APS1070

Foundations of Data Analytics and Machine Learning

Winter 2022

Week 12:

- *Review Concepts*
- *Final Assessment Details*
- *Past Final Exam Questions*

Sinisa Colic and Samin Aref

Example: Set Cover Problem

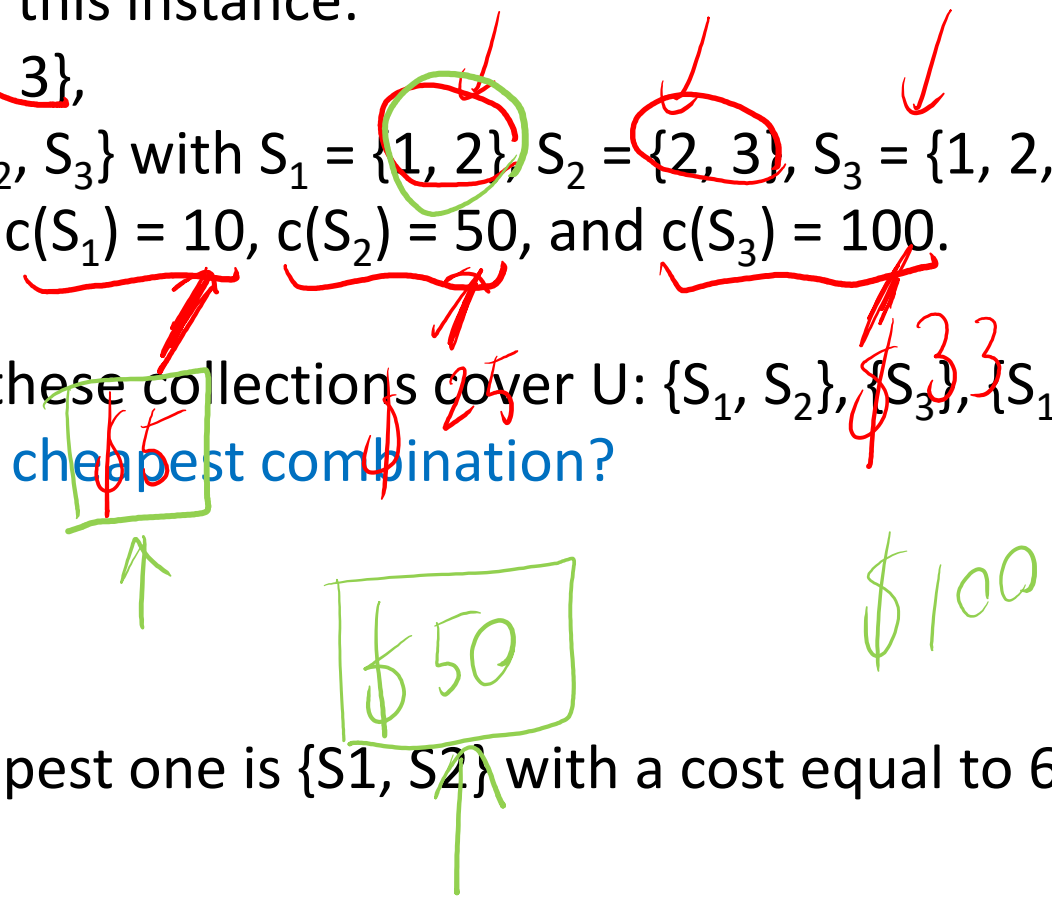
- Set cover is a classical problem in combinatorics!
- Given a universe ***U*** of ***n*** elements (**U = {1,2,…*n*}**), a collection of subsets ***S*** ={S_1,…,S_k} of ***U***, what is the smallest/cheapest sub-collection of ***S*** whose union equals the universe ***U***.
- A Cover is a subfamily ***C*** of sets (from ***S***) for which the union is ***U***

- For example:
  - Consider a universe *U* = {1,2,3,4,5} and the collection of sets *S* = {{1,2,3}, {2,4}, {3,4}, {4,5}}. Identify the smallest sub-collection of *S* whose union equals the universe.

Example with Cost Associations

Q: Consider this instance:

- U = {1, 2, 3},
- S = {S_1, S_2, S_3} with S_1 = {1, 2}, S_2 = {2, 3}, S_3 = {1, 2, 3}
- and cost $c(S_1) = 10$, $c(S_2) = 50$, and $c(S_3) = 100$.



Given that these collections cover U: {S_1, S_2}, {S_3}, {S_1, S_3}, {S_2, S_3}, {S_1, S_2, S_3}.

What is the cheapest combination?

A: The cheapest one is {S1, S2} with a cost equal to 60.

More Formally

Problem 5.1 SET COVER

Instance. Universe U with n elements, collection $\mathcal{S} = \{S_1, \ldots, S_k\}$, $S_i \subseteq U$, a cost function $c : \mathcal{S} \to \mathbb{R}$.

Task. Solve the problem

Minimize cost of sets (or # of sets, if costs are 1)

$$\text{minimize} \quad \text{val}(x) = \sum_{S \in \mathcal{S}} c(S) x_S,$$

All elements are covered (at least once)

$$\text{subject to} \quad \sum_{S : e \in S} x_S \geq 1 \quad e \in U,$$

$$x_S \in \{0, 1\} \quad S \in \mathcal{S}.$$

Variable indicating whether it's chosen or not

Source: Alexander Souza

The Greedy Algorithm

- Iteratively pick the most cost-effective set and remove the covered elements, until all elements are covered.

Input. Universe U with n elements, collection $\mathcal{S} = \{S_1, \ldots, S_k\}$, $S_i \subseteq U$, a cost function $c : \mathcal{S} \to \mathbb{R}$.

Output. Vector $x \in \{0, 1\}^k$

C -> sets of elements already covered, x -> vector of chosen sets

Step 1. $C = \emptyset$, $x = 0$.

Step 2. While $C \neq U$ do the following: *Until we have all elements of U covered*

(a) Find the most cost-effective set in the current iteration, say S.

(b) Set $x_S = 1$ and for each $e \in S - C$ set $\text{price}(e) = c(S)/|S - C|$.

(c) $C = C \cup S$.

Cost of set / Elements not yet added

Cost-effectiveness of a set S – the average cost of covering new elements

Step 3. Return x.

