JOHNS HOPKINS
CAREY BUSINESS SCHOOL

Lecture 3

BU.330.775 Machine Learning

Minghong Xu, PhD.
Associate Professor

Reflections

- Representation and features
- Training, testing, validation and cross-validation
- Ordinal encoding vs one-hot encoding
  - If you are unsure, one-hot encoding is safer but more expensive
- Outliers (are there for you to explain!)
- Imputation with zero, mean, or median
- Normalization vs standardization

- *There is no "one rule fits all"*

Today's Agenda

- Supervised machine learning models, part I
  - Training of regression models: gradient descent
- Regularization
  - Ridge regression
  - Lasso regression
  - Early stopping
- Model evaluation

- Hands-on learning on MNIST dataset

Classification and Regression (Task)

- Classification task: predict a **class** label
  - Binary: two classes
  - Multiclass
- Regression task: predict a continuous number

- Note that we also have regression-type of models

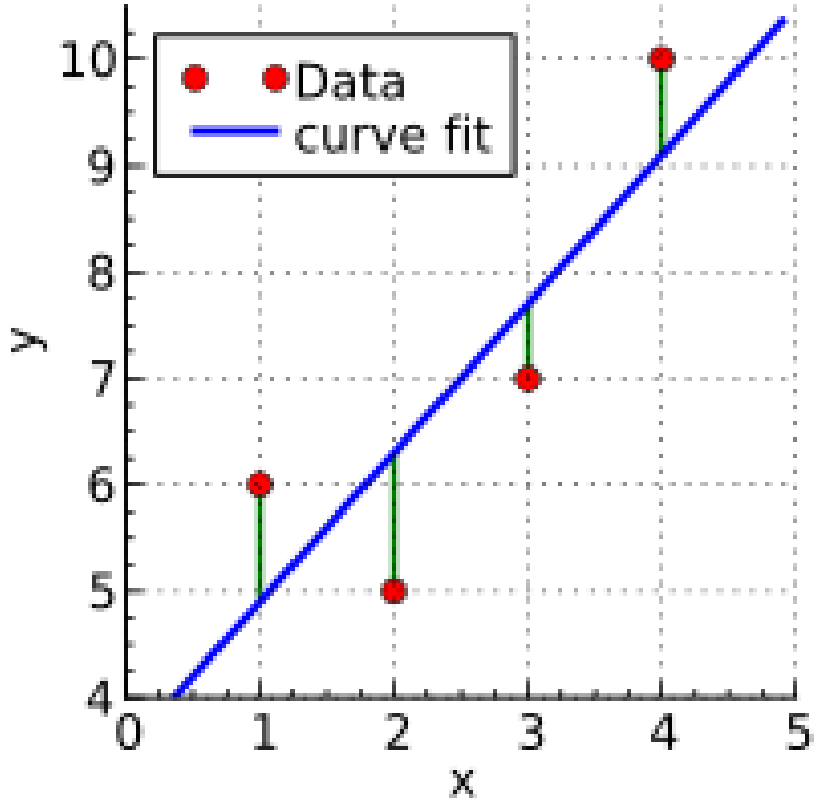
Major Supervised Algorithms

- **Linear regression: linear relationship**
- **Logistic regression: modeling the probability** — Classification algorithm.
- K-nearest neighbors: similar data points tend to have similar labels or values
- Decision trees, random forests, and boosting: tree structures
  - Decision trees and random forests will be discussed in Data Science and Business Intelligence course in Spring II
- Support Vector Machines: max-margin models
  - Topic of Data Science and Business Intelligence course in Spring II
- Neural networks: deep learning
  - Topic of AI Essentials for Business course in Spring I

Regressions and Model Training

Linear Regression

- Or ordinary least squares (OLS), simplest and most classic linear method for regression
- Find a linear relationship that minimize the mean squared error between predictions and targets



Parameters and Hyperparameters

- We will use y=Wx+b to denote
  - To be consistent with deep learning
  - **W**: weights
  - **b**: bias
- W and b are called **parameters**
  - Variables that are learnt by the model during the training process
  - Determined by machine via training
- **Hyperparameters**: fixed before the training process
  - Examples: K in K-nearest-neighbor
  - Determined by human before training

parameter vs hyperparameter

Notes

Training Process

- Use the training data set to find the best values of parameters in the model
- Use linear regression as example
- *What is the goal?*
- Minimize the mean squared error between predictions and targets
- Thus, also called **model optimization**
  - To optimize the model's parameters

Gradient Descent

- A common optimization algorithm
  - You will see it again in deep learning
- Use again, linear regression as example
- *What is the shape of MSE?*
- MSE here is referred to as **loss function** or **cost/objective function**
  - Measure the loss between predictions and targets
  - But MSE is just one type of loss functions

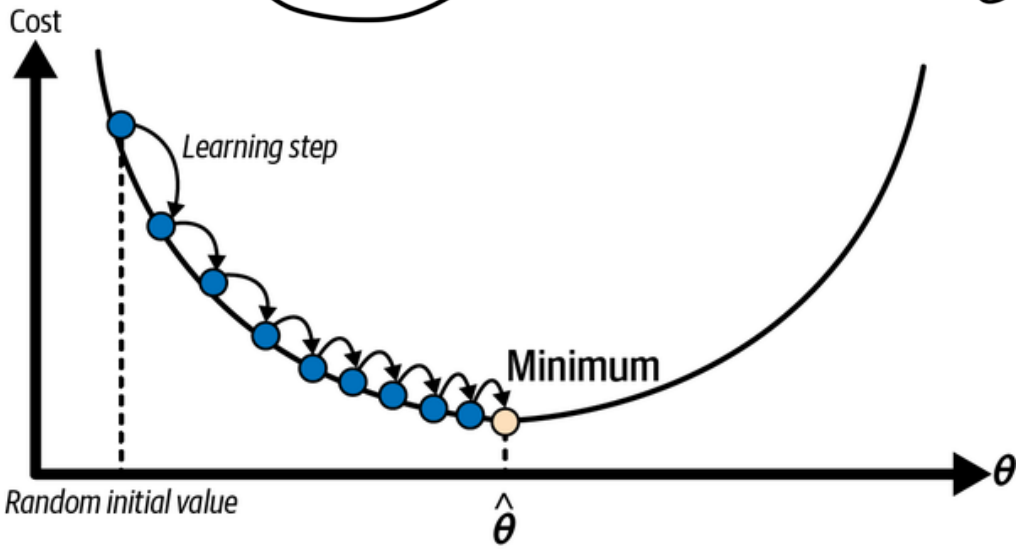
error due to single data point

average of loss function over complete dataset.

