Supervised Machine Learning
A Gentle Introduction

Dr. Jia-Jie (JJ) Zhu
Max Planck Institute for Intelligent Systems

Guest lecture with Prof. Moritz Diehl
January 10th, 2020
University of Freiburg

What exactly is ML/AI?

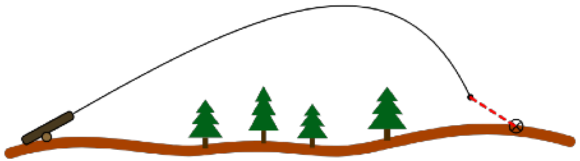
Field of study that gives computers the ability to learn without being explicitly programmed.

Supervised Learning

Regression

Classification

The optimal control problem



$$\underset{u_t, t=0,\dots,N-1}{\text{minimize}} \quad \sum_{t=1}^{N-1} g(x_t, u_t) + J(x_N)$$

$$\begin{aligned} \text{subject to} \quad & x_{t+1} = f(x_t, u_t), \ \forall t \\ & h_t(x_t, u_t) \leq 0, \ \forall t \\ & x_0 = x_p \end{aligned}$$

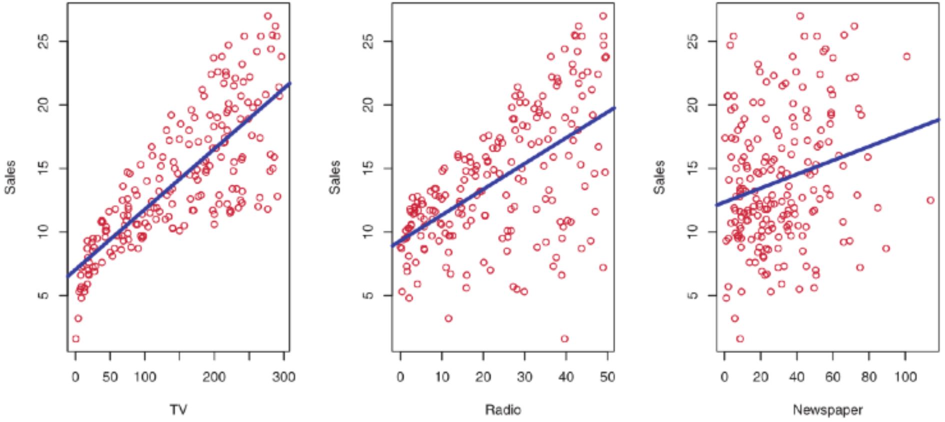
- The system model $f : (x, u) \mapsto x^+$ makes predictions (next state) based on given data (current state and control).
- Such predictive modeling is the study of (supervised) machine learning.

The system model may be uncertain/unknown

Figure adopted from M. Kelly 2017

If you are a ML consultant hired by a client…

The **Advertising** data set consists of the **sales** of that product in 200 different markets, along with advertising budgets for the product in each of those markets for three different media: **TV** , **radio** , and **newspaper**. Your job is to provide advice on how to improve sales of a particular product.



Data in machine learning

In machine learning, we usually work with data sets denoted by $\{x_i, y_i\}_{i=1,2,\dots,n}$

$$x_i = \begin{pmatrix} x_{i1} \\ x_{i2} \\ \vdots \\ x_{ip} \end{pmatrix}$$

x_i	y_i
input	output
features	dependent variable
independent variable	response
predictors	

In our example, x_{i1} might be the TV budget, x_{i2} the radio budget, and x_{i3} the newspaper budget. y_i is the sale of the product.

Data in machine learning

In machine learning, we usually work with data sets denoted by $\{x_i, y_i\}_{i=1,2,\ldots,n}$

