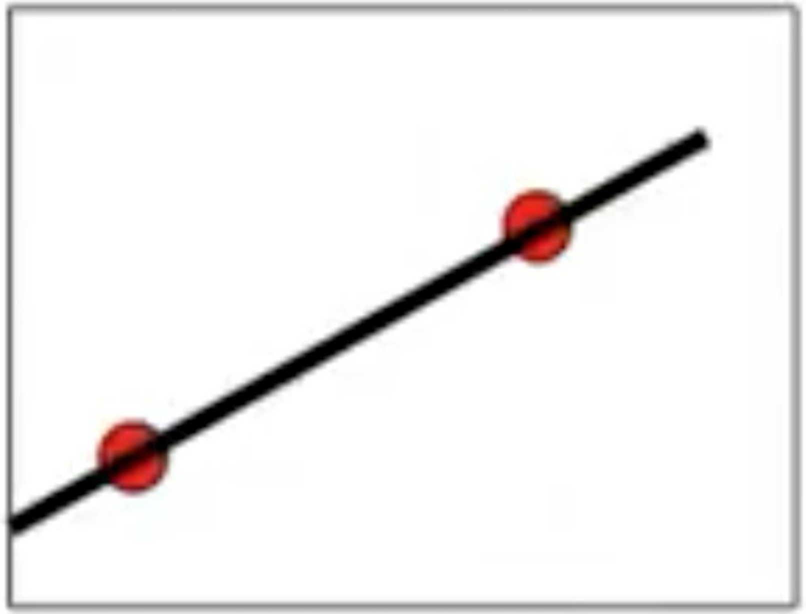
Characteristics of overfitting:

- Our model is too complex (picks up patterns in noise / outliers)
- High variance, but low bias
- The model has too many features, and the parameters are too big
- We are able to *perfectly predict training data, but not test data*
- Small changes in the training data, leads to big change in model parameters

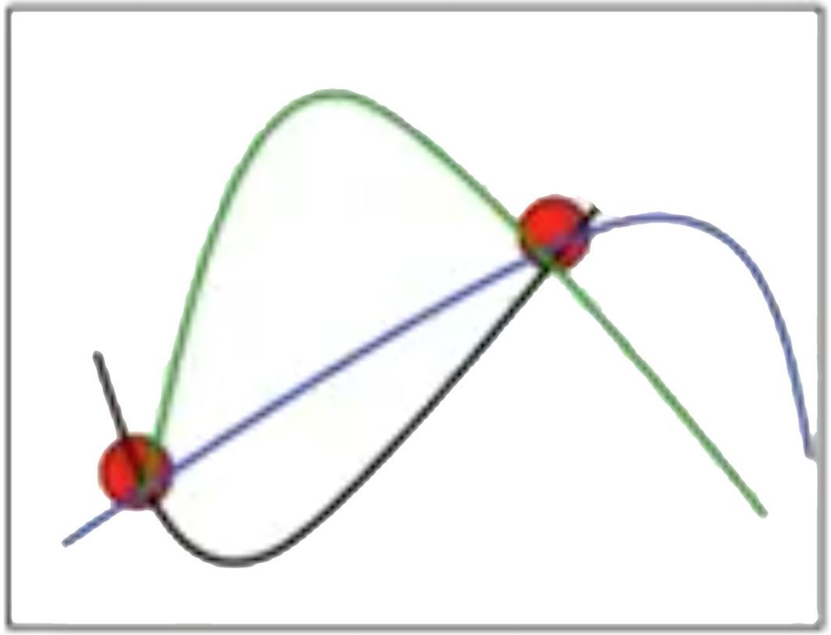
Overfitting / High variance

What model to choose? The simplest one!