- General idea: update parameters iteratively to minimize loss

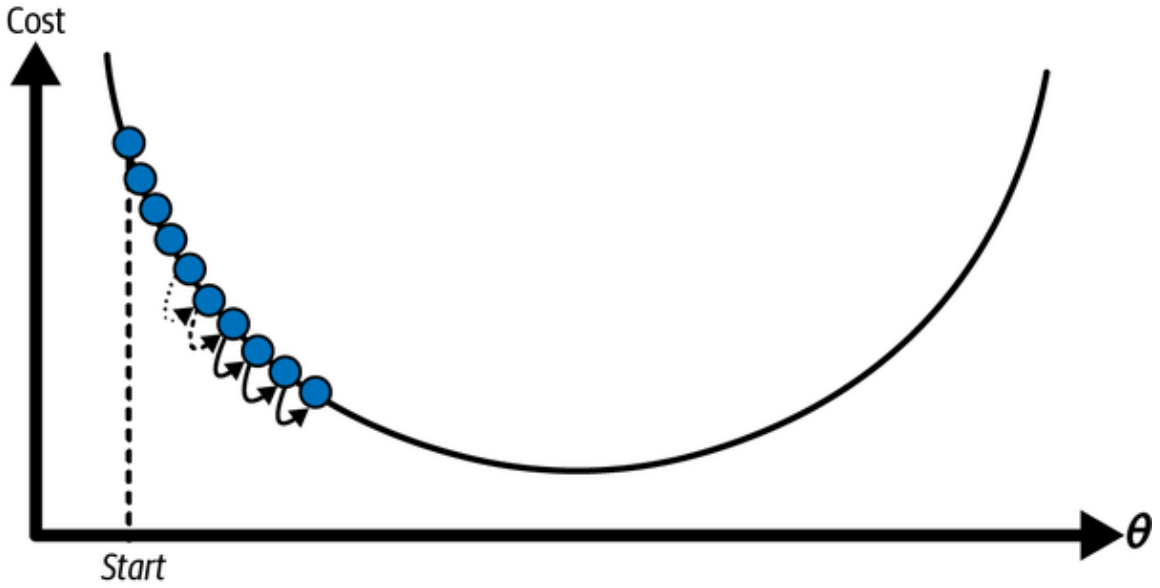
Gradient Descent: How

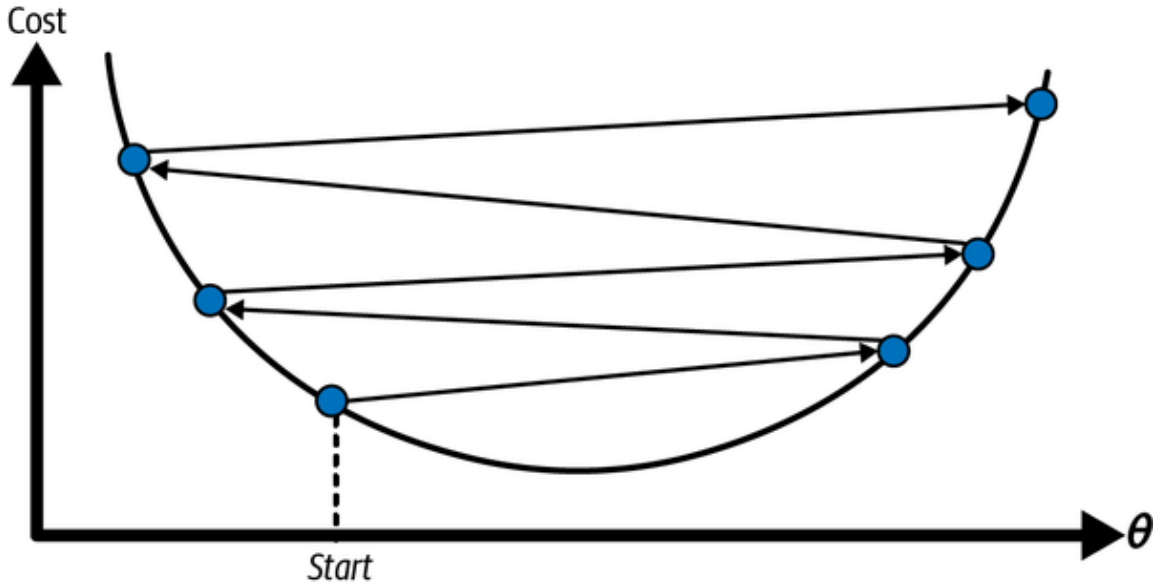
- First order approach
- Get gradient of L with respect to parameters W, b
- Update
  - $W_{k+1} \leftarrow W_k - \alpha \nabla_W L$
  - $b_{k+1} \leftarrow b_k - \beta \nabla_b L$
  - α, β are step sizes or learning rates
- Repeat Until gradient is approximately small (when update is little)



Learning Rate

- The size of steps, a hyperparameter
- Too small vs too big





Adaptive Learning Rates

- Popular and simple idea: reduce the learning rate by some factor every few training epochs
  - At the beginning, we are far from destination, use larger learning rate
  - After several epochs, we are close to the destination, reduce the learning rate
- AdaGrad (Duchi, 2011): adaptive gradient algorithm
- RMSprop (Hinton, 2012): root mean square propagation
- AdaDelta (Zeiler, 2012): adaptive delta
- AdaSecant (Gulcehre, 2014): adaptive secant
- Adam (Kingma, 2015): adaptive moment estimation