#new item

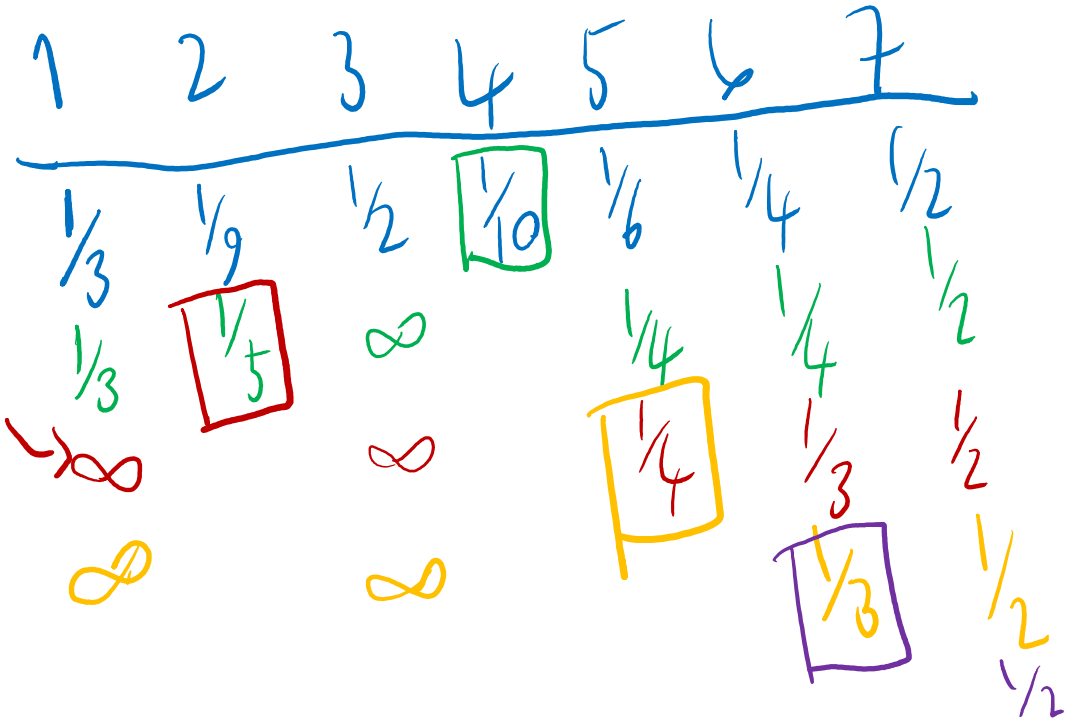
$C = \{\}$

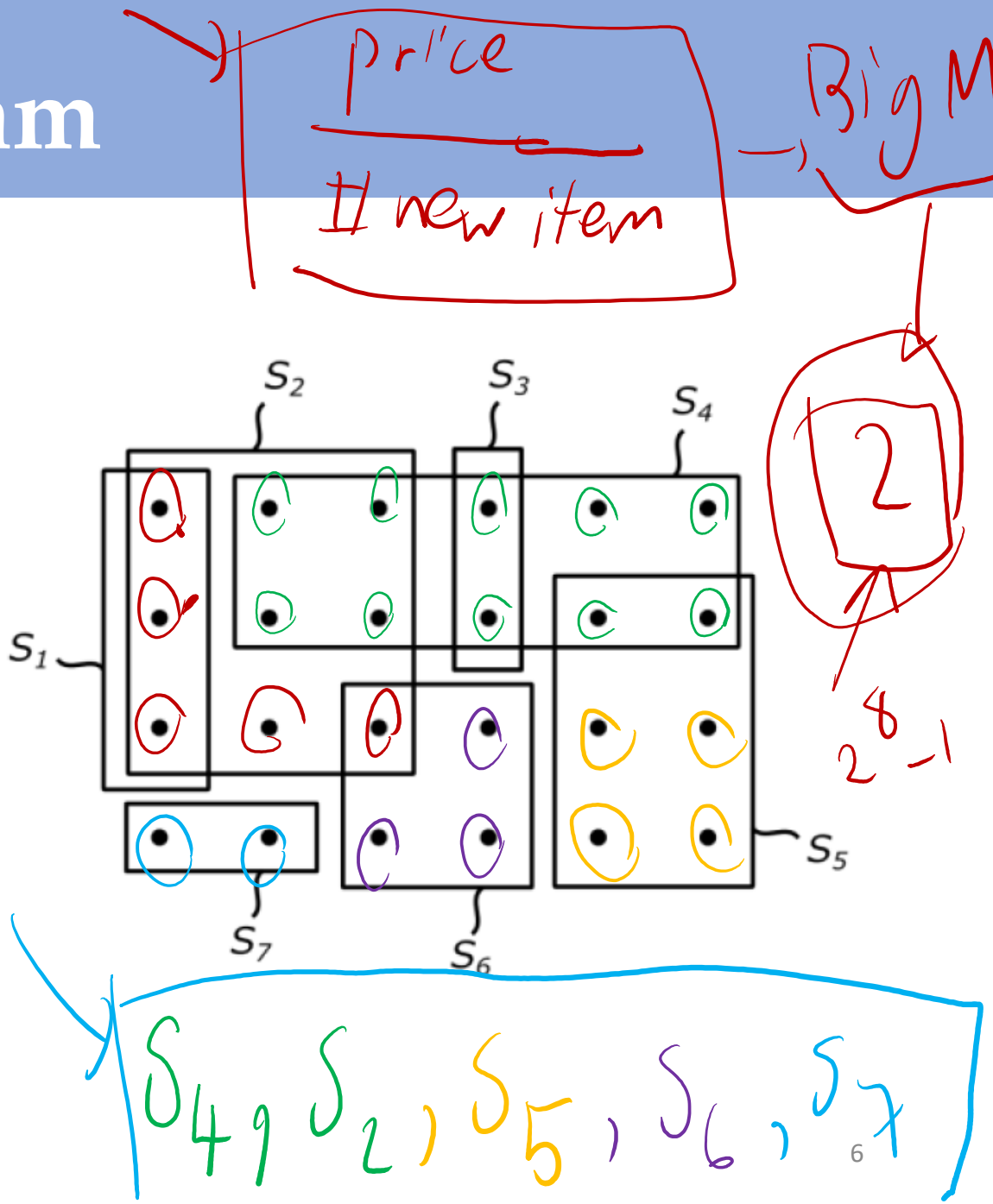
$C = S_4$

$C = S_4 \cup S_2$

Example: Past Final Exam

The schematic to the right has sets S_1, S_2, S_3, S_4, S_5, S_6 and S_7. What sets, and in what order, would a greedy algorithm select to cover the universe (i.e., cover each point) if all sets are weighted equally?