$$x_i = \begin{pmatrix} x_{i1} \\ x_{i2} \\ \vdots \\ x_{ip} \end{pmatrix}$$

input
features
independent variable
predictors

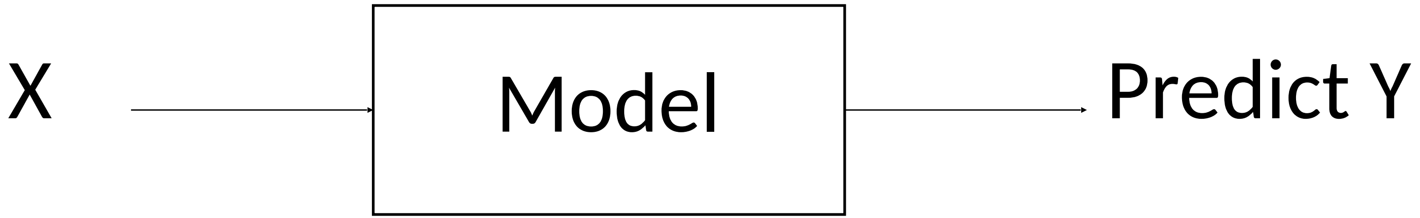
y_i

output
dependent variable
response

In machine learning, we are interested in figuring out the relationship between in order to make predictions. This relationship can be denoted as $Y = f(X) + \epsilon$

Regression

Regression

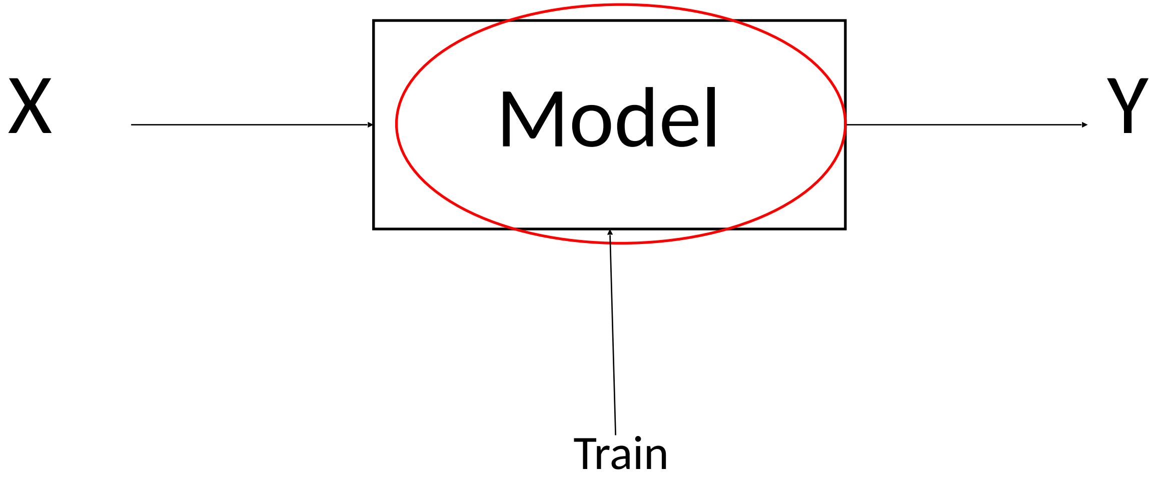


Budget
Age
Education level
…

Sales
Income
Survival time
…

How does it work?

Given training data: $\{x_i, y_i\}_{i=1,2,\ldots,n}$, we train (fit, teach) the model.



How does it work?

Then we pass the new (unseen) test data x_* to the model to predict the unknown quantity of interest y_*.

x_* $\longrightarrow$ $\boxed{\text{Model}}$ $\longrightarrow$ predict y_*

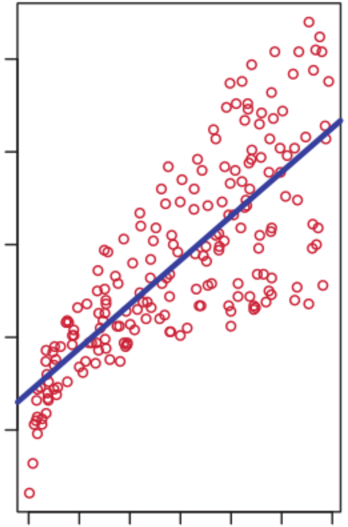
$\xrightarrow{\hspace{3em}}$

Test

Linear Regression

The model: $f(x) = \theta^T x$

How do we learn from the data?



Linear Regression

The model: $f(x) = \theta^T x$

How do we learn from the data?