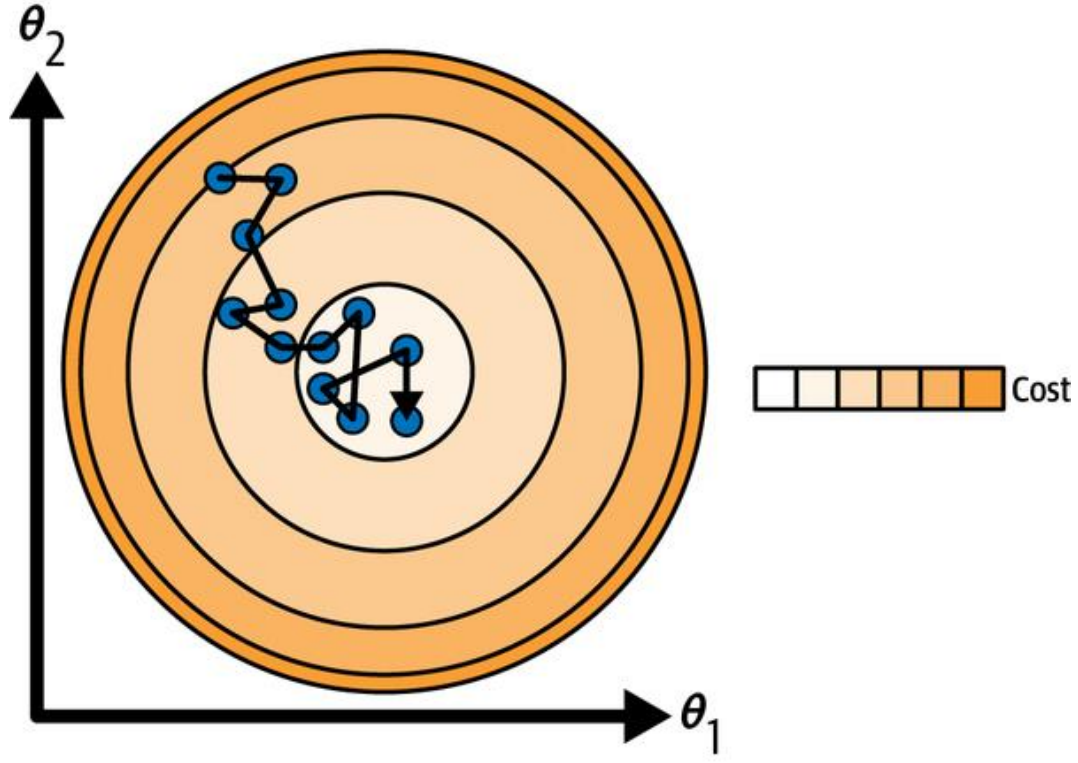
Batch Gradient Descent

- To implement gradient descent, we need to compute the gradient of the loss function with regard to each parameter, and then update
- *When to update? Or how often?*

- If you compute over the whole training set
- It is called Batch Gradient Descent
  - Use the whole batch of training set to update once
- *Drawback?*
- One step can be very slow if you have a very large training set

Stochastic Gradient Descent

- At the opposite extreme is Stochastic Gradient Descent
- Pick one data point in training set at every step and update once
  - Every step is fast!
- Result in a more "bouncing" behavior
  - Every time, one instance is used to evaluate the loss function
  - Thus, "stochastic"



Mini-Batch Gradient Descent

- Sit in the middle
- Divide the whole training set into mini-batches
- Use one mini-batch to update the parameter at every step
- Usually preferred over BGD and SGD

Randomly initialize network parameters

- Pick the 1st mini-batch, update parameters once
- Pick the 2nd mini-batch, update parameters once
- …
- Until all mini-batches are picked

Training Epoch

- One training iteration over the training set is called an **epoch**
- # of epochs: # of times that the entire training set update the model parameters

Randomly initialize network parameters

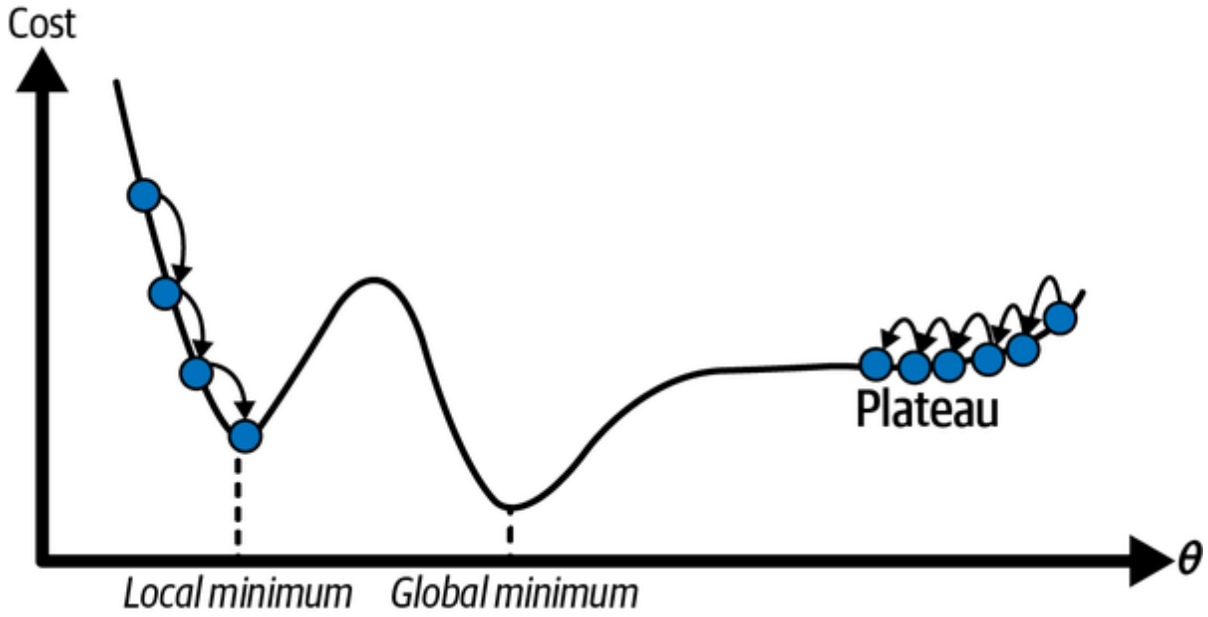
- Pick the 1st mini-batch, update parameters once
- Pick the 2nd mini-batch, update parameters once
- …
- Until all mini-batches are picked

