Module 4 – Part I
Machine Learning Fundamentals

JON M. HUNTSMAN SCHOOL OF BUSINESS
UtahStateUniversity

Pedram Jahangiry

Class Modules

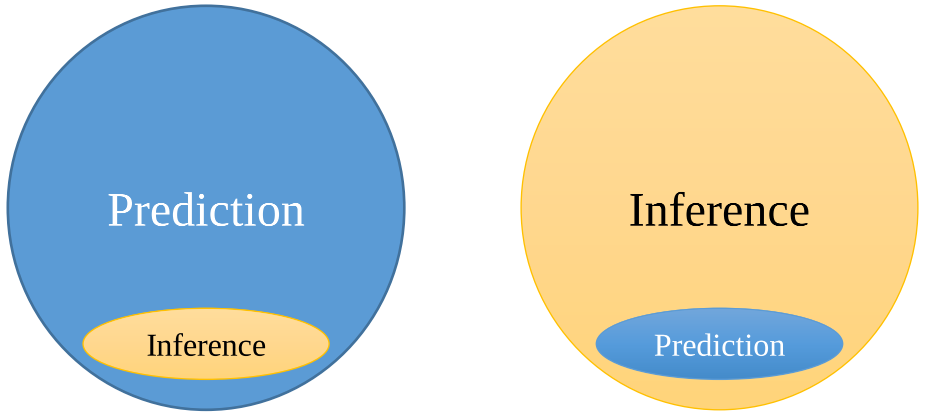
- Module 1- Introduction to Deep Learning
- Module 2- Setting up Machine Learning Environment
- Module 3- Linear Regression (Econometrics approach)
- Module 4- Machine Learning Fundamentals
- Module 5- Linear Regression (Machine Learning approach)
- Module 6- Penalized Regression (Ridge, LASSO, Elastic Net)
- Module 7- Logistic Regression
- Module 8- K-Nearest Neighbors (KNN)
- Module 9- Classification and Regression Trees (CART)
- Module 10- Bagging and Boosting
- Module 11- Dimensionality Reduction (PCA)
- Module 12- Clustering (KMeans – Hierarchical)

Jon M. Huntsman School of Business Utah State University

Pedram Jahangiry

Topics

- Inference vs Prediction
- The Model
- Train, Test, Validation
- Resampling methods
- Evaluation metrics
- Bias-Variance tradeoff, overfitting, learning curve
- How do machines actually learn?
    - Cost Function
    - Solvers/learners (GD, SGD)
- Scaling the features



The Model

The Model

$$y = f(X, \theta) + \epsilon = f(X_1, X_2 \; \dots, X_m, \theta_1, \theta_2, \dots, \theta_k) + \epsilon$$

y : response, dependent variables, output, Target

X: predictors, independent variables, input, Features

θ: estimates, specifications, Parameters

✓ It is all about estimating f by $\hat{f}$ for two purposes:

1) Inference (interpretable ML)
2) Prediction

Parameters and Hyperparameters

$$y = f(X, \theta) + \epsilon = f(X_1, X_2\ \ldots, X_m, \theta_1, \theta_2, \ldots, \theta_k) + \epsilon$$

Model parameters are estimated from data automatically and model hyperparameters are set manually (prior to training the model) and are used in processes to help estimate model parameters.

Example?

JON M. HUNTSMAN SCHOOL OF BUSINESS UtahStateUniversity

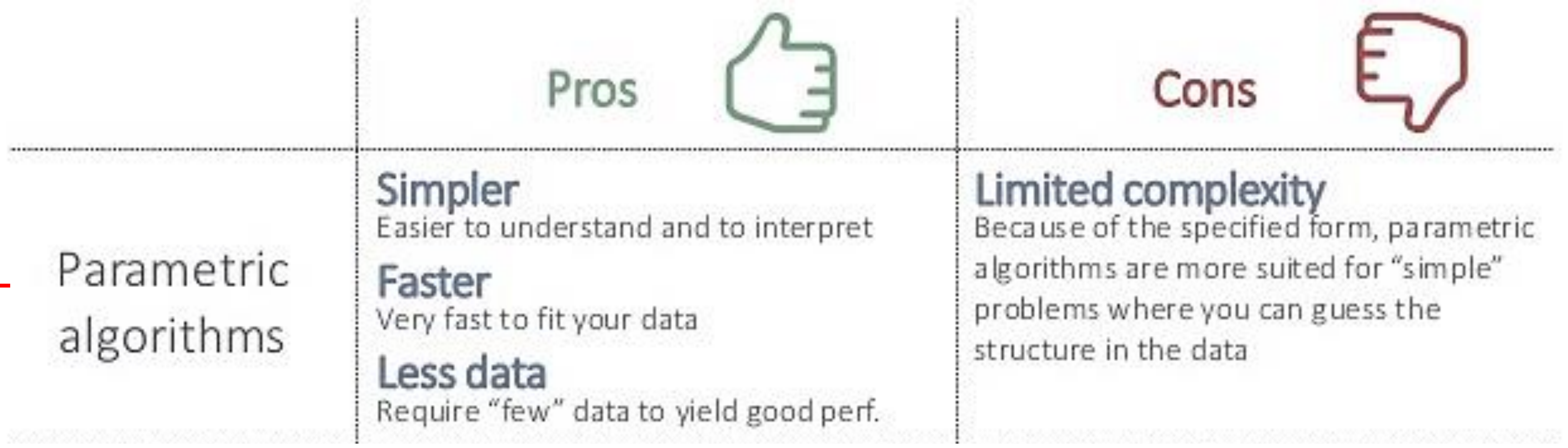
Pedram Jahangiry

Parametric Vs. Nonparametric models

$y = f(X, \theta) + \epsilon$

- The true relationship, $f(X)$ is unknown and the goal is to see which ML algorithm is better at approximating it. An algorithm learns/estimates $f(X)$ from training data.

$f(X)$ is assumed. Examples: Linear regression, GLM, logistic regression, simple Neural networks, ….



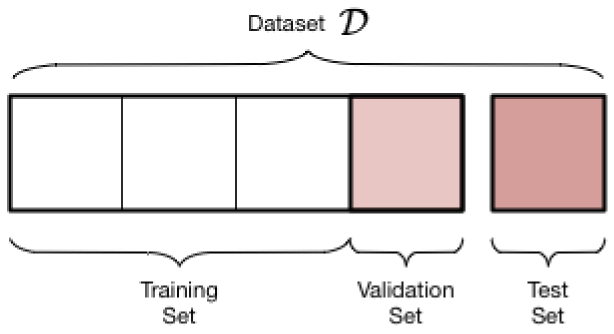
	Pros	Cons
Parametric algorithms	**Simpler** Easier to understand and to interpret **Faster** Very fast to fit your data **Less data** Require "few" data to yield good perf.	**Limited complexity** Because of the specified form, parametric algorithms are more suited for "simple" problems where you can guess the structure in the data

Train, Validation, Test
Resampling Methods

Partitioning of the dataset

The data set is typically divided into three non-overlapping samples:

1) Training set: to train the model
2) Validation set: to validate and tune the model
3) Test set: to test the model’s ability to predict well on new data (generalize)