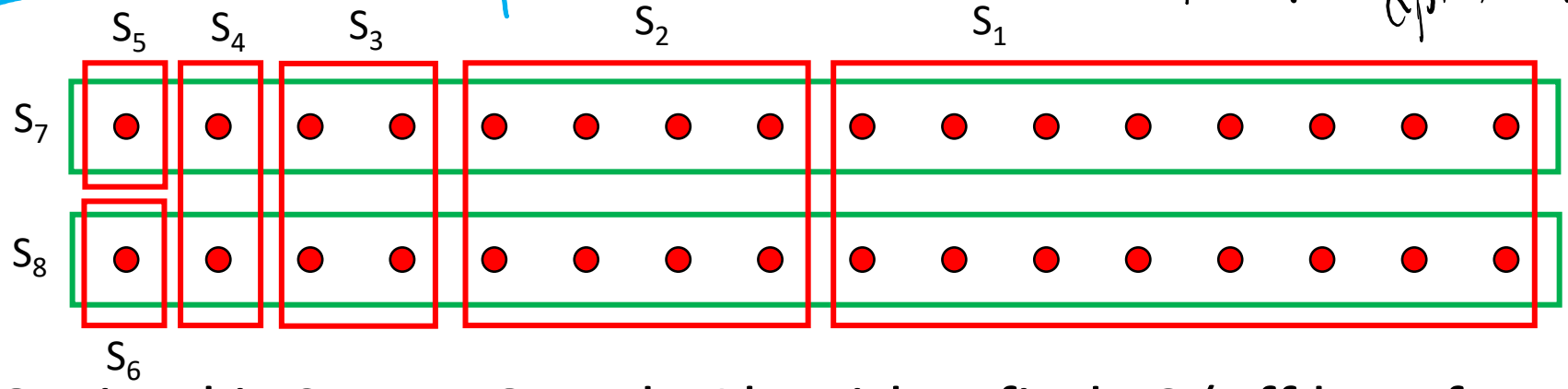




Approximation factor

Greedy algorithm

Tie-breaking rule → smallest index



Min heurist 6
optimal 2

Gap / optimal = 4/2

- Optimal is 2 sets, Greedy Algorithm finds 6 (off by a factor of 3)

≥ 200%

S_1 S_7 S_8 S_2 S_3 S_4 S_5 S_6

1/16 1/16 1/16 > > > > >

1/8 1/8 1/8

81! 50!

optimal: S_7, S_4 $2

greedy: $6

Source: Dave Mount

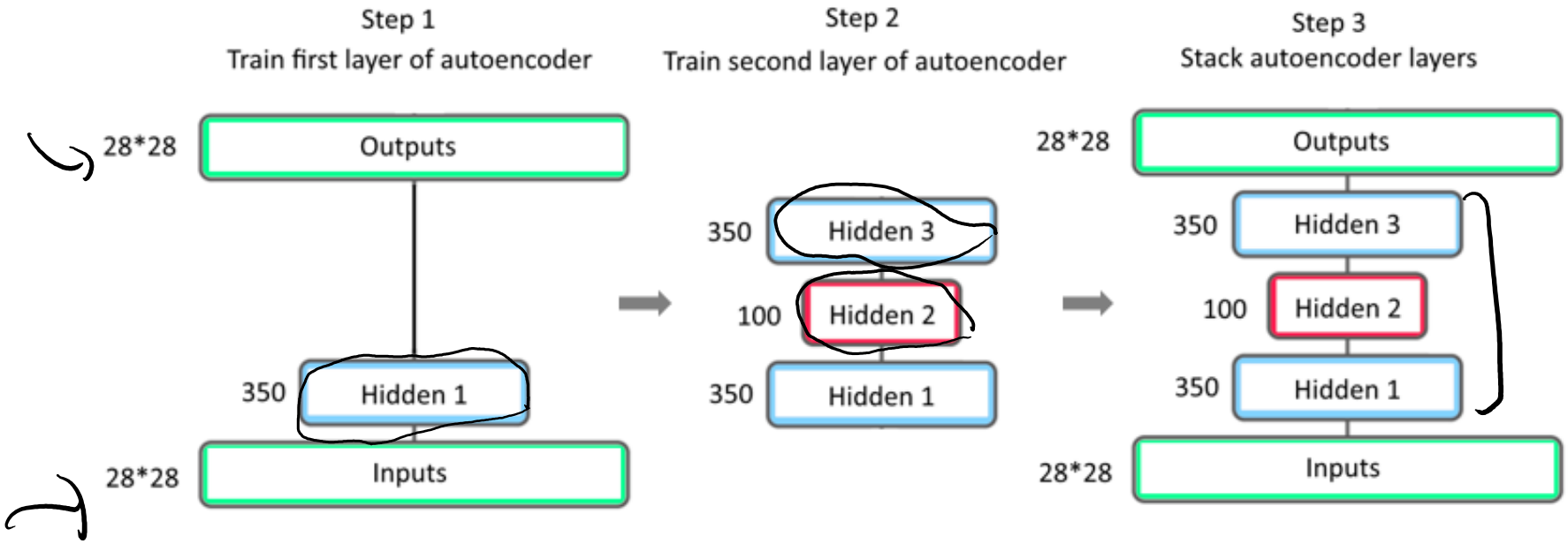
Key Takeaways

- Optimization isn't only continuous (gradient descent), but can also be discrete -> requires a different way of thinking

- Greedy algorithms are sometimes useful methods for obtaining a "good" heuristic solution.

Link to Machine Learning

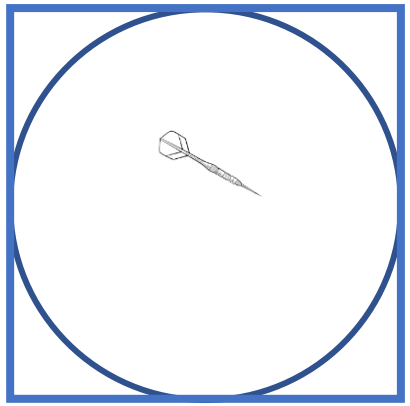
- We used a Greedy Discrete Optimization Algorithm in Project 1 to select the top features.
- Some deep neural networks can be trained using a greedy approach.
- For example, a Deep Autoencoder:



Review of Concepts

Confidence Intervals

- A numerical approach to estimate pi = 3.14159… by simulation



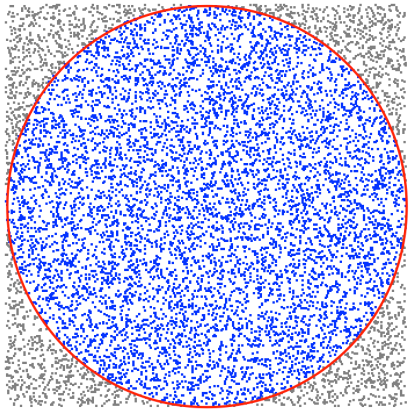
Probability =
area of circle/area of square
$\frac{\pi r^2}{a^2} = \frac{\pi 1^2}{2^2} = \pi/4$
= in_circle/num_darts

---------- Simulation Code ----------

```
def throwDarts(num_darts):
    in_circle = 0
    for darts in range(num_darts):
        x = random.random()
        y = random.random()
        if (x*x + y*y)**0.5 <= 1.0:
            in_circle +=1
    return (4*in_circle/num_darts)

Estimate:  3.13868
Truth:  3.141592653589793
```

one estimate



95 %

[3.1 , 3.2]