} one epoch

Repeat the above process

More about Gradient Descent

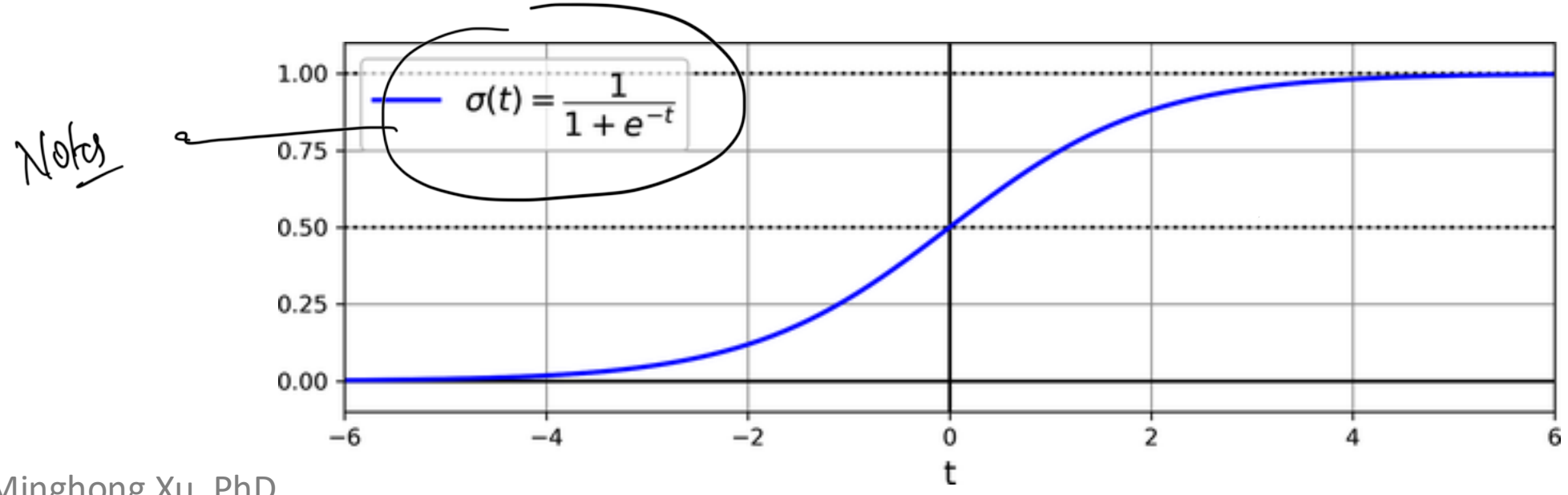
- In AI Essentials course
- When loss function is non-convex in neural network models
- We need a technique called **Back-propagation**, key to neural network training



- *Why machine learning models need sufficient training?*
- *Why machine learning models need a lot of data?*

Logistic Regression

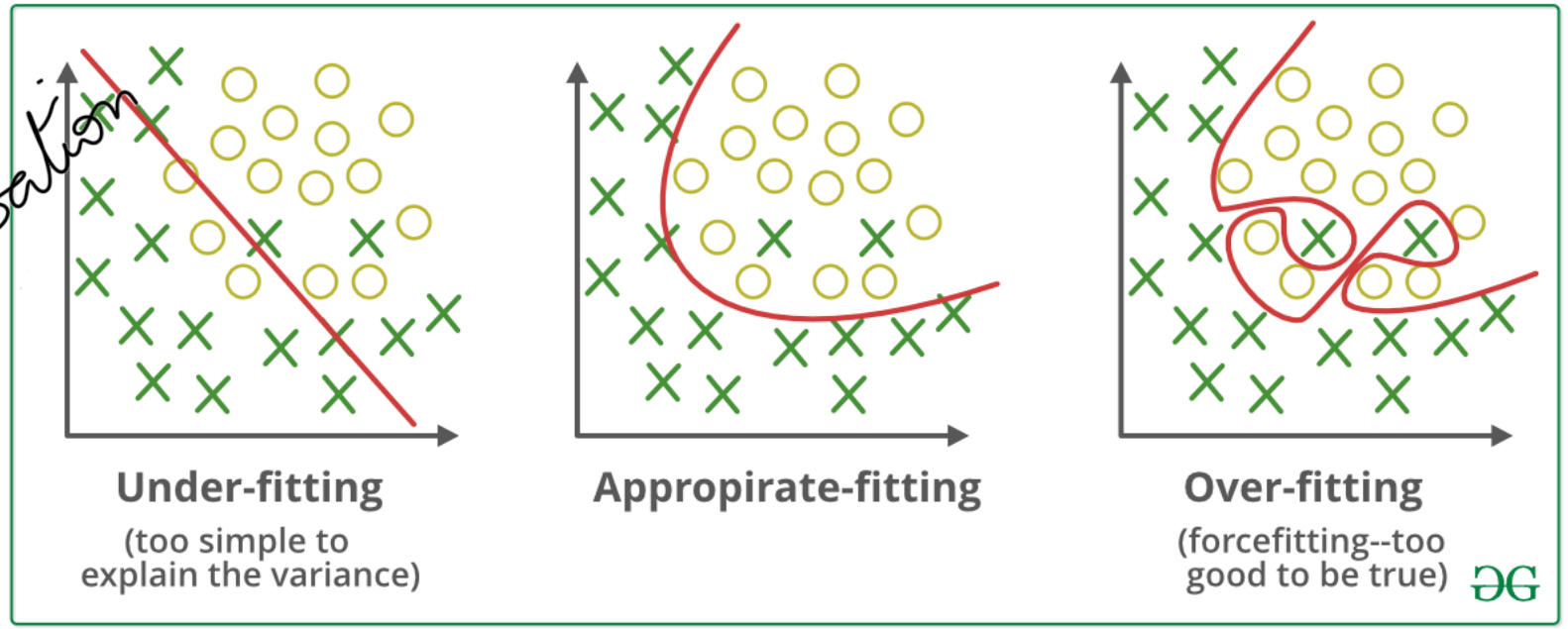
- Also called logit regression, a **classification model** despite the name
  - Estimate the probability that an instance belongs to a particular class
- Work as linear regression (LR)
- Convert the result of LR to logistic of the result, and set a threshold