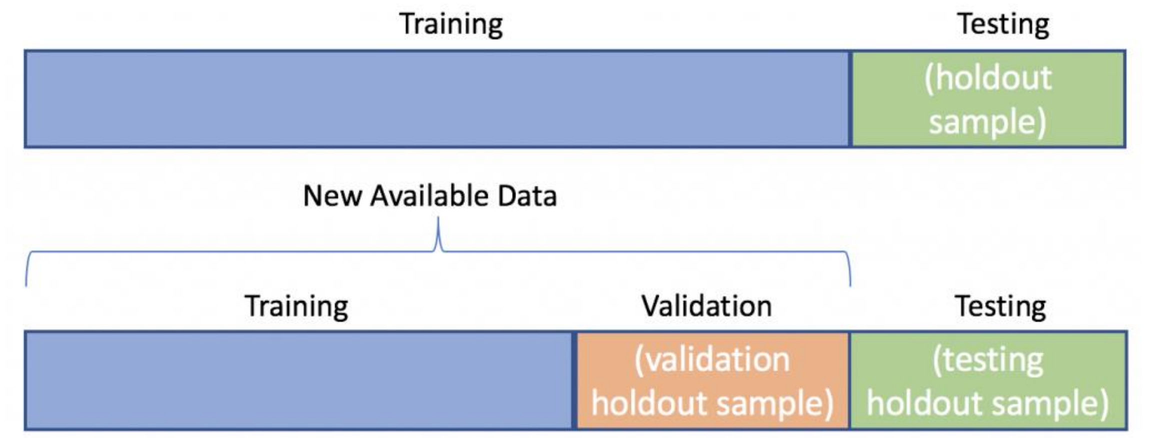


To be valid and useful, any supervised machine learning model must generalize well beyond the training data.

Large dataset is needed! But what if we don’t have it?

Resampling methods

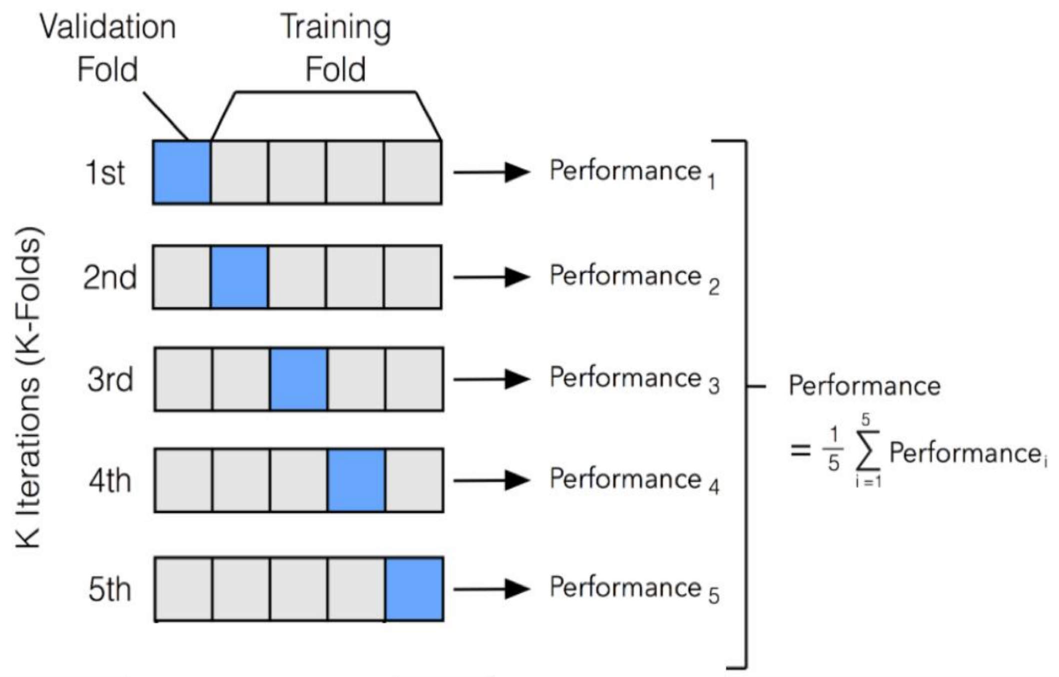
- Sometimes we cannot afford to split the data in three because the algorithm may not learn anything from a small training dataset!
- Small validation set is also problematic because we cannot tune the hyperparameters properly! Unstable model performance in validation set!
- Solution: combining the training and validation sets and use cross validation!



K-fold Cross Validation

1) Divide the training data into K roughly equal-sized non-overlapping groups. Leave out k^{th} fold and fit the model to the other $k-1$ folds. Finally, obtain predictions for the left-out k^{th} fold.

2) This is done in turn for each part k and then the results are combined.



Why do we use Cross Validation?

Cross validation is mainly used for two purposes:

1. Model architecture selection (optimization vs generalization)
2. Estimation of model performance in the test set

- After selecting the best model architecture, we estimate the generalization error using the test set.
- Different model comparison is based on test set performance!

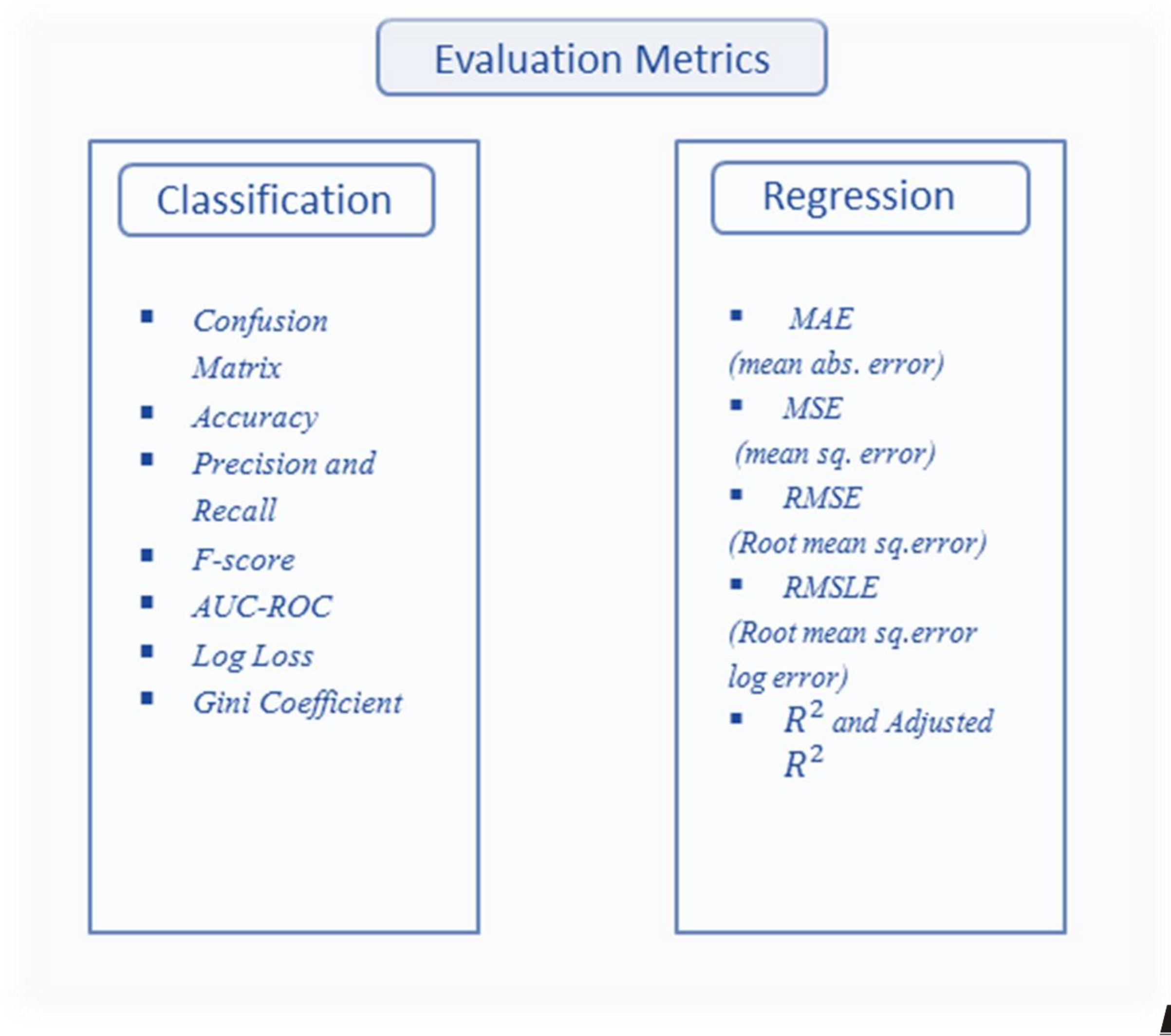
Evaluation Metrics

Evaluation metrics

In general, we want to compare how close are the predictions to the actual numbers in the test set.