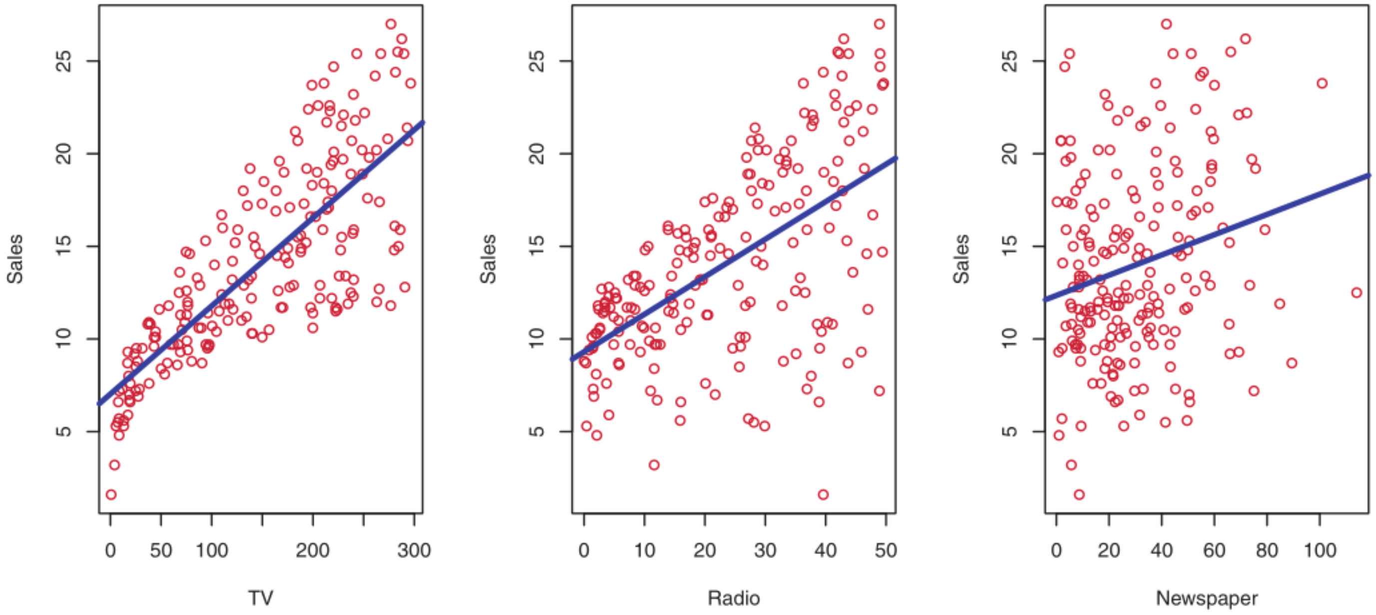
We minimize the objective function:

This is called least square regression

$$\min_{\theta} J(\theta) = \sum_{i} (\theta^T x^{(i)} - y^{(i)})^2,$$

where $\{x^{(i)}, y^{(i)}\}_{i=1,2,\dots}$ is the training dataset. This process is called optimization.