Confidence Intervals

- The equation for obtaining the confidence intervals is:

sample mean

$$\bar{x} = \frac{1}{n}\sum_i x_i$$

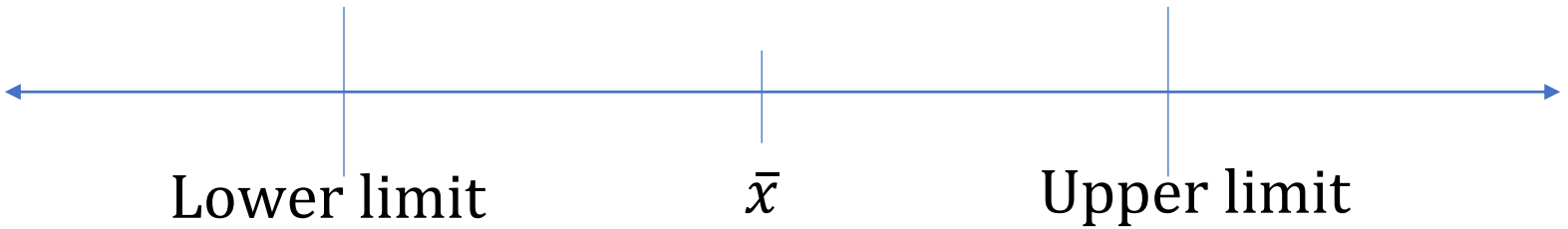
sample standard deviation

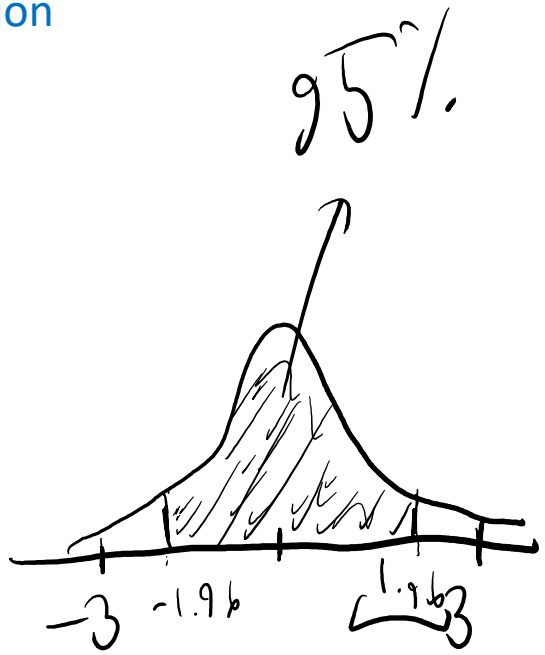
$$s^2 = \sum_i \frac{(x_i - \bar{x})^2}{n-1}$$

$$CI = \bar{x} \pm z\frac{s}{\sqrt{n}}$$

confidence level value
(typically set based on a 95% confidence)

sample size

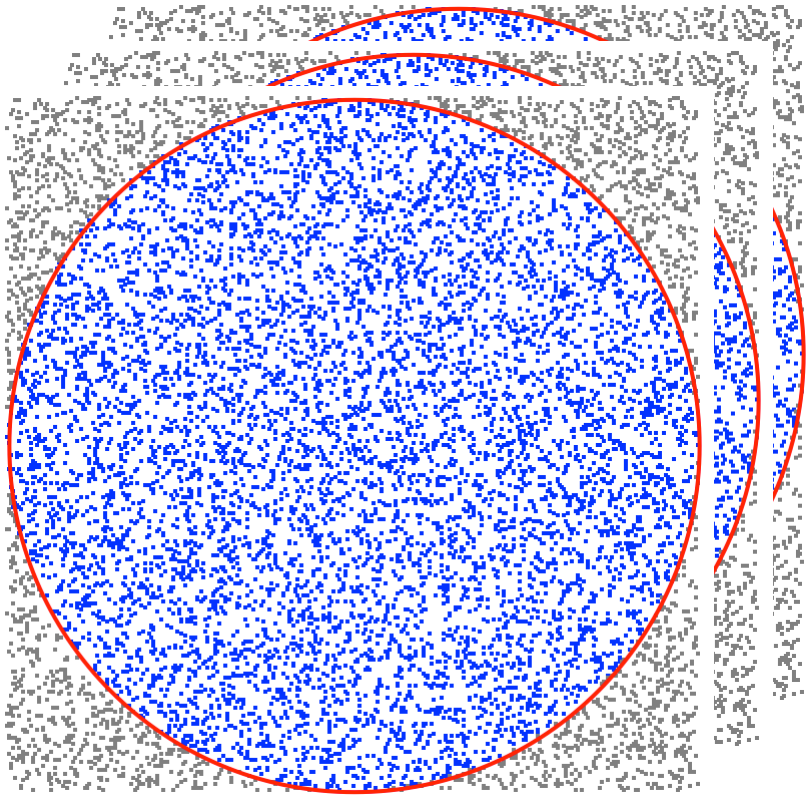




Obtain More Samples

- Let us test this by repeating our simulation 100 times:

Repeat multiple times

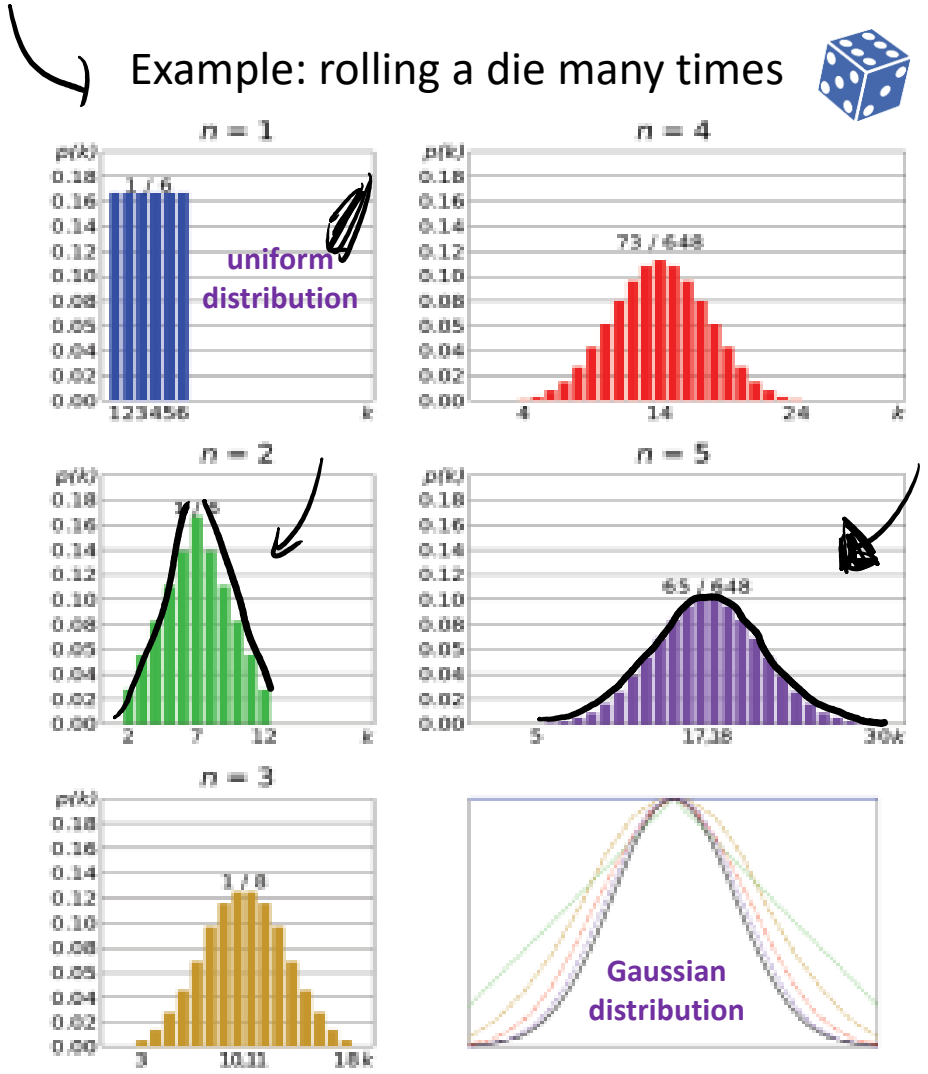


---------- Simulation Code ----------

```
def getEst(num_darts, num_trials):
    estimates = []
    for t in range(num_trials):
        guess = throwDarts(num_darts)
        estimates.append(guess)
    s_dev = np.std(np.array(estimates),ddof=1)
    s_err = s_dev/(len(estimates)**.5)
    cur_est = sum(estimates)/len(estimates)
    return (s_mean,s_err)
```

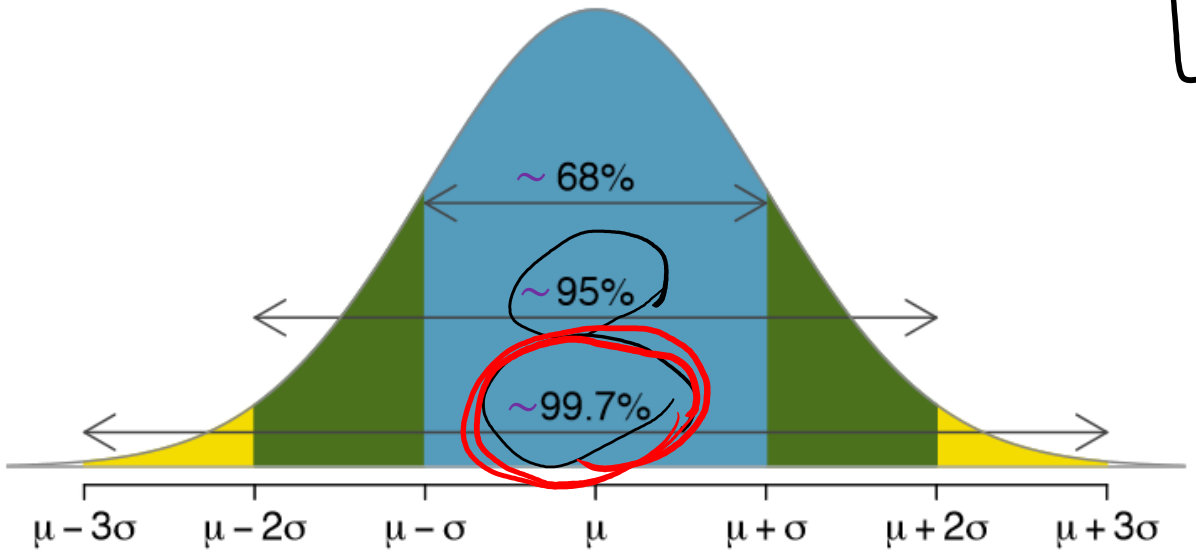
Central Limit Theory

- Q: What happens when you perform a measurement multiple times and compute the average.
- A: By the central limit theory the **distribution will become Gaussian**, and the mean will be close to the population (true) mean.

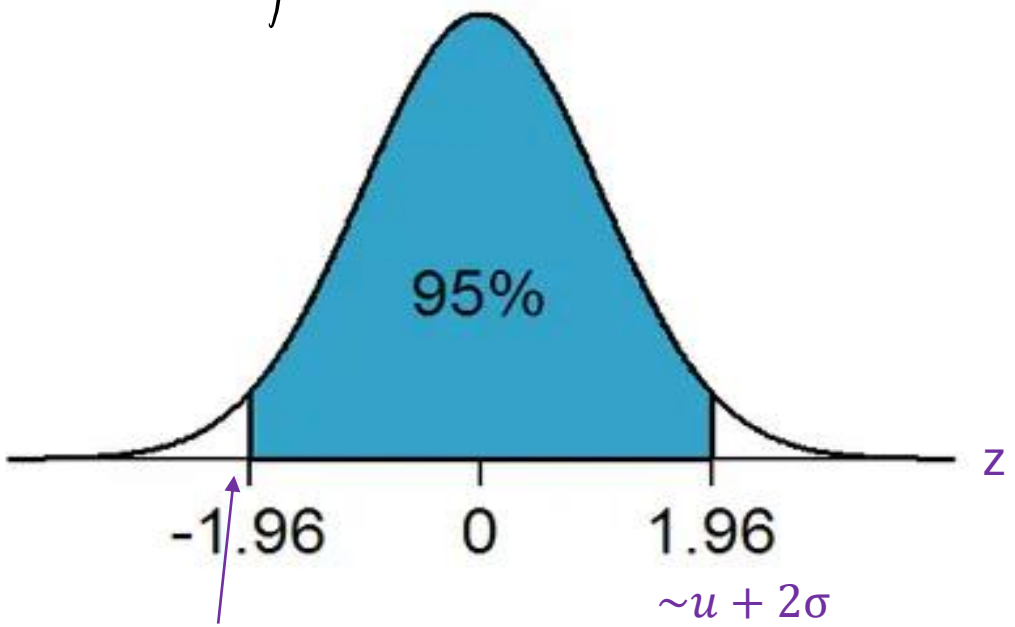


Gaussian Distribution

- What does a 95% confidence really mean?