This is typically assessed using

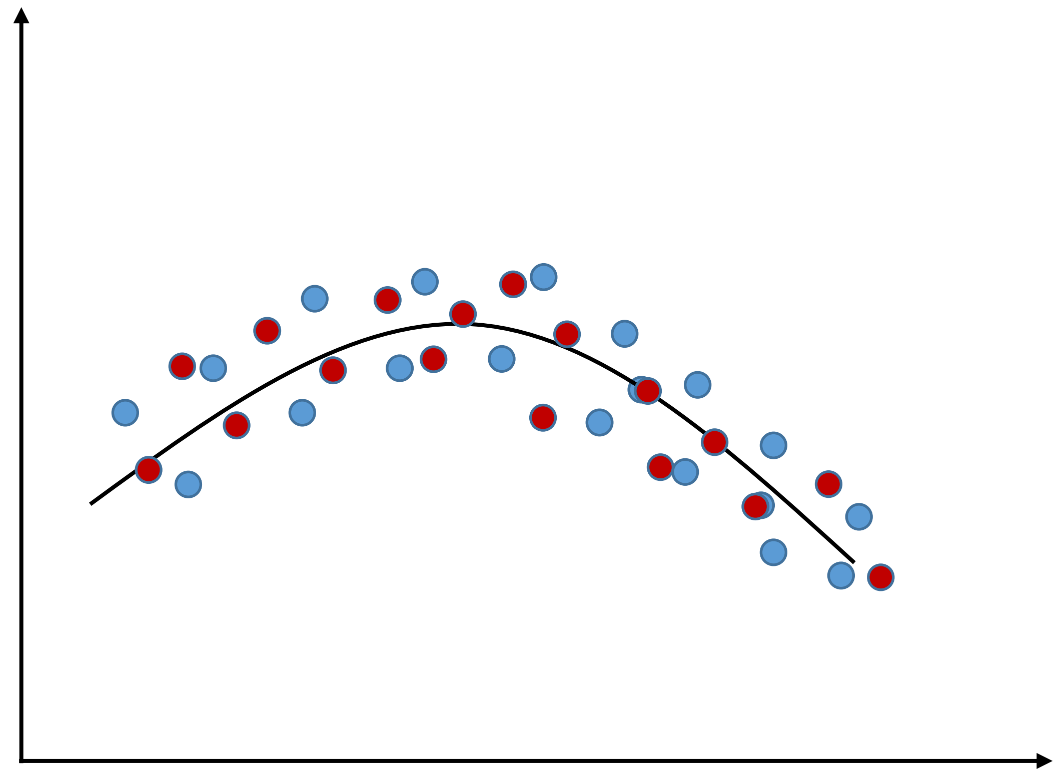
- MSE for quantitative response
- Misclassification rate for qualitative response



Bias-Variance Tradeoff

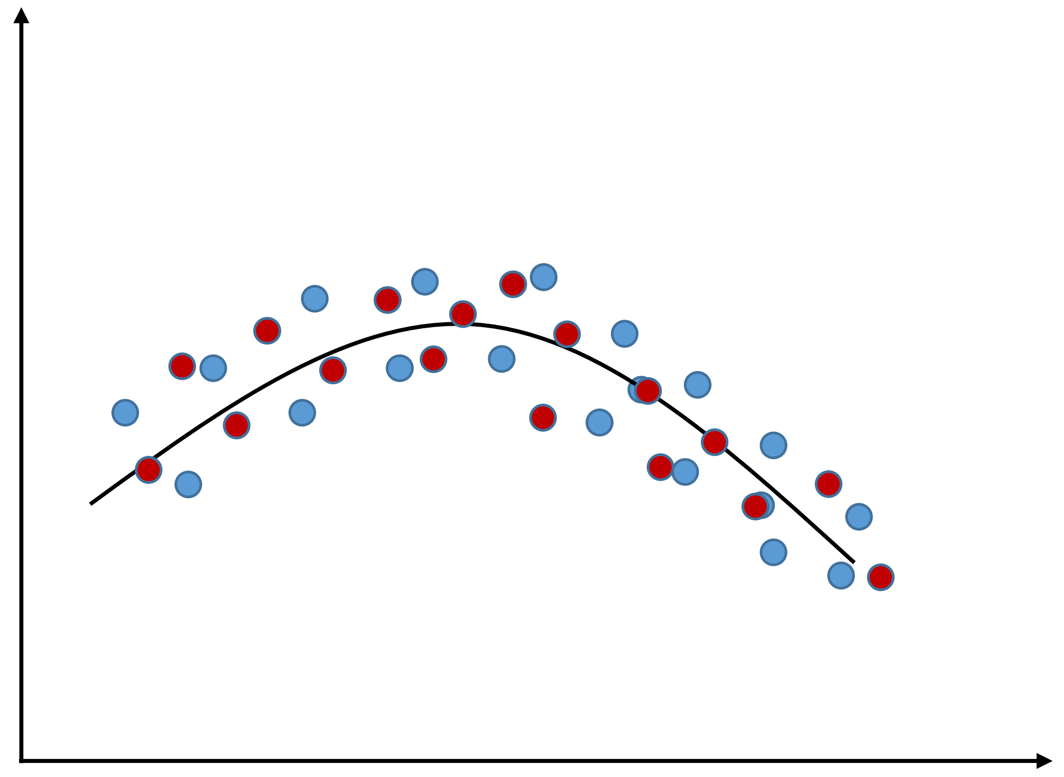
Optimization vs Generalization

Model Bias & Model Variance in machine learning



Prof. Pedram Jahangiry

Model Bias & Model Variance in machine learning



MSE decomposition

$$MSE = model\ variance + model\ bias + irreducible\ error$$

1) Model variance is the variance if we had estimated the model with a different **training set**
2) Model bias is the error due to using an approximate model (model is too simple)
3) Irreducible error is due to missing variables and limited samples. Can't be fixed with modeling

- The goal is to minimize the sum of model variance and model bias.
- This is known as the bias-variance tradeoff because reducing one often leads to increasing the other.
- Choosing the flexibility (complexity) of $\hat{f}(X)$, will amount to bias-variance tradeoff.

MSE decomposition

The bias-variance tradeoff is one of the core concepts in supervised learning.