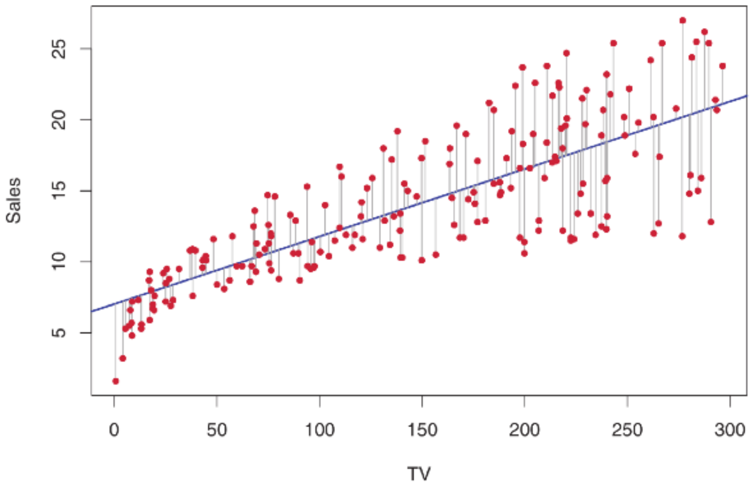
Case study: Advertising data set



Case study: Advertising data set

$\text{sales} \approx \theta \times \text{TV} + \theta_0$

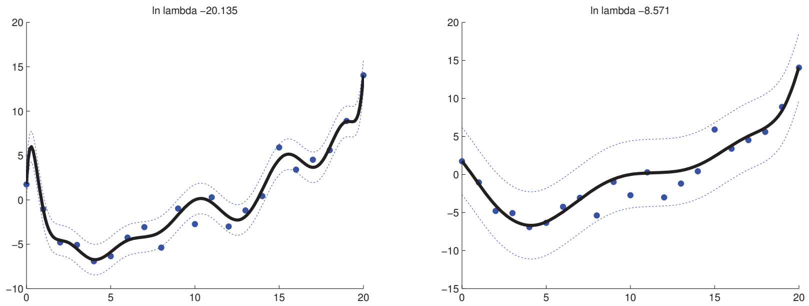
Given data, we want to solve $\min_\theta J(\theta) = \sum_i (\theta^T x^{(i)} - y^{(i)})^2$



Solving this optimization problem, we have found that $Y = 7.03 + 0.0475X$.

Generalized linear model

- Consider the the generalized linear model: $f(x) = \theta^\top \phi(x)$.
  - The mapping $\phi : x \mapsto \phi(x)$ is called a "feature map".
- This allows us to model non-linear relationship. (you have learned non-linear least square)
-



How to do it on a computer? (sklearn)

Training (a.k.a. learning, fitting):

```
model = linear_model.LinearRegression()
model.fit(data: X, Y)
```

Testing (a.k.a. predicting, evaluating):

```
Y = model.predict(new data: X)
```

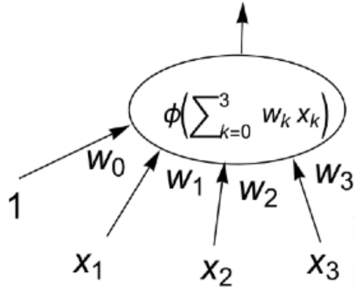
More about this during the hands-on session.

Deep neural nets

The hottest topic in machine learning and beyond

Neural networks are computational models motivated by our understanding of the brain.

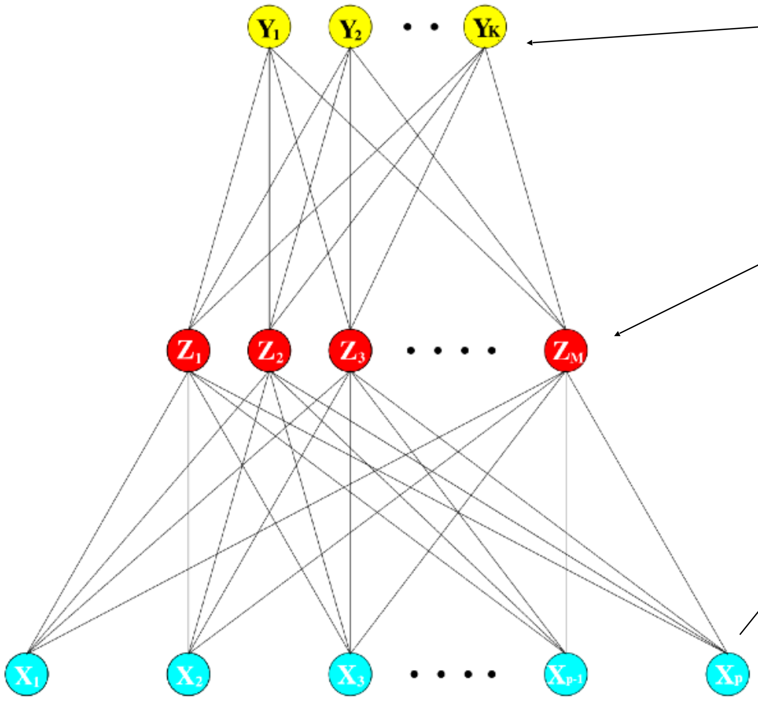
A single neuron



$$\hat{y}_i = \sigma\left(\sum_{j=1}^{d} w_{ij} x_j\right)$$

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$ sigmoid

A network of neurons



$$\hat{y}_i = \sigma\left(\sum_{j=1}^{d} w_{ij} x_j\right)$$

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$ sigmoid

How to learn/train NN?