Occam's razor theorem



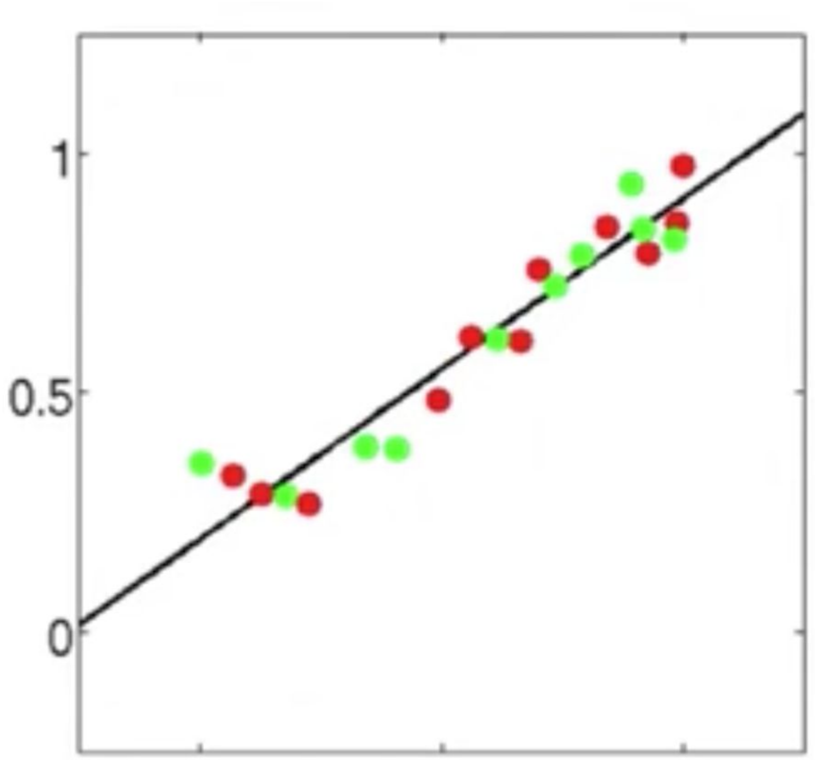
Best (Simple) Model!



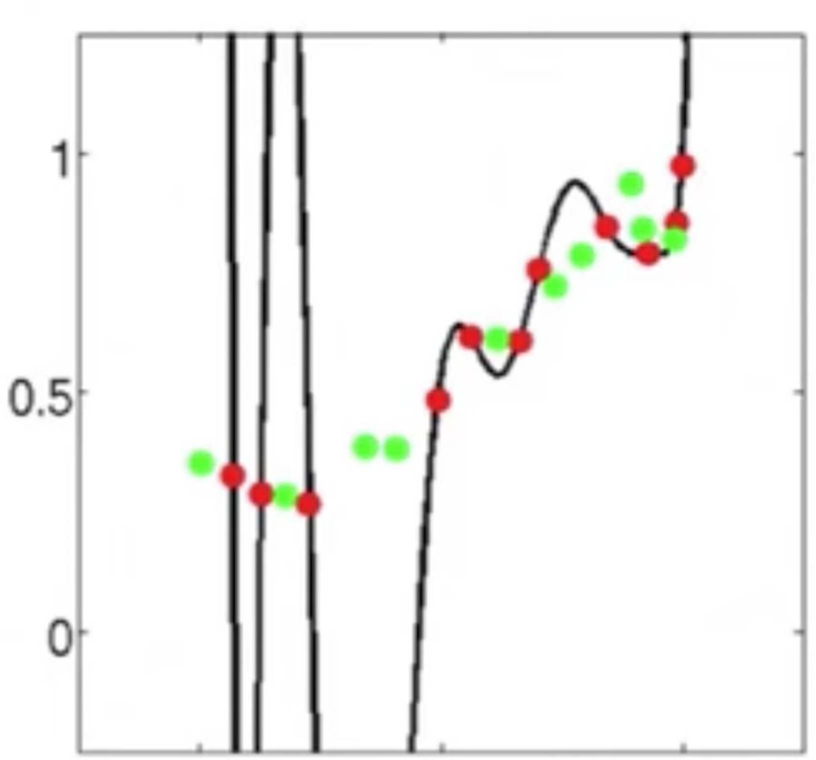
Worse (too complex) models!

Overfitting / High variance

Examples of overfitting: Training data = Red Test data = green



Best (Simple) Model!