Normalized Gaussian
($\mu = 0, \sigma = 1$)



confidence level value
(typically set based on a 95% confidence)

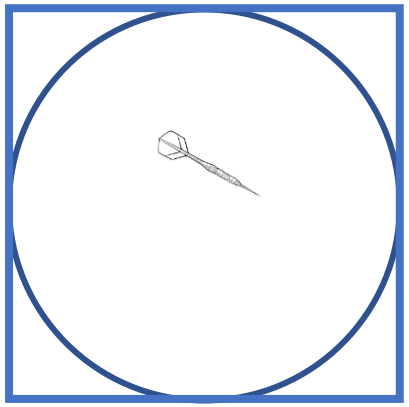
Confidence Intervals

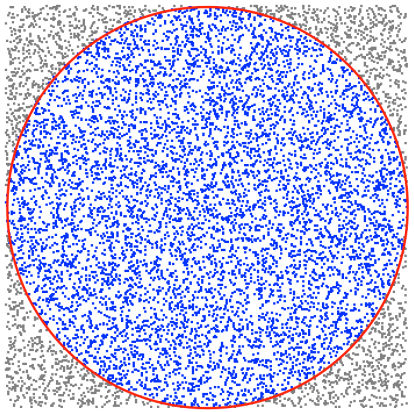
100 repetition

100 run

10 000 darts

- Calculating for our example…





one sample mean

```
Sample Mean:  3.141528
CI:  3.140608  to  3.142448
True Mean:  3.141592653589793
```

sample mean

$$\bar{x} = \frac{1}{n}\sum_i x_i$$

3.1409936

sample standard deviation

$$s^2 = \sum_i \frac{(x_i - \bar{x})^2}{n-1}$$

s = 0.005649

$$CI = \bar{x} \pm z\frac{s}{\sqrt{n}}$$

confidence level value (typically set at 95%)

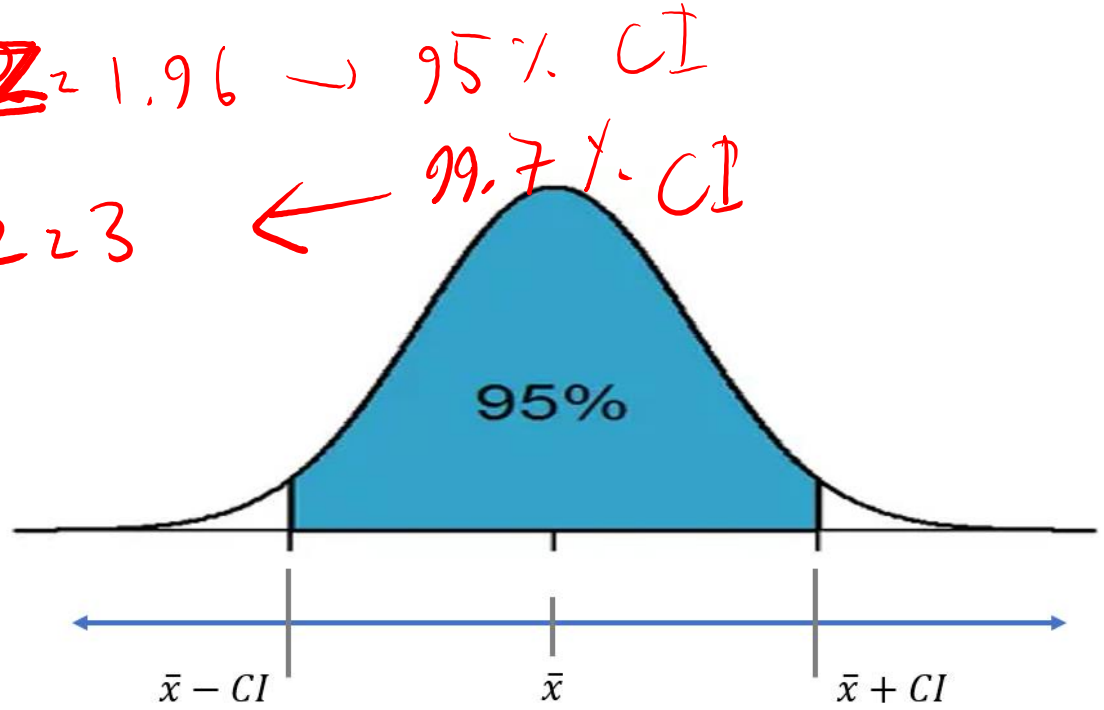
1.96

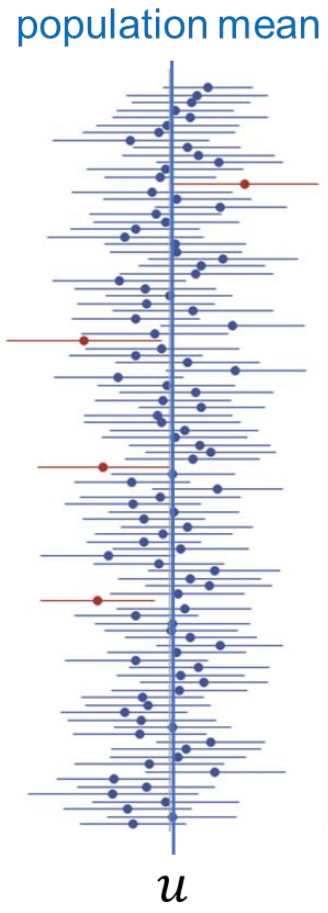
sample size

100

Confidence Intervals

- What does a 95% confidence really mean?



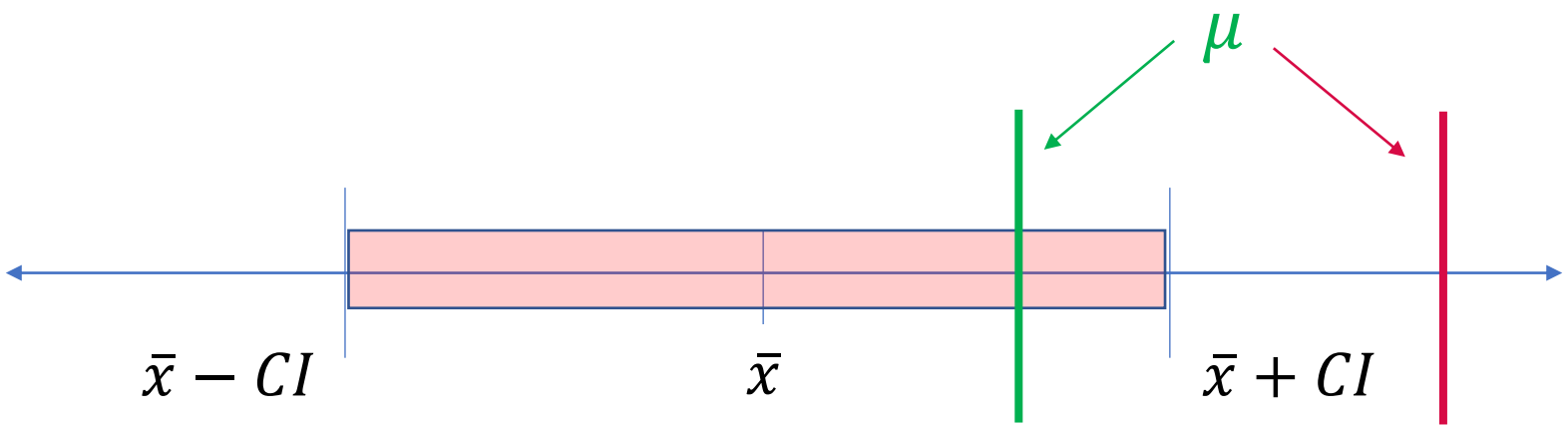


- If we were to repeat the simulation to obtain 100 sample means, then the confidence interval will contain the population mean ~95 times.
- And ~5 of the times, the interval will not contain the population mean.