Recall that we minimize the objective function in linear regression

$$\min_{\theta} J(\theta) = \sum_i (\theta^T x^{(i)} - y^{(i)})^2$$

When training DNN, we solve the following optimization problem.

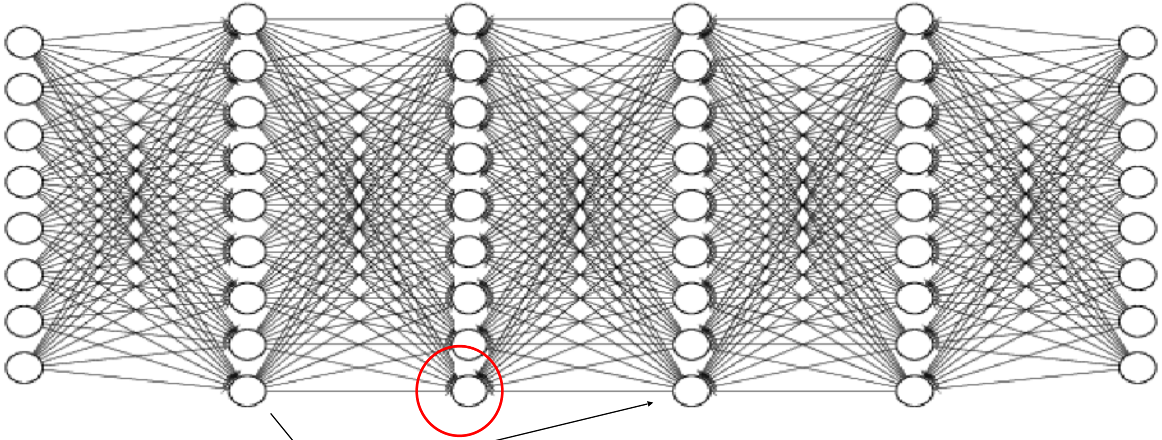
$$\min_{\theta} \sum_{i=1}^{N} \left(f_\theta(x^{(i)}) - y^{(i)}\right)^2,$$

Where f_θ is the NN with weights θ.

Recall, you have learned the derivation of this in MLE/MAP.

When applying stochastic gradient descent to the above problem, the training is sometime called back-propagation (due to the chain rule).

Deep Neural Network – Stack them up!

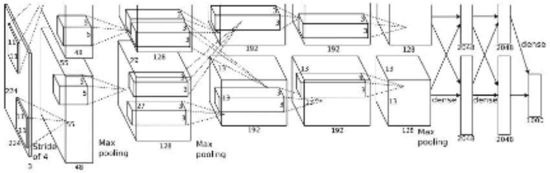


$$\hat{y}_i = \sigma\left(\sum_{j=1}^{d} w_{ij} x_j\right)$$

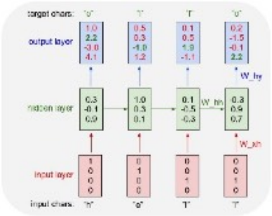
This is called multi-layer perceptron (MLP).

Other structures

Convolutional neural net (CNN)



Recurrent neural net (RNN)



A closer look: what is learning?

Model assessment and selection

Empirical risk minimization (ERM)

The ML task is to minimize the following **empirical risk**.

$$R_{emp} = \frac{1}{N} \sum_{i=1}^{N} L\left(y_i, \delta\left(\mathbf{x}_i\right)\right),$$

where $\delta(x_i)$ is the predictive model. It can be linear or arbitrary form such as NN.

However, exact minimization of ER will result in overfitting. (nature's distribution is typically not degenerative)

Solution: regularization

Instead of the empirical loss, we minimize the regularized version:

$$\frac{1}{N}\sum_{i=1}^{N} L\left(y_i, \delta\left(\mathbf{x}_i\right)\right) + \lambda R(\delta)$$