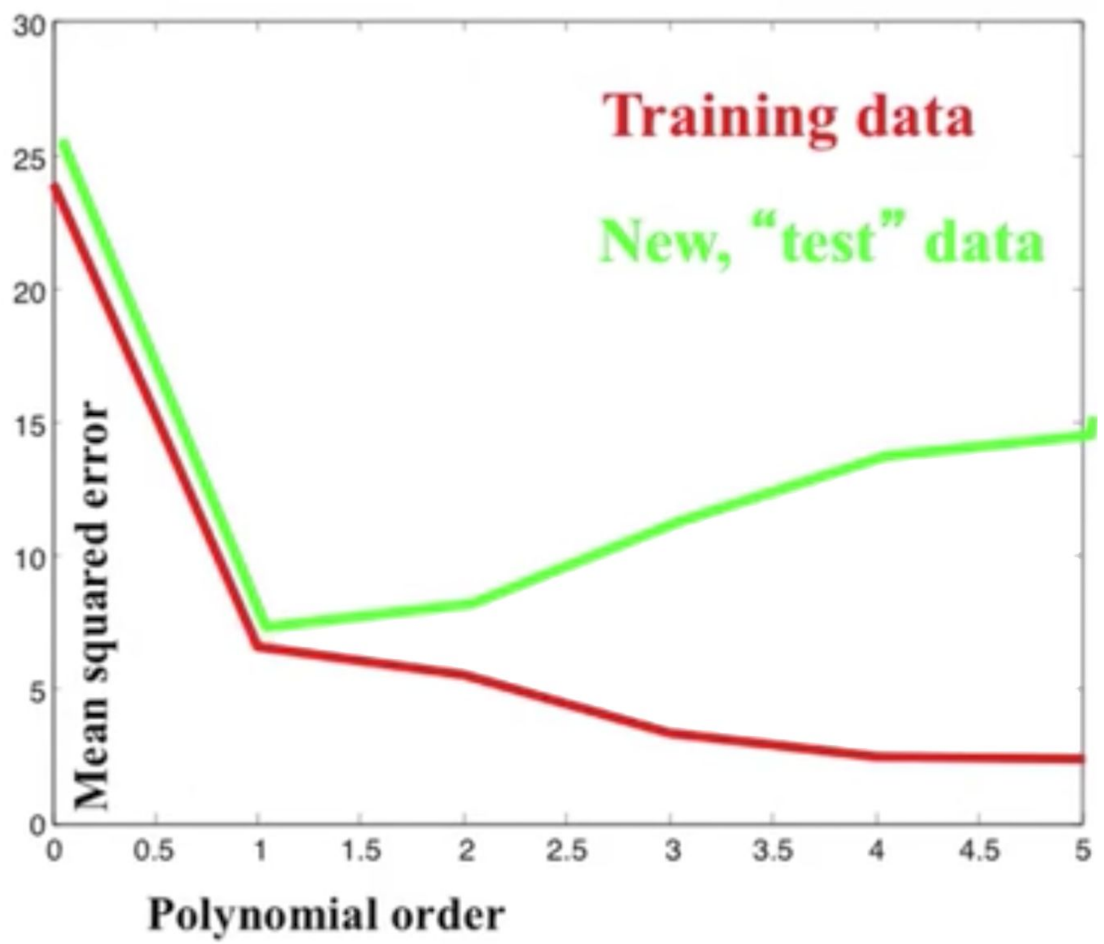


Extremely high variance, bad model!

How to check if you are overfitting

Plot MSE against model complexity

When MSE starts to increase for the test data, that is the point where we are starting to overfit.



Overfitting / High variance

Check if you're overfitting:

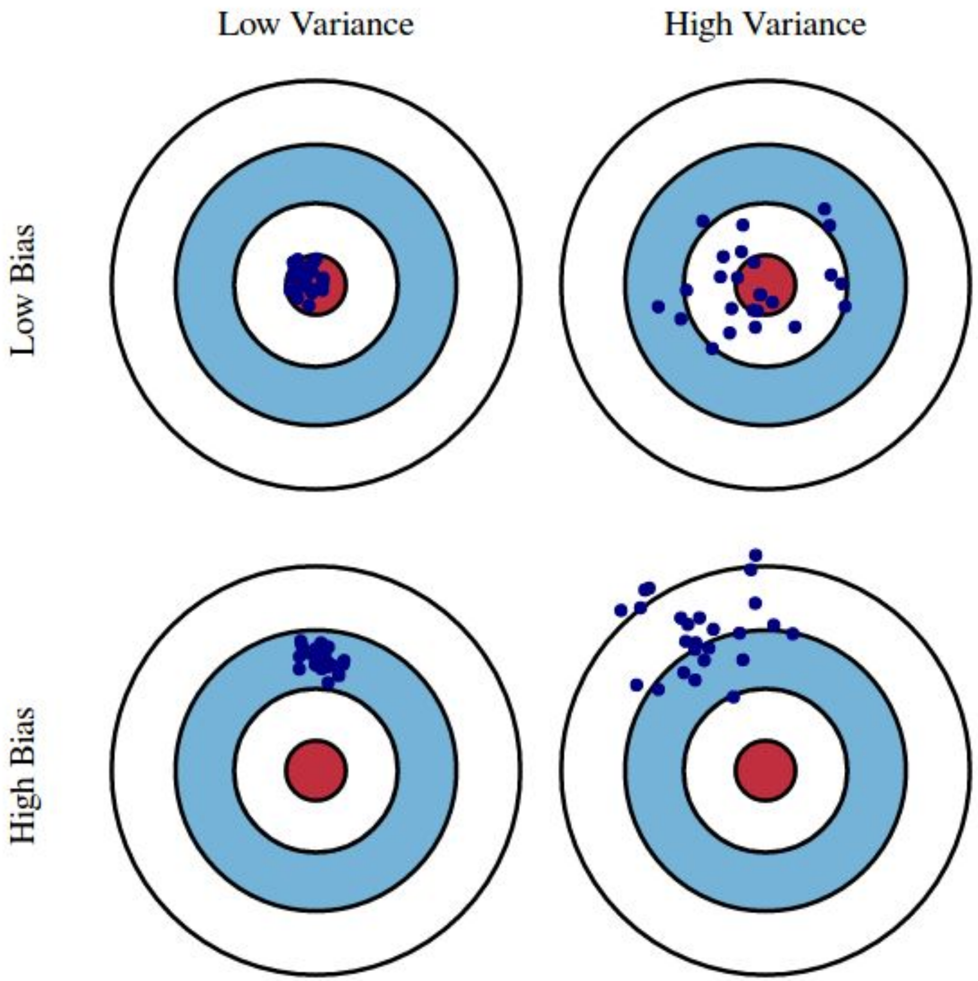
- Use cross-validation / train-test-split and predict on test data
- Plot MSE against model complexity for training and test set, see how the error changes.

Prevent overfitting:

- Exclude some features from the model (feature engineering)
- Introduce a stopping criteria in the optimization algorithm
- ***Regularization!***

Bias (underfitting) - Variance (Overfitting) tradeoff

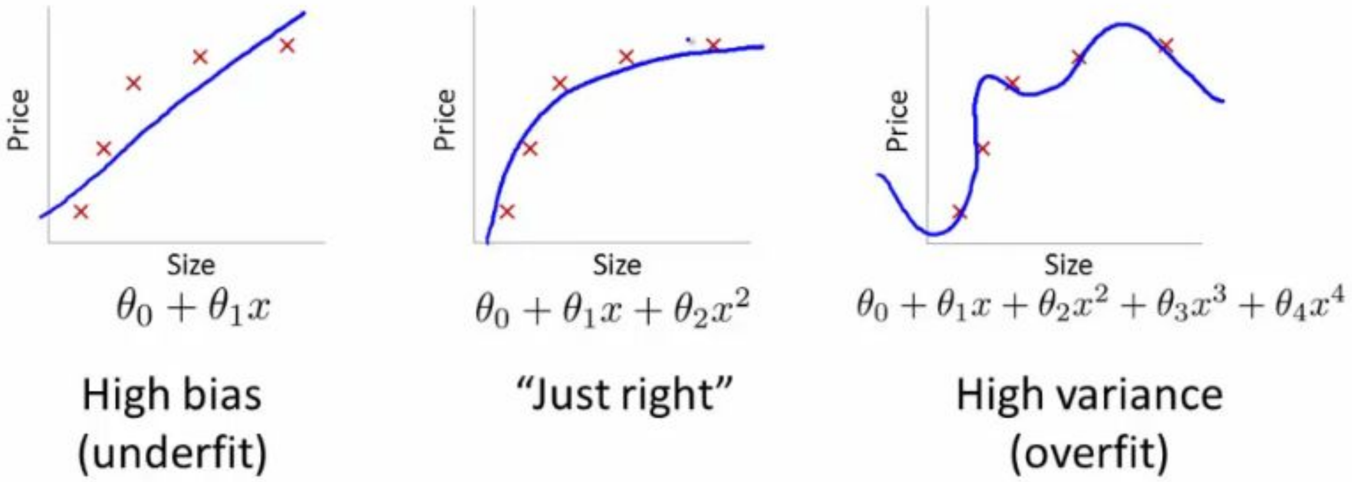
Bias-Variance Tradeoff:



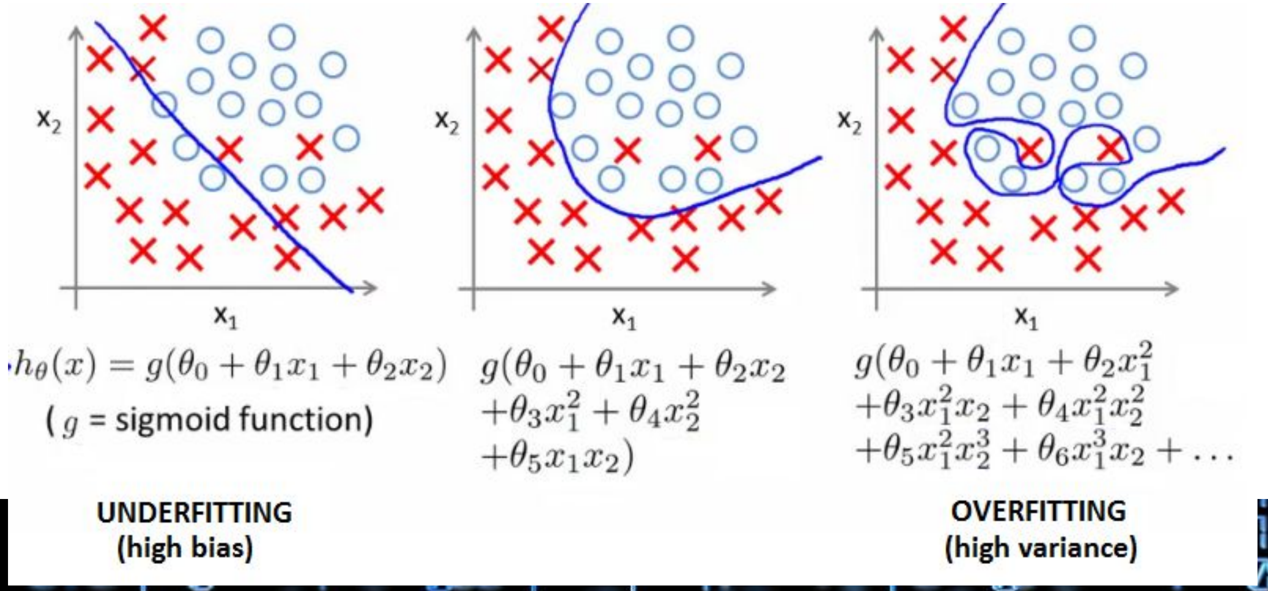
Data X

Bias-Variance Tradeoff:

REGRESSION CASE:



CLASSIFICATION CASE:



Mathematical Derivation

$$y = f(x) + \varepsilon$$

Our goal is to model:

$$\hat{f}(x) \approx f(x)$$

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

- f(x) is the "true" function our data is generated from
- Ɛ is zero-mean Gaussian noise that affects our samples