Let's test it out in code!

Hypothesis Testing

- Q: How does confidence interval relate to hypothesis testing?



- A: Null hypothesis is that the sample and population mean come from the same distribution. If CI overlaps with population mean, then we do not reject the null hypothesis. Otherwise, we reject the null hypothesis.

Hypothesis Testing

- In the example we implanted what is referred to as a z-test. One of the **requirements of a z-test is that we know the population variance**.
  - or often if we have n > 30 samples we can assume that the sample variance is close enough.

- Alternatively, we can **apply a t-test** which uses a slightly different distribution that works well when population variance is unknown.

Short Break

Two or More samples

- We can also do a hypothesis test between two or more samples. That is, we can test if samples have the same population mean (Null Hypothesis):

$H_0: \mu_1 = \mu_2$

$H_a: \mu_1 \neq \mu_2$

two sample t-test or ANOVA

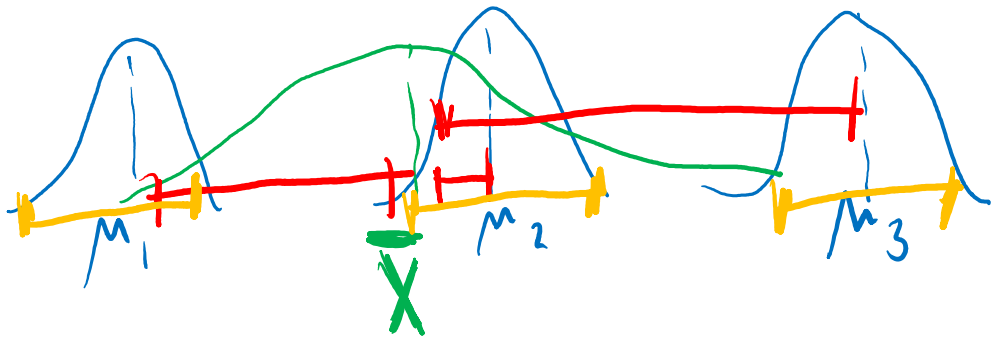
$H_0: \mu_1 = \mu_2 \ldots = \mu_m$

$H_a: \mu_1 \neq \mu_2 \ldots \neq \mu_m$

ANOVA

- If we reject the null hypothesis then the means are different.