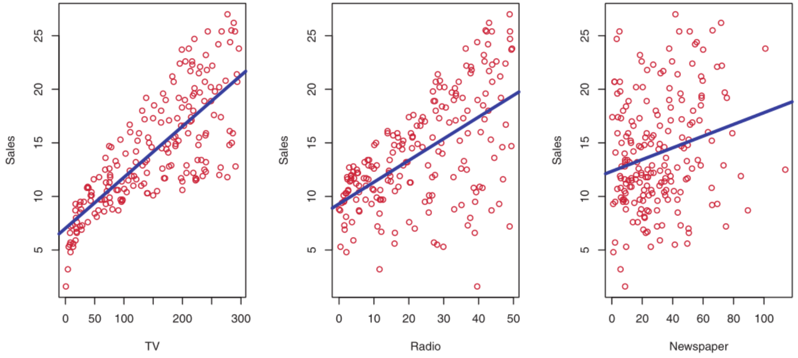
- This works with simple linear models, as well as more complicated models such as deep neural nets.
- This forces the function map to have some regularity commonly found in natural domains, such as smoothness. 
- “learning = extraction of the regularity from the data’’
(B. Schoelkopf or someone else)

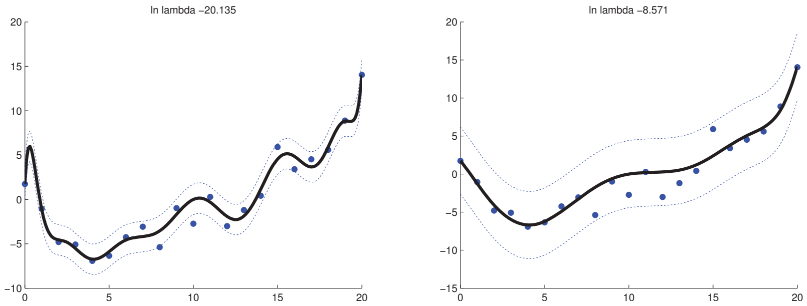
Example: ridge regression (RR)

- Given dataset,

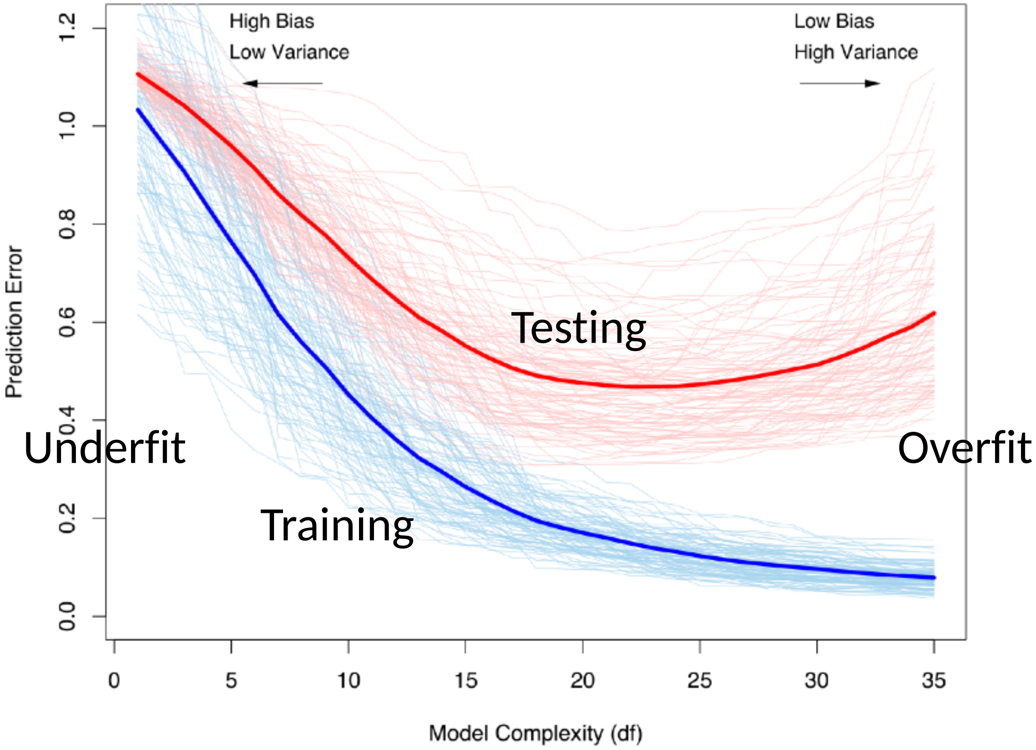
$$\sum_i \left(\theta^\top \phi(x^{(i)}) - y^{(i)}\right)^2 + \lambda \|\theta\|_2^2.$$

- The **regularization** term controls functions to not ‘wiggle’ too much.