Assume that the data is generated by a simple model!

$$y_i = f(\mathbf{x}_i) + \epsilon_i, \quad \mathbb{E}[\boldsymbol{\epsilon}] = 0, \quad \mathbb{V}[\boldsymbol{\epsilon}] = \sigma^2$$

The estimated model yields

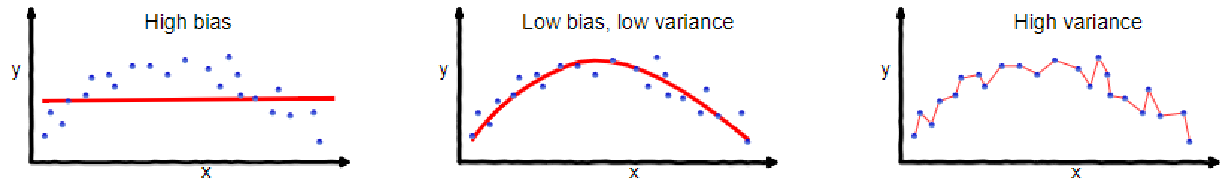
$$\hat{y}_i = \hat{f}(X_i)$$

Let us decompose the mean squared error (MSE):

$$\mathbb{E}[\hat{\epsilon}^2] = \mathbb{E}[(y - \hat{f}(\mathbf{x}))^2] = \mathbb{E}[(f(\mathbf{x}) + \epsilon - \hat{f}(\mathbf{x}))^2]$$
$$= \underbrace{\mathbb{E}[(f(\mathbf{x}) - \hat{f}(\mathbf{x}))^2]}_{\text{total quadratic error}} + \underbrace{\mathbb{E}[\epsilon^2]}_{\text{irreducible error}} \quad \ldots\ldots$$

$$= \underbrace{\mathbb{V}[\hat{f}(\mathbf{x})]}_{\text{variance of model}} + \underbrace{\mathbb{E}[(f(\mathbf{x}) - \hat{f}(\mathbf{x}))]^2}_{\text{squared bias}} + \sigma^2$$

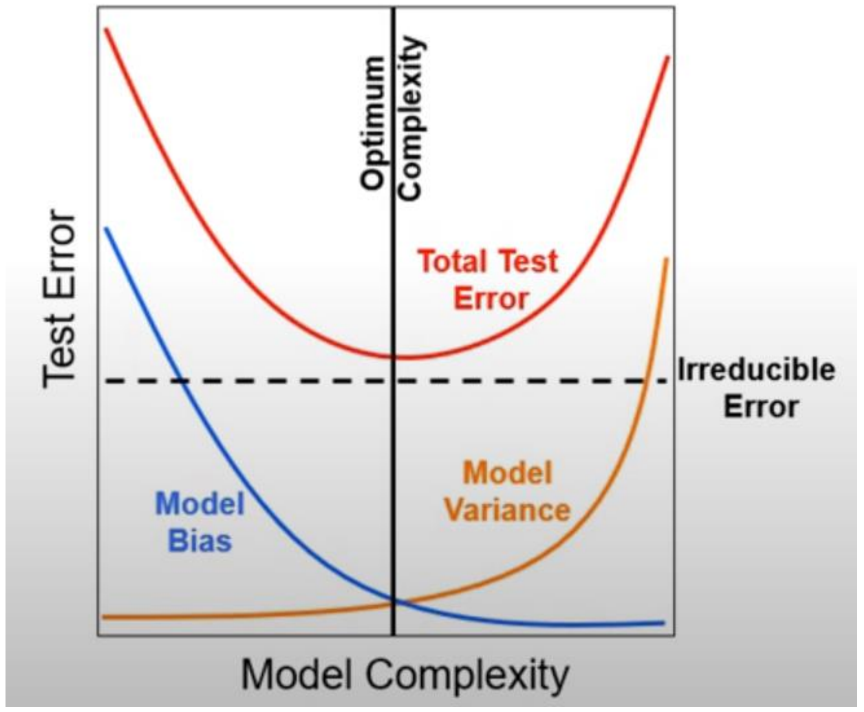
Representations of the bias-variance tradeoff



underfitting

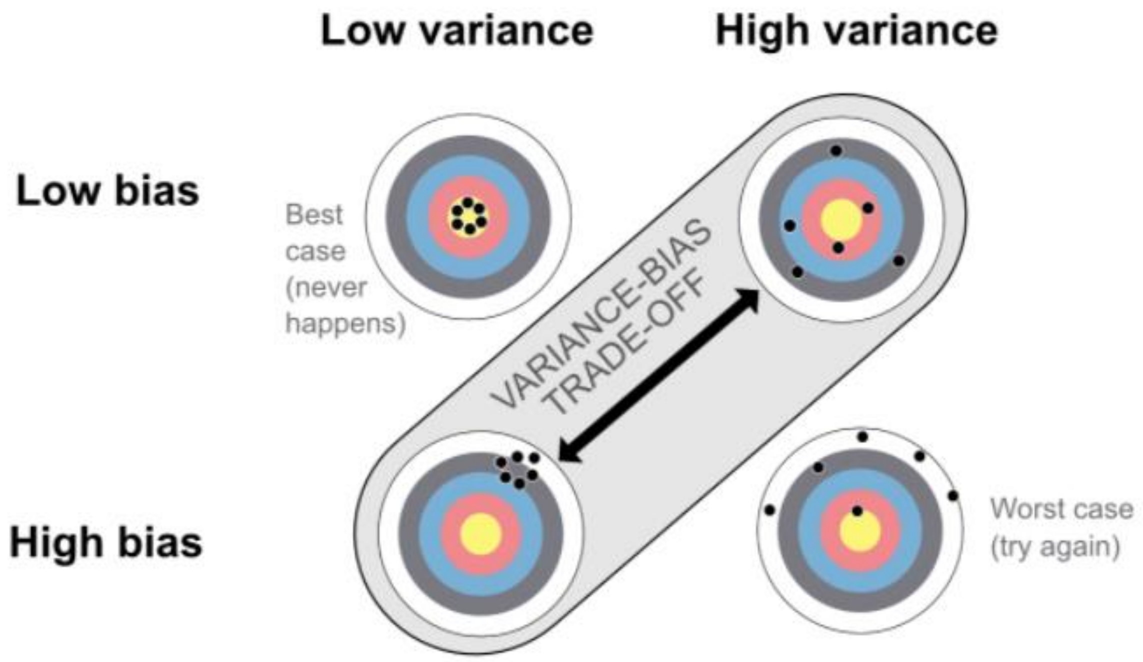
Good balance

overfitting



Optimization
Vs
Generalization

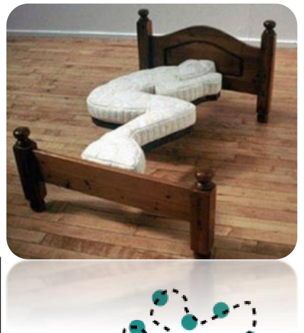
Other representations of the bias-variance tradeoff

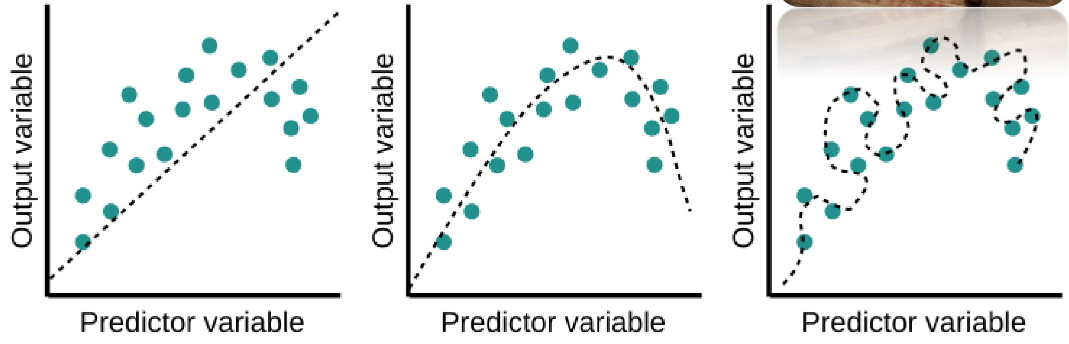


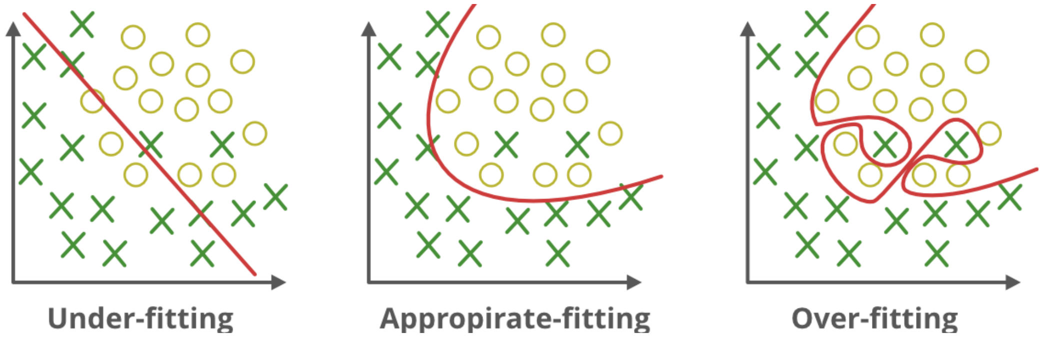
Overfitting

Overfitting happens when the fitted algorithm does not generalize well to new data:

- The model fits the training data too well while not predicts well in the new data
- The model fits the noise (ϵ) in training data (finds a pattern that does not exist)
- The algorithm has simply memorized the data, rather than learned from it!
- The model is too complex!







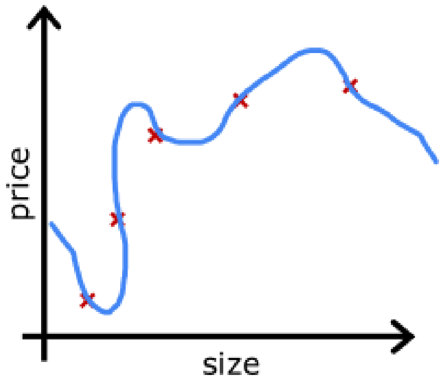
Mitigate overfitting

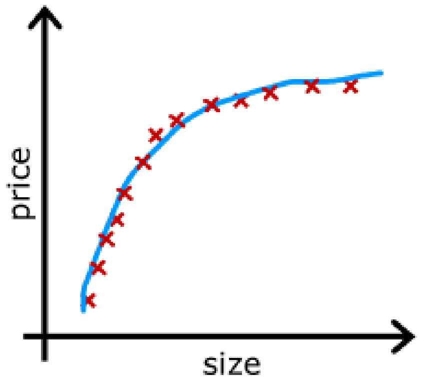
The main techniques used to mitigate overfitting risk in a model construction are:

1) Collect more data (Can reduce AND variance)

2) Complexity reduction (regularization, feature selection)

3) Cross validation (estimate the performance in test set)





With more training example

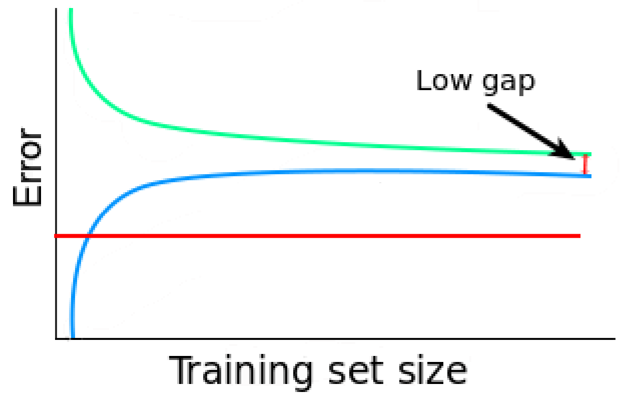
JON M. HUNTSMAN SCHOOL OF BUSINESS UtahStateUniversity

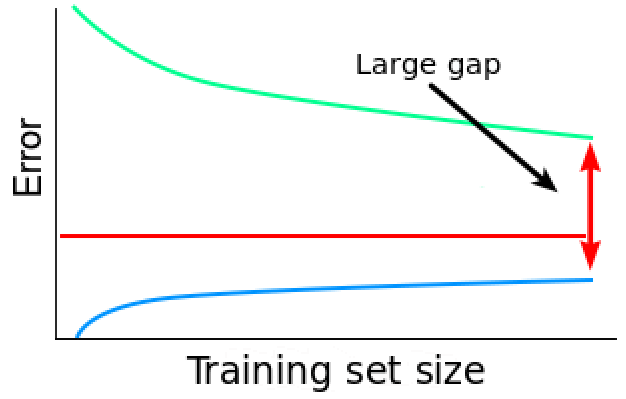
Pedram Jahangiry

The Learning Curve: Do we need to collect more data?

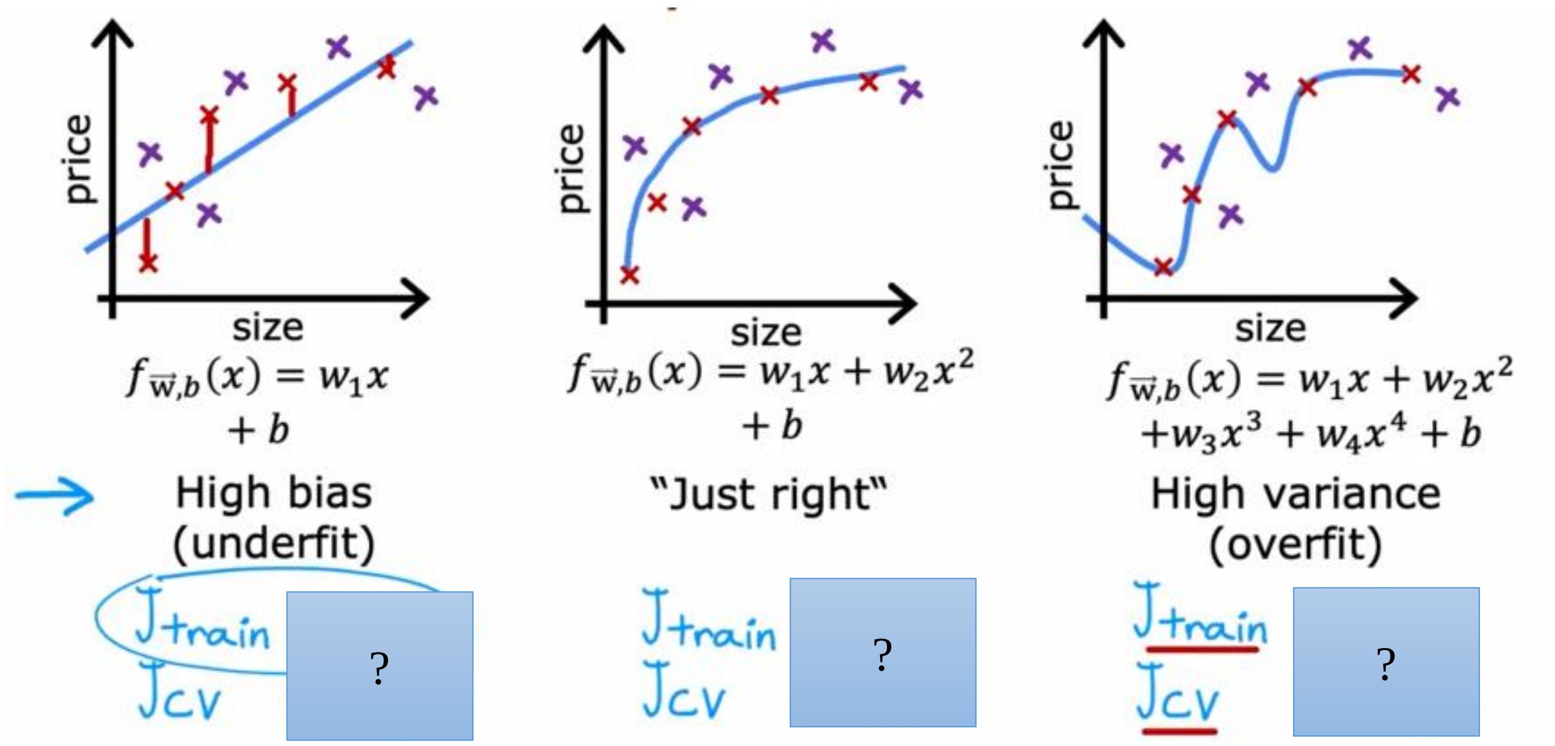
- A learning curve is a plot that shows the relationship between the amount of training data and the performance of a machine learning model.
- It is used to diagnose whether a model has high bias, high variance, or is just right.