Regularization

Overfitting vs. underfitting

are handled by regularization

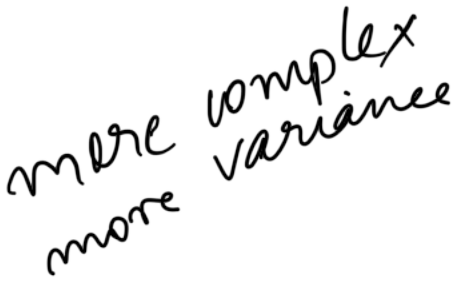


- *Why overfitting is a problem?*

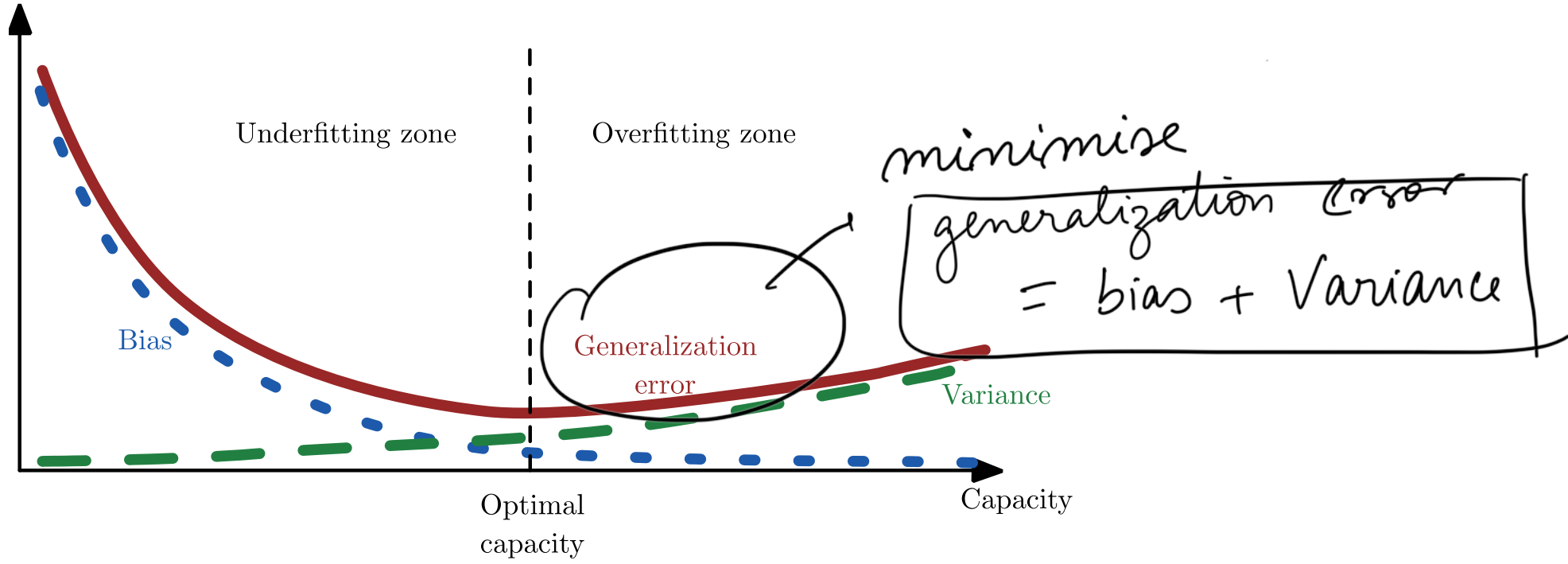
Regularization

- General idea: control overfitting on training data

- *How?*
- Bias: gap between prediction and true label
  - Reduced when model becomes more capable(complex)
- Variance: how model varies as a function of the data
  - Increase when model becomes more complicated(capable)



Bias/Variance Trade-Off



- Regularization: modifications we make to a learning algorithm that is intended to reduce its generalization error

Note

Norm Penalties

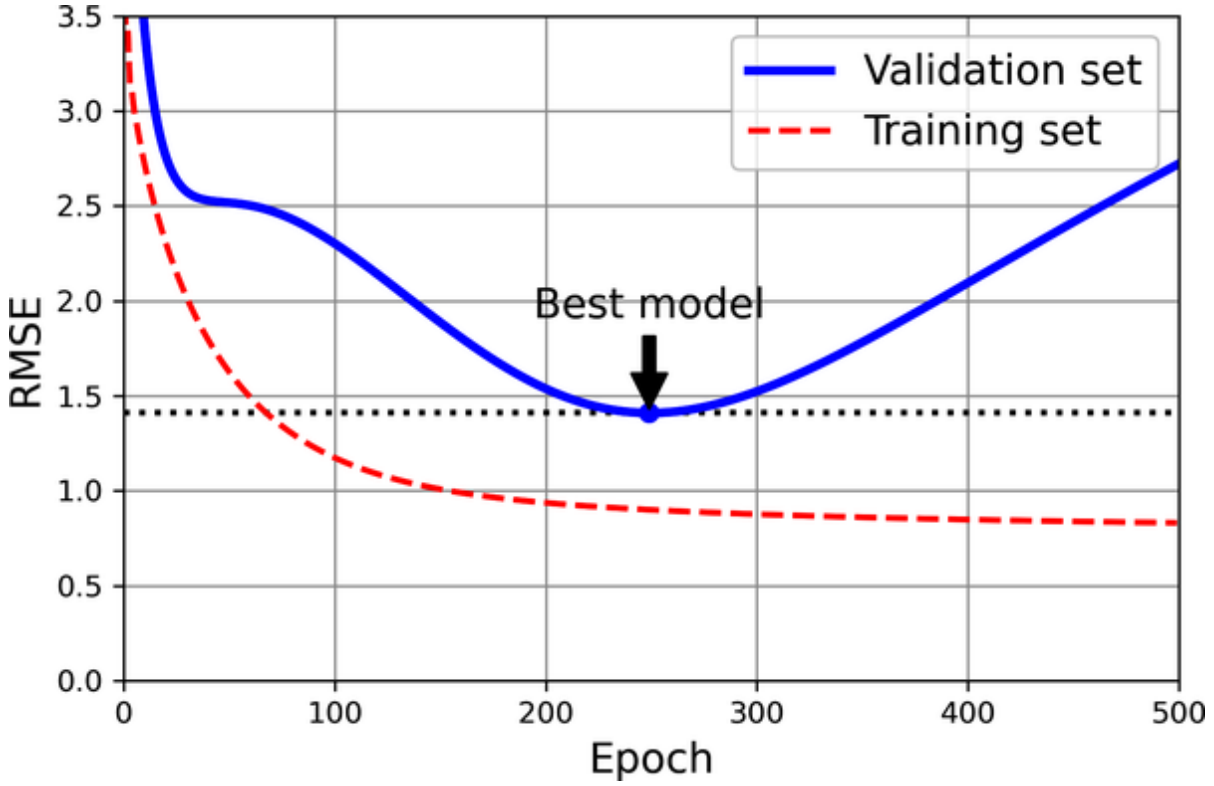
- Model capacity is usually measured in # of parameters
- L2 regularization: $\sum_i w_i^2$
  - Weight decay, Tikhonov regularization
  - Ridge regression
- L1 regularization: $\sum_i |w_i|$
  - Lasso regression (least absolute shrinkage and selection operator regression)
  - The sum of absolute values of the individual parameters
- Can be combined in elastic net regression