The regularization reflects in the so-called variance and bias trade-off

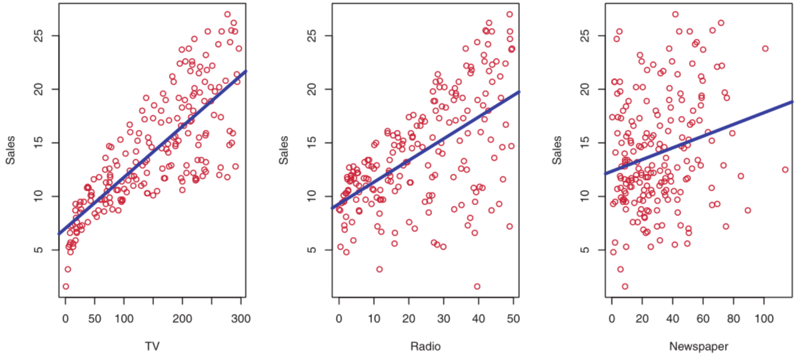


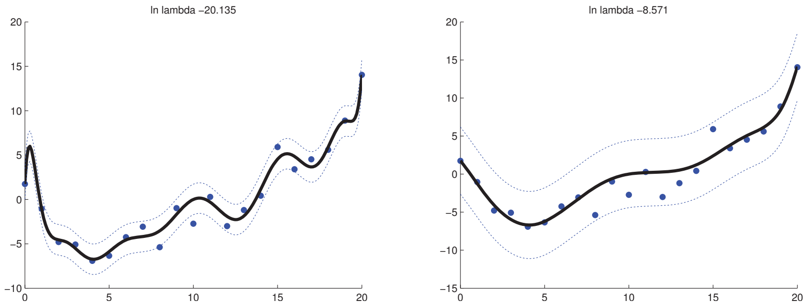
Example: ridge regression (RR)

- Given dataset,

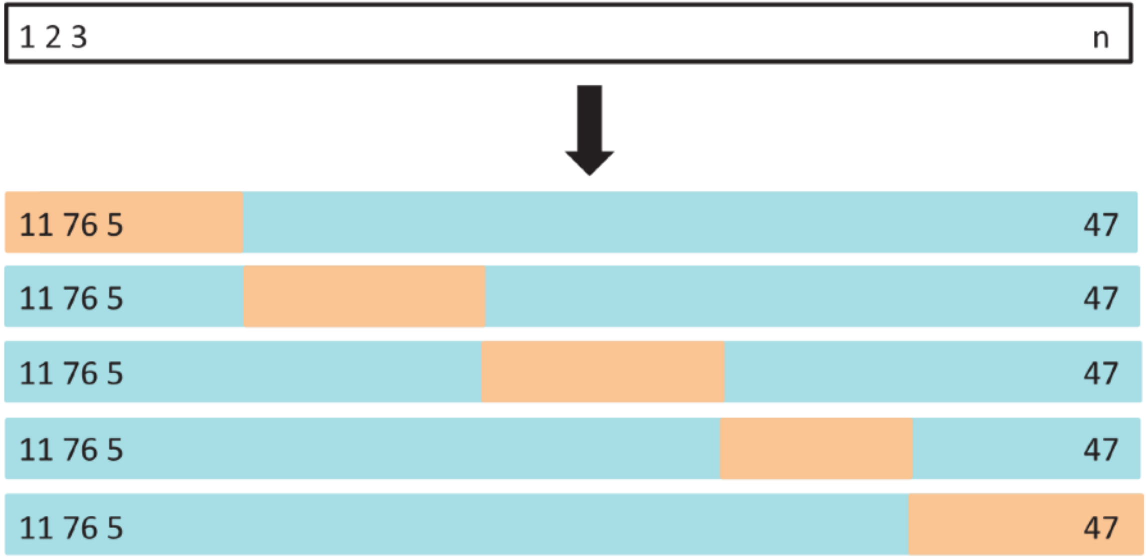
$$\sum_i \left(\theta^\top \phi(x^{(i)}) - y^{(i)}\right)^2 + \lambda \|\theta\|_2^2.$$