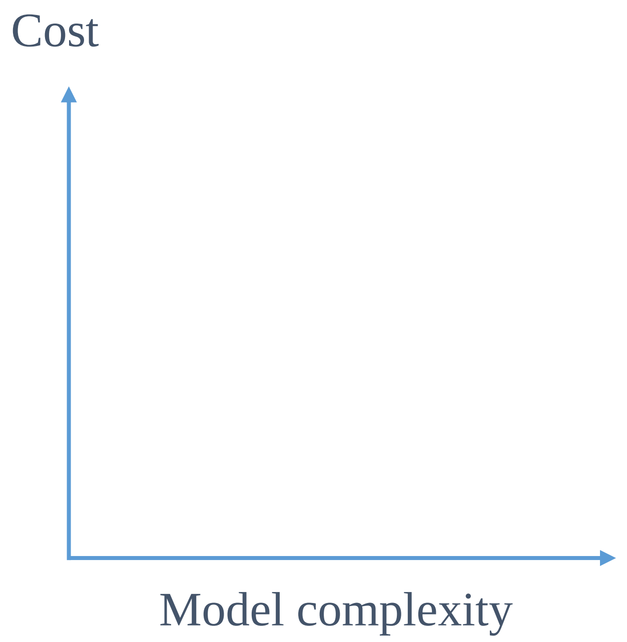
-- Training score
-- Cross validation score
-- Benchmark performance
(common sense performance)





Class exercise

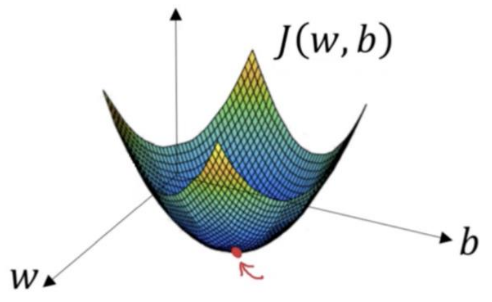




How do Machines Learn?

Terminology

- Learning: Finding the model weights (parameters' values)
- Cost Function: Tells us "how good" our model is at making predictions for a given set of parameters.
- The cost function has its own curve and its own gradients. The slope of this curve tells us how to update our parameters to make the model more accurate.
- The two most frequently used optimization algorithms when the cost function is continuous and differentiable are Gradient Descent (GD) and Stochastic GD.



Solvers (learners)!

- Gradient Descent: is an iterative optimization algorithm for finding the minimum of a function.
- We starts at some random point and take steps proportional to the negative of the gradient of the function at the current point.

$$\theta_j \coloneqq \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

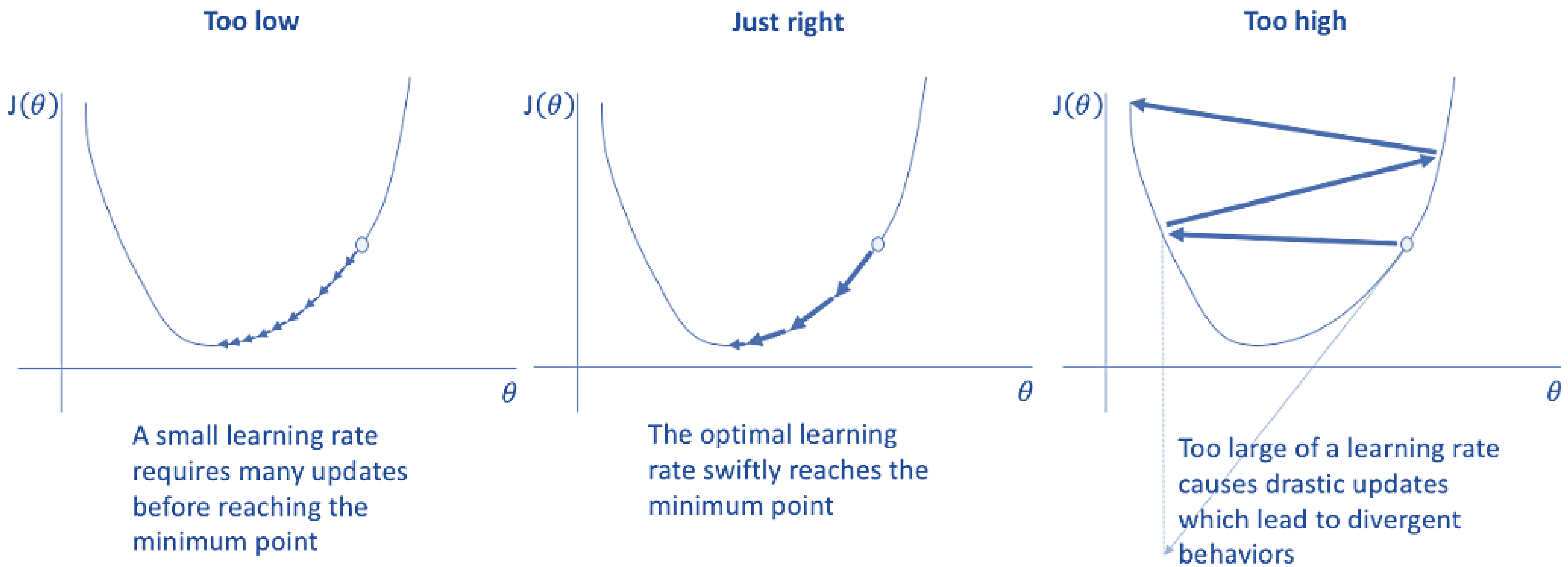
- θ_j is the model's j^{th} parameter
- α is the learning rate
- $J(\theta)$ is the cost function (which is differentiable)

JON M. HUNTSMAN SCHOOL OF BUSINESS
UtahStateUniversity

Pedram Jahangiry

Choice of learning rate

- If α is too small, gradient descent can be slow
- If α is too large, the gradient descent can even **diverge**.