L1 vs L2

- Compared to L2, L1 results in a solution that is more sparse
- In general, L1 is preferred over L2 when
  - Want a sparse model
  - Need feature selection
  - Have high-dimensional data
  - Require interpretability
- L2 is more computational efficient, no need to handle absolute values

Early Stopping

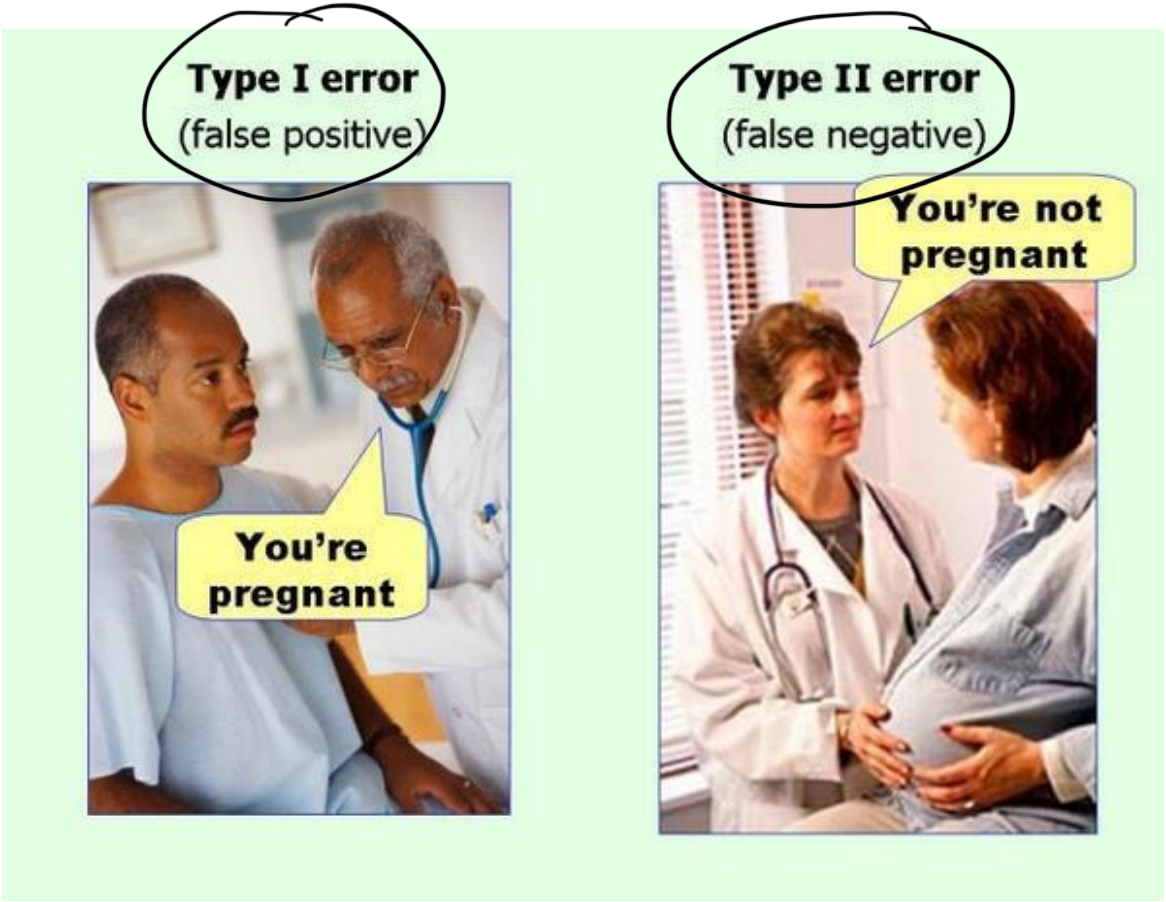
- Stop training as soon as the validation error reaches a minimum
- "Beautiful free lunch"



Model Evaluation

Two Types of Errors

- Null hypotheses (H0):
  - Life as usual
  - No treatment needed
- Alternative hypothesis (H1):
  - New conditions
  - Needs treatment
- *Why distinguish the two types?*