Bias-Variance Decomposition

$$\mathrm{E}\left[\left(y - \hat{f}(x)\right)^2\right] = \mathrm{Bias}\left[\hat{f}(x)\right]^2 + \mathrm{Var}\left[\hat{f}(x)\right] + \sigma^2$$

where

$$\mathrm{Var}\left[\hat{f}(x)\right] = \mathrm{E}[\hat{f}(x)^2] - \mathrm{E}[\hat{f}(x)]^2$$

$$\mathrm{Bias}\left[\hat{f}(x)\right] = \mathrm{E}\left[\hat{f}(x) - f(x)\right]$$

σ^2 Irreducible error (variance of the noise), lower bound for the model MSE

Data X

Mathematical Derivation

$$y = f(x) + \varepsilon$$

Our goal is to model:

$$\hat{f}(x) \approx f(x)$$

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

- f(x) is the "true" function our data is generated from
- ε is zero-mean Gaussian noise that affects our samples

Proof of the Bias-Variance Decomposition?

$$\mathrm{E}\left[\left(y - \hat{f}(x)\right)^2\right] = \mathrm{Bias}\left[\hat{f}(x)\right]^2 + \mathrm{Var}\left[\hat{f}(x)\right] + \sigma^2$$

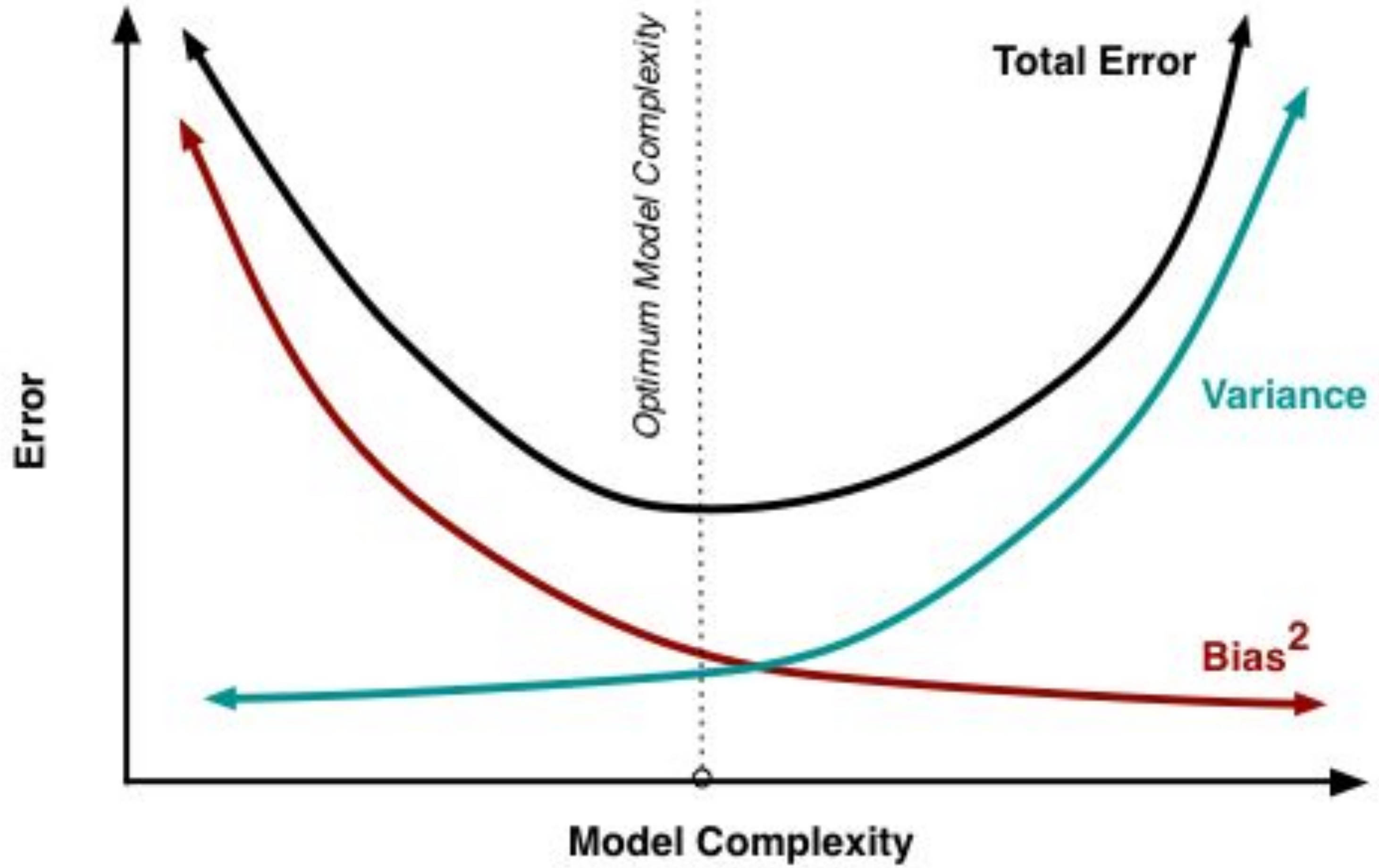
where

$$\mathrm{Var}\left[\hat{f}(x)\right] = \mathrm{E}[\hat{f}(x)^2] - \mathrm{E}[\hat{f}(x)]^2$$

$$\mathrm{Bias}\left[\hat{f}(x)\right] = \mathrm{E}\left[\hat{f}(x) - f(x)\right]$$

σ^2 Irreducible error (variance of the noise), lower bound for the model MSE

Bias-Variance against model complexity



Prediction Error =

Bias^2 + Variance

Optimum when increase in bias equals decrease in variance:

$$\frac{dBias}{dComplexity} = -\frac{dVariance}{dComplexity}$$

http://scott.fortmann-roe.com/docs/BiasVariance.html

How to detect: Bias / Variance

Plot error (e.g. MSE)

- Overfitting results in high test error, low training error
- Underfitting results in high train and test errors

Use k-fold Cross validation

- To get a good approximation of the error when there are few samples

Find the right tradeoff

- Combat underfitting with more complex model