Too low
Just right
Too high
$J(\theta)$
θ
A small learning rate requires many updates before reaching the minimum point
The optimal learning rate swiftly reaches the minimum point
Too large of a learning rate causes drastic updates which lead to divergent behaviors

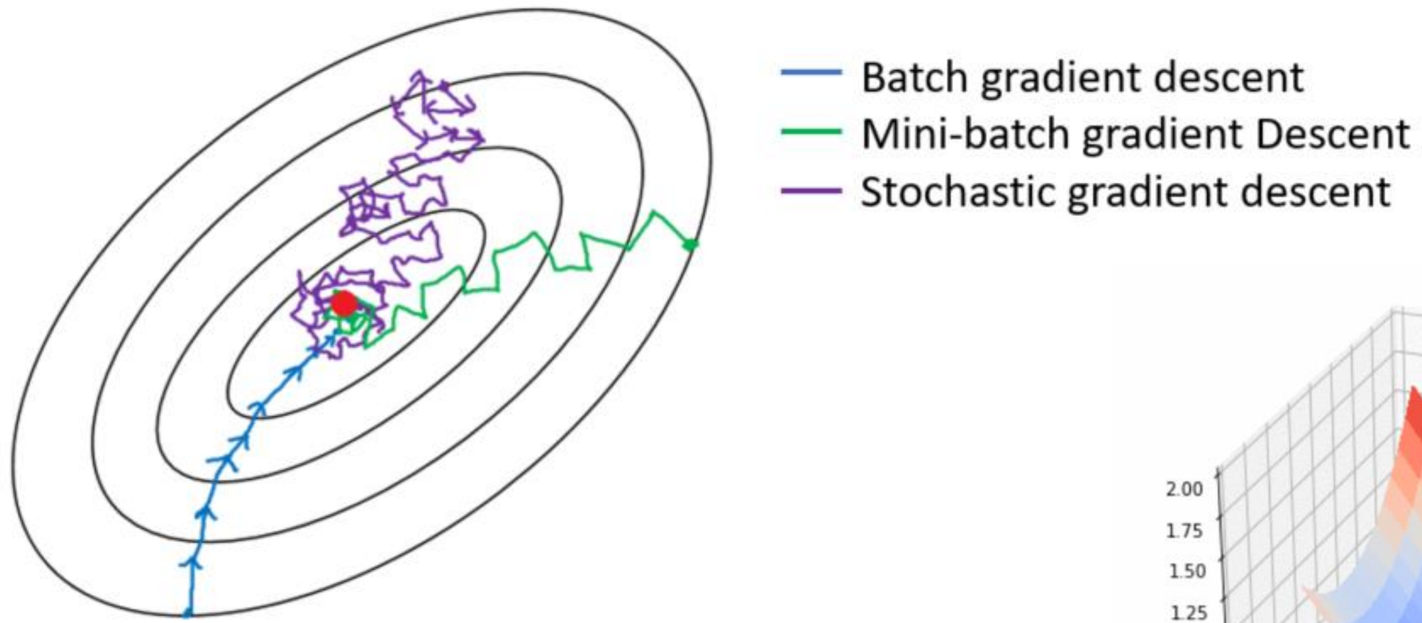
Beyond Gradient Descent?

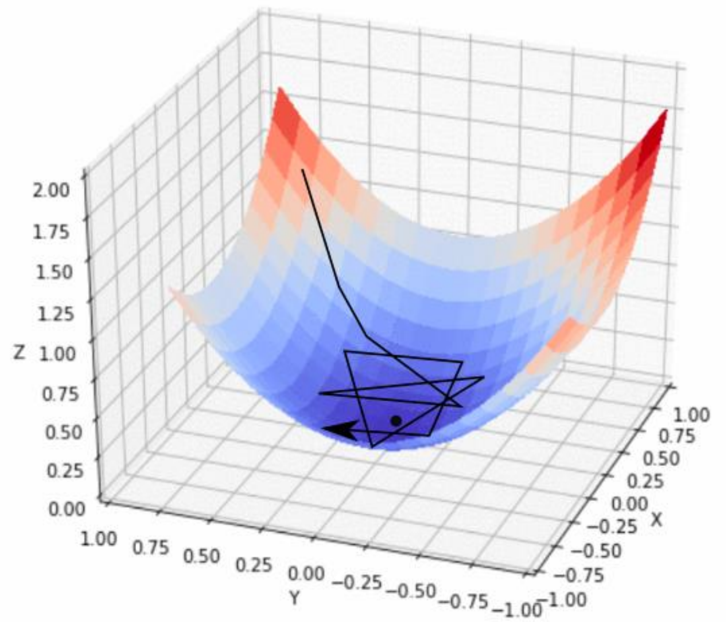
Disadvantages of gradient descent:

- Single batch: use the entire training set to update parameters!
- Sensitive to the choice of the learning rate
- Slow for large datasets

(Minibatch) Stochastic Gradient Descent: is a version of the algorithm that speeds up the computation by approximating the gradient using smaller batches (subsets) of the training data. SGD itself has various “upgrades”.

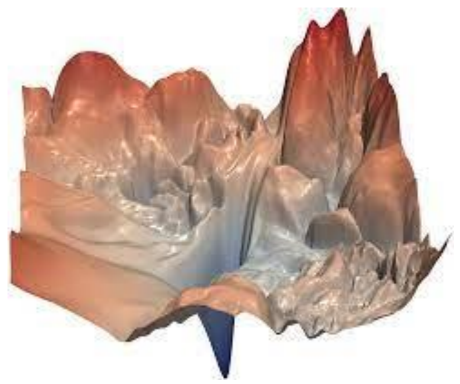
SGD vs GD


Batch gradient descent
Mini-batch gradient Descent
Stochastic gradient descent


2.00
1.75
1.50
1.25
Z 1.00
0.75
0.50
0.25
0.00
1.00 0.75 0.50 0.25 0.00 −0.25 −0.50 −0.75 −1.00
Y
1.00 0.75 0.50 0.25 0.00 −0.25 −0.50 −0.75 −1.00
X

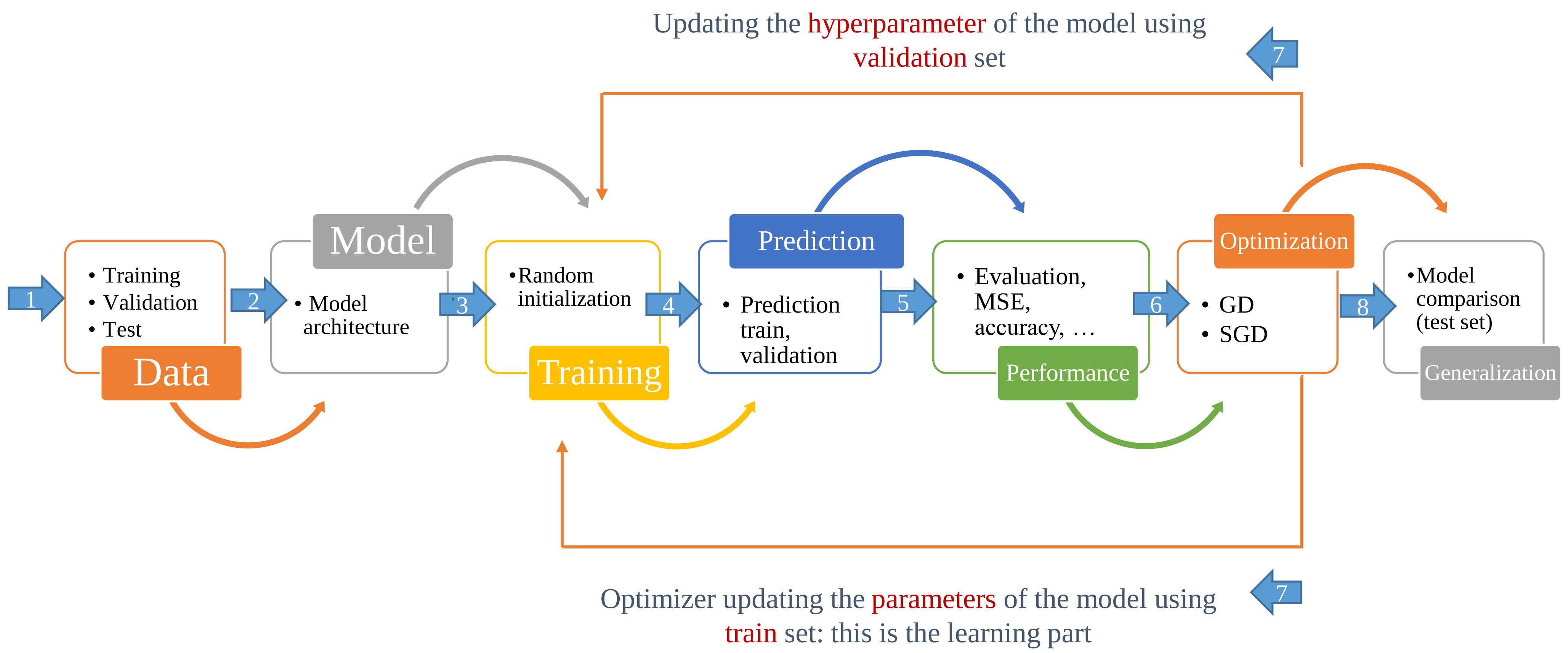
Beyond SGD?