- **How do we choose λ?**





K-fold Cross-Validation (CV)



Choose the model (hyper-parameter) with the smallest CV error.

$$\mathrm{CV}_{(k)} = \frac{1}{k}\sum_{i=1}^{k} \mathrm{MSE}_i$$

Use CV to choose hyperparameter for ridge regression

Recall that we minimize the objective function in linear regression

$$\min_\theta J(\theta) = \sum_i (\theta^T x^{(i)} - y^{(i)})^2$$

We add a term to this objective function to arrive at the ridge regression model

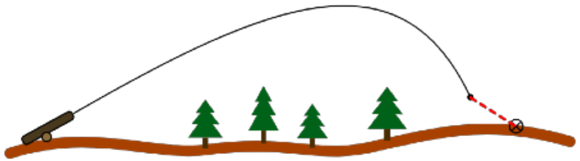
$$\min_\theta J(\theta) = \sum_i (\theta^T x^{(i)} - y^{(i)})^2 + \lambda \|\theta\|^2.$$

Quiz: what happens to RR when $\lambda = 0$?

$\lambda = \infty$

λ is a hyperparameter which we may choose by CV.

Big picture: ML as function approximation



$$\underset{u_t, t=0,\dots,N-1}{\text{minimize}} \quad \sum_{t=1}^{N-1} g(x_t, u_t) + J(x_N)$$

$$\text{subject to} \quad x_{t+1} = f(x_t, u_t), \ \forall t$$

$$h_t(x_t, u_t) \leq 0, \ \forall t$$

$$x_0 = x_p$$

- What we have talked about so far is to build a model using tools such as GLM and DNN to approximate the function

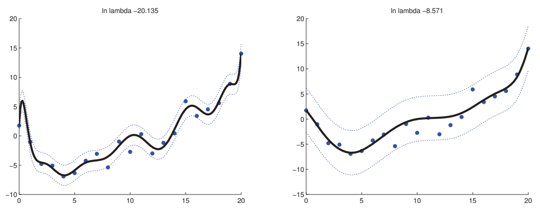
$$f : (x, u) \mapsto x^+$$

- Mathematically, this is called function approximation.

Figure adopted from M. Kelly 2017

Big picture: ML as function approximation

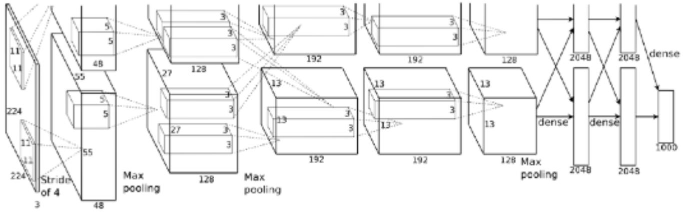
(Generalized) linear model



Data efficient
Interpretable — white/gray box

Need to design features
Hard to cope with large datasets
Can only model limited functions
But often enough

Neural nets



Extremely scalable
Can work with huge datasets

Data hungry
Optimization is questionable

Other

Gaussian process, polynomial chaos, mixture density, kernel regression,
…

In general, we can classify methods into parametric and non-parametric ones.

Summary

In this lecture, we have learned:

- The training—testing paradigm of supervised machine learning
- Model classes such as GLM (ridge reg.) and NN
- How to regularize and select models
- The framework of empirical risk minimization
    - The essence of learning is *seeking regularity in nature*

References & Recommendations for further reading

- Online courses
  - A Ng's ML/DL courses on Coursera
  - G Hinton's course on Coursera
  - N de Freitas's machine learning lectures
  - Stanford CS230/1n (deep learning, CNN stuff)
- Textbooks:
  - James et al., An Introduction to Statistical Learning with Applications in R
  - *Hastie et al., The Elements of Statistical Learning
  - Murphy, Machine Learning: A Probabilistic Perspective
  - *Bishop, Pattern Recognition and Machine Learning
  - *Schoelkopf et al., Learning with Kernels
  - Sutton et. al., Reinforcement Learning, an Introduction
  - Goodfellow et. al, Deep Learning

*technical