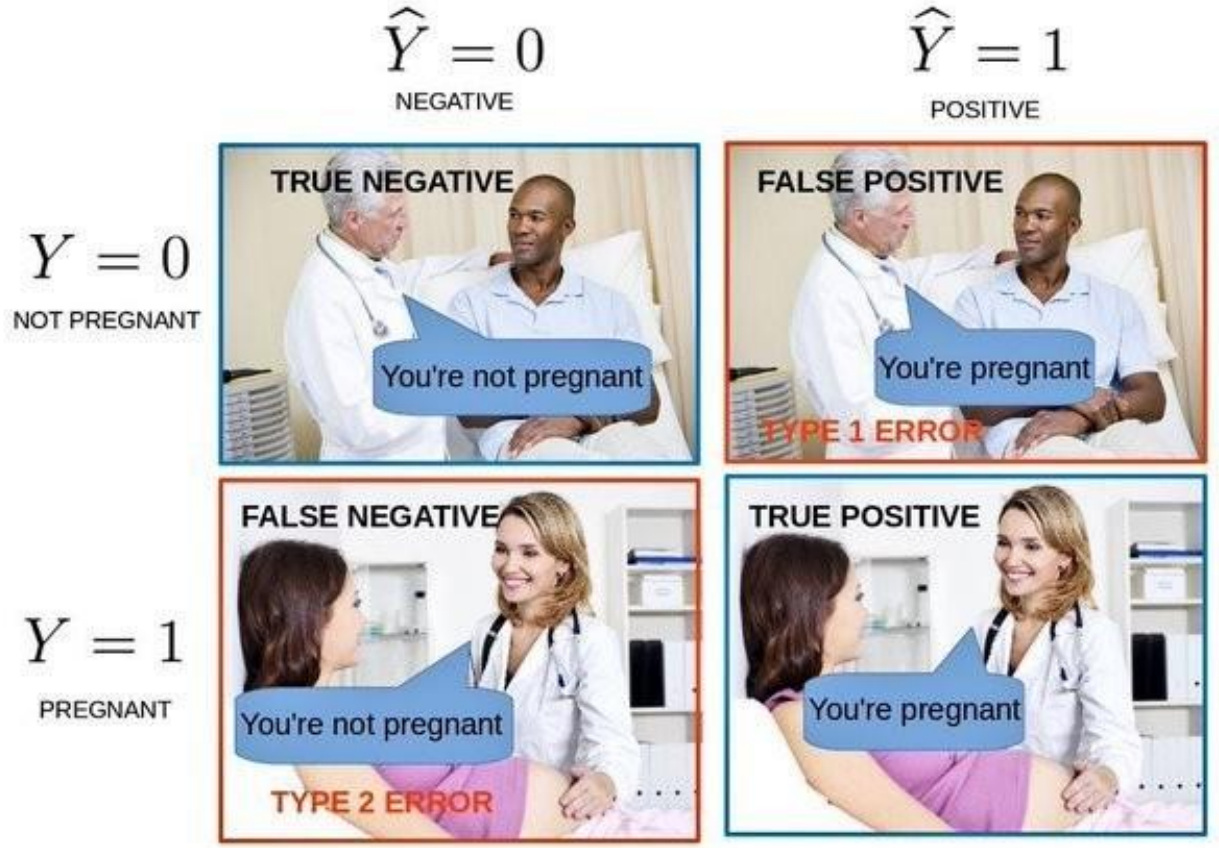


Discussion

- *In which situations is a*
  - False positive
  - False negative

  more costly?

Confusion Matrix



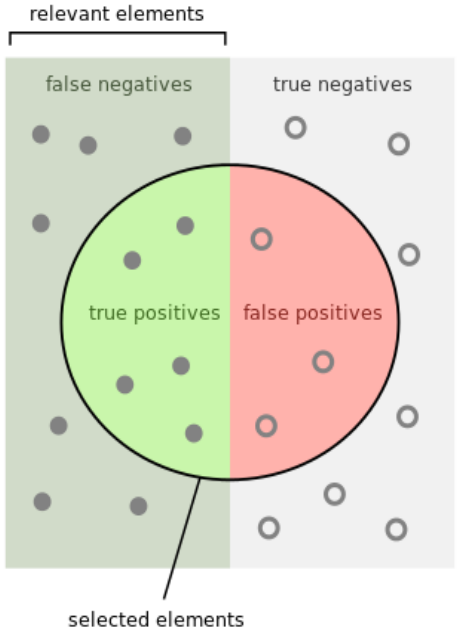
Credit: TowardsDataScience

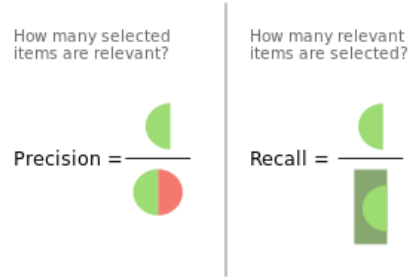
Performance Measures

- **Accuracy**: $\frac{true\ positives + true\ negatives}{all\ cases}$
- **Precision**: $\frac{true\ positives}{predicted\ positives}$
- **Recall**: $\frac{true\ positives}{all\ positives}$, also called **Sensitivity**
- **F1-score**: $2 \times \frac{precision \times recall}{precision + recall}$
- **Specificity**: $\frac{true\ negatives}{all\ negatives}$