- Loss functions can be difficult to optimize!
- *Visualizing the loss landscape of neural nets*, Li et all, 2018



- Solution: Designing an adaptive learning rate that can adapt to the loss landscape.
- Rather than just looking at the current gradient, consider the previous weight updates.
- This is called, momentum!
- Examples: Adam, Adadelta, Adagrad, RMSProp!

How do machines learn?



Updating the hyperparameter of the model using validation set
7
1
• Training
• Validation
• Test
Data
2
Model
• Model architecture
3
•Random initialization
Training
4
Prediction
• Prediction train, validation
5
• Evaluation, MSE, accuracy, …
Performance
6
Optimization
• GD
• SGD
8
•Model comparison (test set)
Generalization
7
Optimizer updating the parameters of the model using train set: this is the learning part

Jon M. Huntsman School of Business
Utah State University

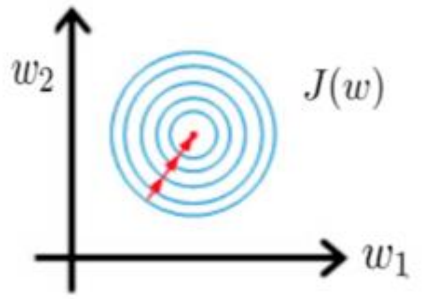
Pedram Jahangiry

Feature Scaling

Why feature scaling?

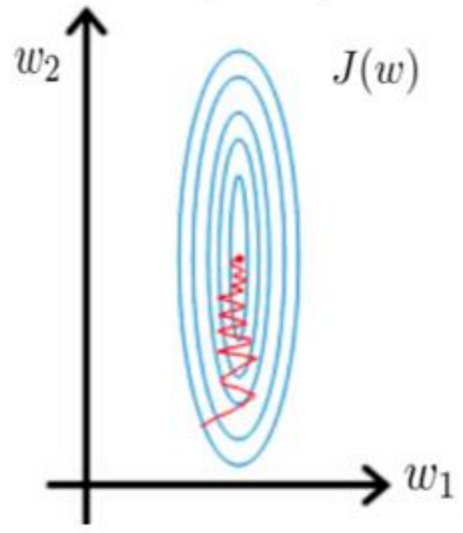
- Feature scaling in machine learning is a critical step during the pre-processing of data before creating a machine learning model.
- Feature scaling is essential for machine learning models that calculate distances between data.
- Feature scaling could:
  - Avoid numerical overflow and speed up the algo
  - Reduce dominant effects of specific variables

$0 \leq x_1 \leq 1$
$0 \leq x_2 \leq 1$



Both parameters could be updated in equal proportions

$x_1 >> x_2$



Gradient of larger parameters dominates the updates

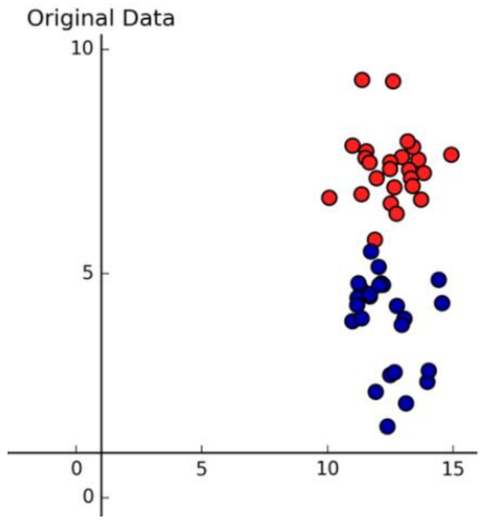
Scaling the features

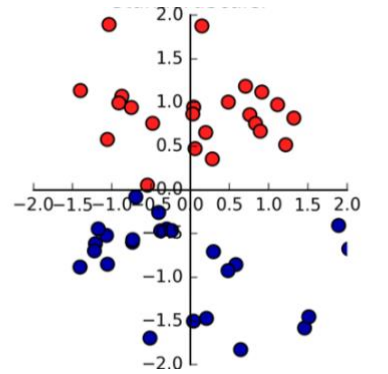
Let us use x_i for raw input and $\widetilde{x_i}$ for the transformed data. Common scaling practices include:

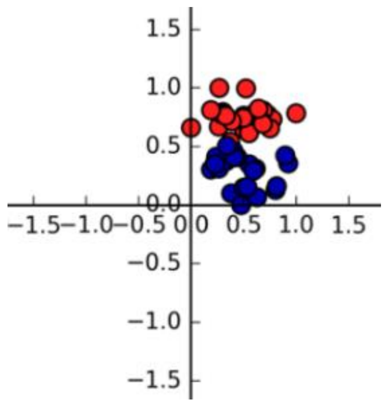
- Standardization (Z-score): $\widetilde{x_i} = \left(\frac{x_i - \mu_x}{\sigma_x}\right)$

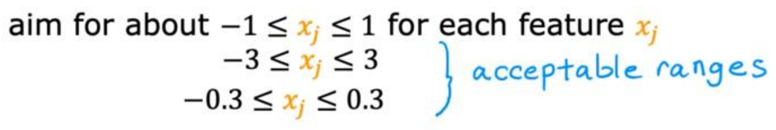
- Normalization:
  - Min-Max scaler over [0,1]: $\widetilde{x_i} = \left(\frac{x_i - \mathrm{Min}(X)}{Max(X) - Min(X)}\right)$
  - Min-Max scaler over [-1,1]: $\widetilde{x_i} = 2 * \left(\frac{x_i - \mathrm{Min}(X)}{Max(X) - Min(X)}\right) - 1$
  - Mean normalization: $\widetilde{x_i} = \left(\frac{x_i - \mathrm{Mean}(X)}{Max(X) - Min(X)}\right)$

Scaling the features (class exercise)









aim for about $-1 \le x_j \le 1$ for each feature x_j

$-3 \le x_j \le 3$

$-0.3 \le x_j \le 0.3$

acceptable ranges

$0 \le x_1 \le 3$

$-2 \le x_2 \le 0.5$

$-100 \le x_3 \le 100$

$-0.001 \le x_4 \le 0.001$

$98.6 \le x_5 \le 105$

Which of these ranges need to be scaled?

Some general hints with scaling

- Be careful when scaling the time series data! Why?
- To avoid data leakage, it is a good practice to fit the scaler on the training data and then use it to transform the testing data.
- Scaling the data does NOT change the shape of the distributions.

Class Modules

- ✓ Module 1- Introduction to Deep Learning
- ✓ Module 2- Setting up Machine Learning Environment
- ✓ Module 3- Linear Regression (Econometrics approach)
- ✓ Module 4- Machine Learning Fundamentals
- Module 5- Linear Regression (Machine Learning approach)
- Module 6- Penalized Regression (Ridge, LASSO, Elastic Net)
- Module 7- Logistic Regression
- Module 8- K-Nearest Neighbors (KNN)
- Module 9- Classification and Regression Trees (CART)
- Module 10- Bagging and Boosting
- Module 11- Dimensionality Reduction (PCA)
- Module 12- Clustering (KMeans – Hierarchical)

Jon M. Huntsman School of Business
UtahStateUniversity

Pedram Jahangiry