- Combat overfitting by reducing model complexity, add samples (error asymptotically -> 0), or even better use **Regularization**

Regularization

Regularization

Why:

Avoid overfitting

(and LASSO - *Least Absolute Shrinkage and Selection Operator* - can perform auto feature selection)

How:

Increase bias by penalizing the model for many and large model parameters.

Add a multiple of an L1 (LASSO) or an L2 (Ridge) norm of the model parameters **θ** to the cost function

Regularized cost function:

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda \cdot ||\theta||_p^p$$

- *λ is the regularization parameter, basically a tuning parameter*
- $||\theta||_p^p$ *is the p:th matrix norm on the parameters*

Regularization (increase error if we have too many or too big parameters)

Non-regularized COST FUNCTION: $$J_{old}(\theta) = MSE(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$$

RIDGE REGRESSION (L2 NORM): $$J(\theta) = MSE(\theta) + \lambda \sum_{j=1}^{n} \theta_j^2$$

LASSO (L1 NORM): $$J(\theta) = MSE(\theta) + \lambda \sum_{j=1}^{n} |\theta_j|$$

Find optimal regularization term λ by tuning it and using Cross-validation:

- Divide your training data,
- Train your model for a fixed value of λ, test it on the remaining subset (unregularized cost function for testing)
- Repeat this procedure while varying λ.
  Then choose the λ that performed best on the test set.

Regularization (increase error if we use many and large model parameters)

The optimal estimates of the model parameters, β, could be denoted as shown below.

This shows us the difference between Ridge and Lasso Regression

$$\hat{\beta}^{\text{lasso}} = \operatorname{argmin} \|y - X\beta|_2^2 + \lambda \, \|\beta\|_1$$

$$\hat{\beta}^{\text{ridge}} = \operatorname{argmin} \|y - X\beta\|_2^2 + \lambda \|\beta\|_2^2$$

which is equivalent to:

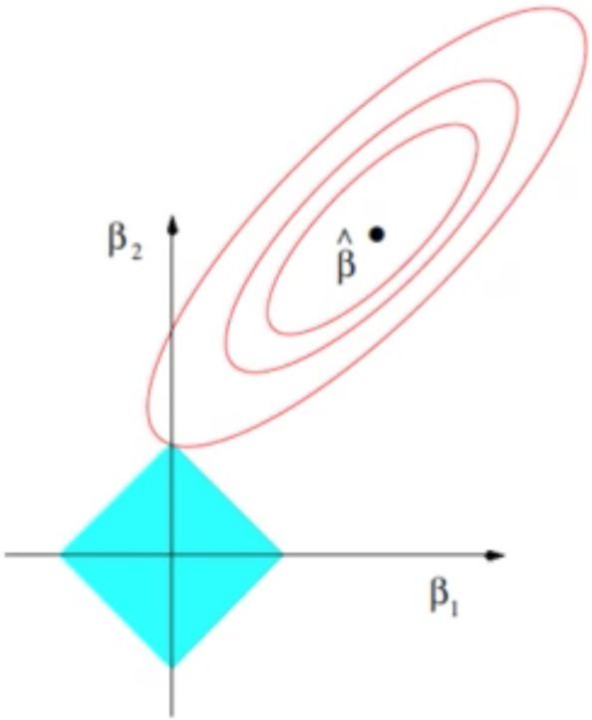
$$\hat{\beta}^{\text{lasso}} = \operatorname{argmin} \|y - X\beta|_2^2 \quad \text{subject to } \|\beta\|_1 \leq t$$

$$\hat{\beta}^{\text{ridge}} = \operatorname{argmin} \|y - X\beta\|_2^2 \quad \text{subject to } \|\beta\|_2^2 \leq t$$

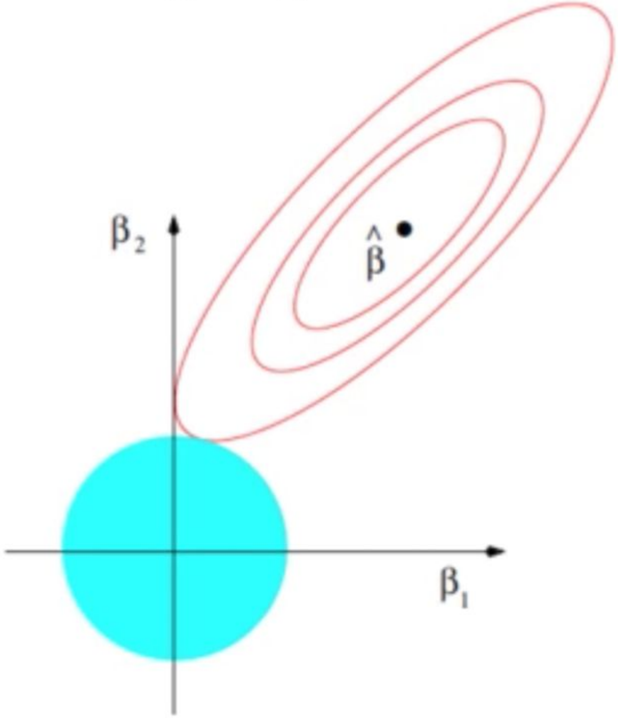
Regularization (increase error if we use many and large model parameters)

We can visualize the difference between Ridge and Lasso Regression for two parameters. Note, there is a trade-off between the Least Square error and the size of the parameters (which are constrained, to the blue areas).

Lasso Regression



Ridge Regression



$$t \propto \frac{1}{\lambda}$$

$$\hat{\beta}^{\text{lasso}} = \text{argmin} \ \|y - X\beta\|_2^2 \quad \text{subject to } \|\beta\|_1 \leq t$$

$$\hat{\beta}^{\text{ridge}} = \text{argmin} \ \|y - X\beta\|_2^2 \quad \text{subject to } \|\beta\|_2^2 \leq t$$

Example Code: Regularization

End

References

- The material presented in this lecture references lecture material draws on the materials the following courses:
- Derek Kane's Data Science Tutorials: https://www.youtube.com/channel/UC33qFpcu7eHFtpZ6dp3FFXw
- Stanford – CS229 (Machine Learning) & Andrew Ng's Machine Learning at Coursera: http://cs229.stanford.edu/ & https://www.coursera.org/learn/machine-learning
- Professor Alexander Ihler, UC Davis: youtube.com/watch?v=sO4ZirJh9ds