Note

all actual positive

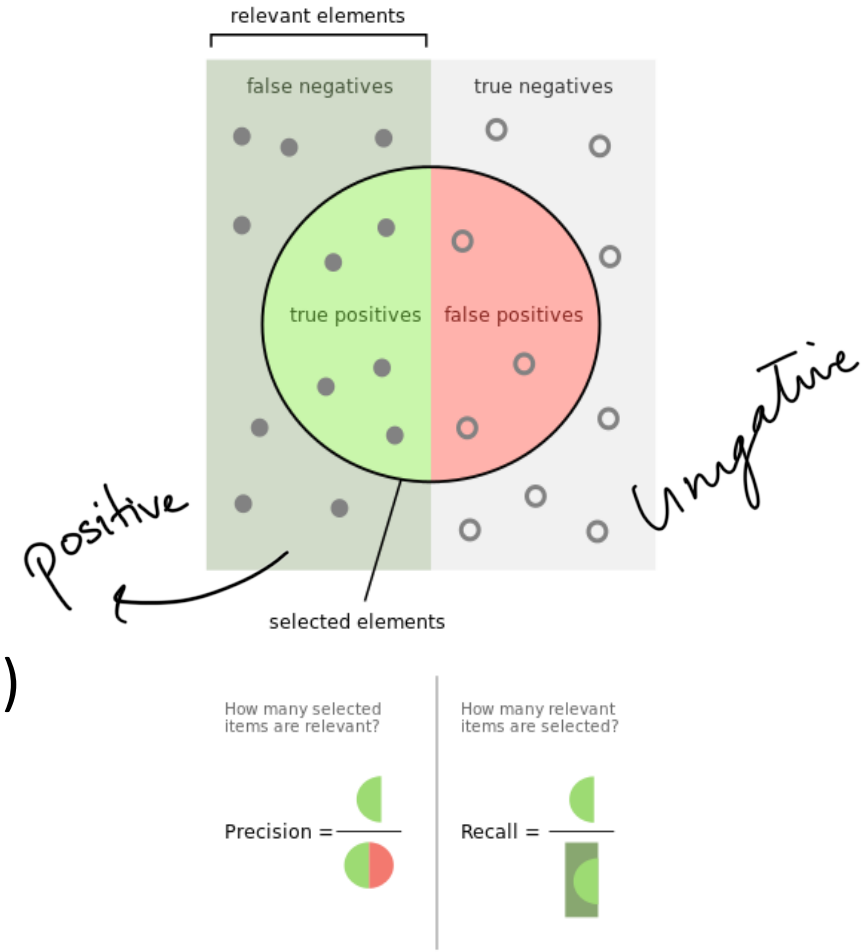




Credit: Wikipedia

Class Practice

- Ground Truth
  - 12 positive (the left panel)
  - 10 negative (the right panel)
- Model Prediction
  - 8 predicted positive (those in the circle)
    - 5 correct (true positive)
    - 3 wrong (false positive)
  - 14 predicted negative (those outside the circle)
    - 7 correct (true negative)
    - 7 wrong (false negative)
- *Accuracy? Precision? Recall? F1-score?*



Precision-Recall Trade-off

- Trade-off between precision and recall: improving one often leads to a decrease in the other, creating a trade-off
  - Improve precision: reduce false positive, raise threshold for positive prediction
  - Improve recall: reduce false negative, lower threshold for positive prediction

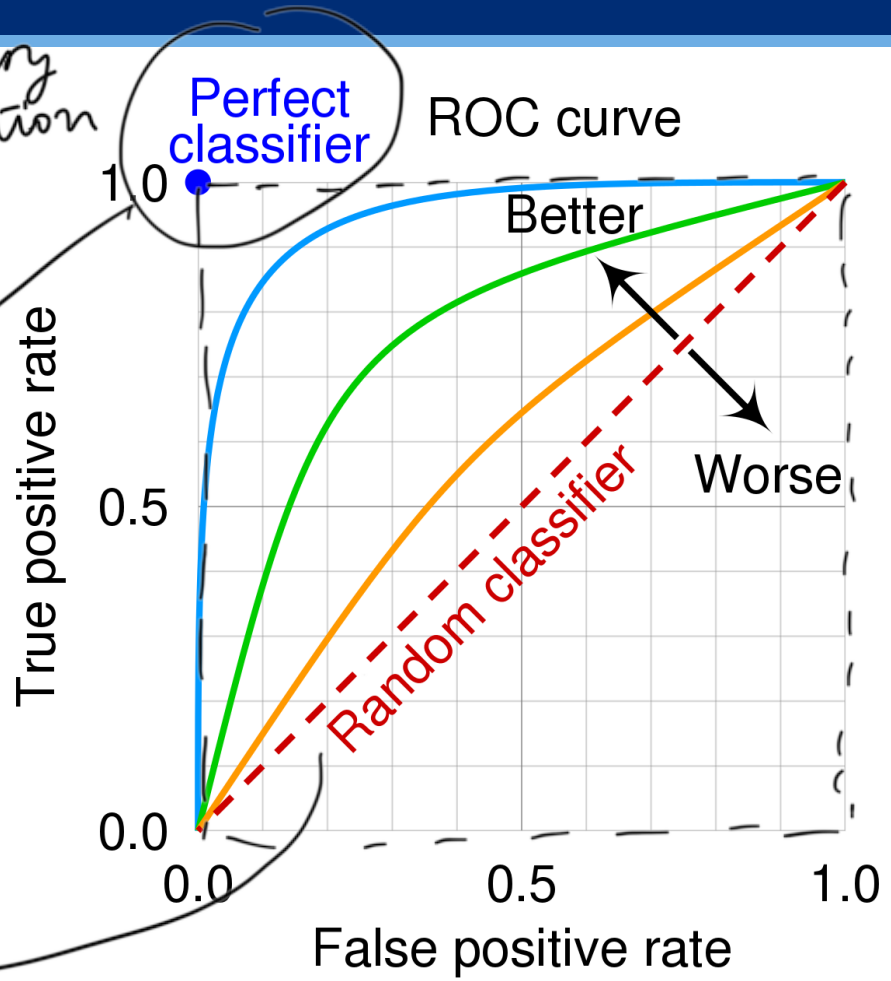
- High precision preferred when false positives are costly
- High recall preferred when false negatives are costly

Notes.

ROC Curve and AUC

used to evaluate performance of binary classification

- ROC: Receiver operating characteristic
  - Plot **sensitivity** against **1-specificity**
  - Or **true positive rate** against **false positive rate**
- AUC: area under the curve
- Ideal model at (0,1)
  - 100% true positives, 0% false positives
  - AOC = 1
- Random guess: diagonal line
  - AOC = 0.5



https://medium.com/@ilyurek/roc-curve-and-auc-evaluating-model-performance-c2178008b02

Lab 3

- Evaluate supervised machine learning approaches on the MNIST dataset
  - MNIST dataset introduced in lecture 2
- Binary classifier: digit 2 vs others
- Multiclass classifier: all 10 classes

Next Week

- Supervised machine learning models and training, part II
- K-nearest neighbors
- Boosting with tree structures