ANOVA

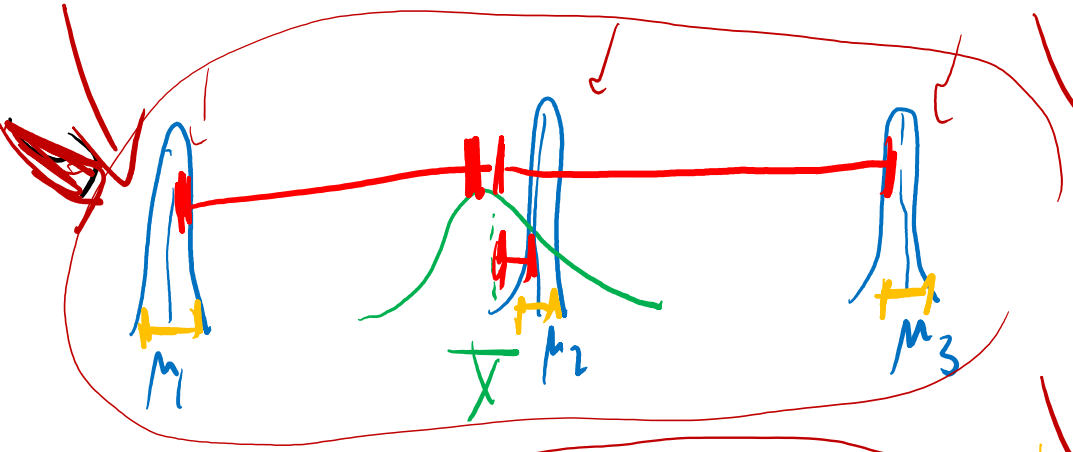


SS_{among}

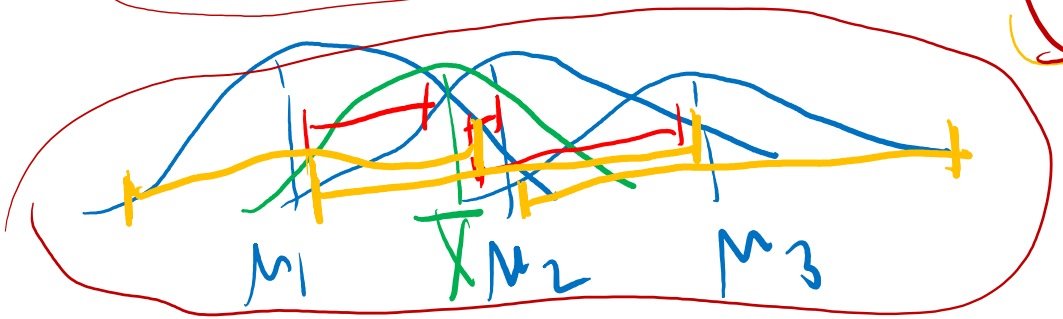
SS_{within}

H_0: $\mu_1 = \mu_2 \ldots = \mu_m$

H_a: $\mu_1 \neq \mu_2 \ldots \neq \mu_m$



Large → Reject



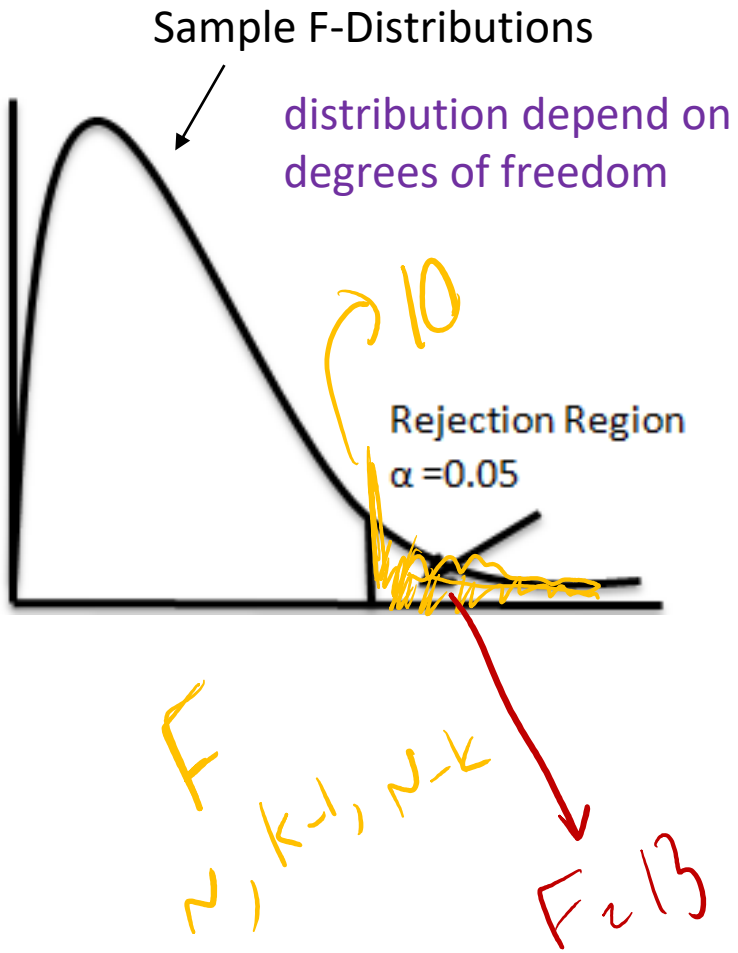
Small → Fail to Reject

ANOVA

- To perform a statistical test on whether **more than two population means are equal** we can use Analysis of Variance (ANOVA).

- The approach:
  1. Find sum of squared **differences within groups** div. by dof
  2. Find sum of square **differences among group** div. by dof
  3. Ratio provides F value which is compared to the F distribution based on desired critical value or significance.

$$F = \frac{\text{variance among groups}}{\text{variance within groups}} = \frac{SS_{among}/(k-1)}{SS_{within}/(N-k)}$$



Example 1: ANOVA

Given scores from a test for 9 students: {1,3,4,5,5,5,6,7,9}

$\bar{X} = 5$

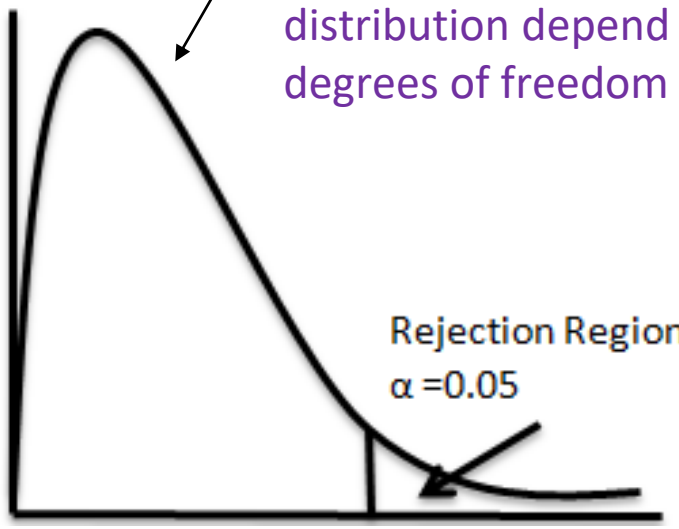
n = 9

k = 3

	Group I	Group II	Group III
	{1,5,9}	{3,5,7}	{4,5,6}
	$\overline{X_1} = 5$	$\overline{X_2} = 5$	$\overline{X_3} = 5$
SS_{within} =	$(-4)^2+0^2+4^2$	$(-2)^2+0^2+2^2$	$(-1)^2+0^2+1^2$
	32	8	2
SS_{among} =	$3(0^2)$	$3(0^2)$	$3(0^2)$
	0	0	0

Sample F-Distributions

distribution depend on degrees of freedom



$$F = \frac{\text{variance among groups}}{\text{variance within groups}} = \frac{0/(3-1)}{42/(9-3)} = 0$$

Example 2: ANOVA

Given scores from a test for 9 students: {1,3,4,5,5,5,6,7,9}

Group A	Group B	Group C
{1,3,5}	{5,7,9}	{4,5,6}
$\overline{X_1} = 3$	$\overline{X_2} = 7$	$\overline{X_3} = 5$

SS_{within} = $(-2)^2+0^2+2^2$ + $(-2)^2+0^2+2^2$ + $(-1)^2+0^2+1^2$

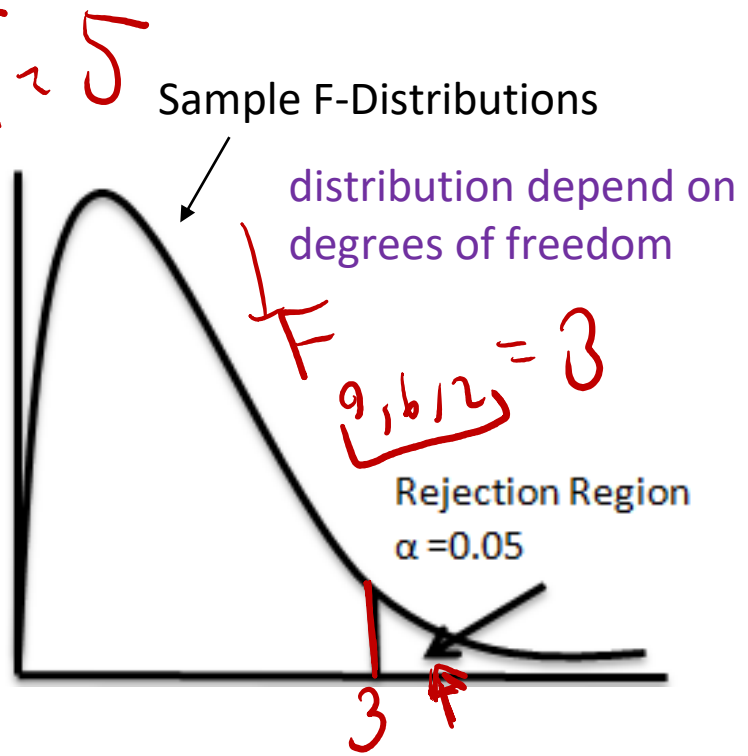
= 8 + 8 + 2

SS_{among} = $3(3-5)^2$ + $3(7-5)^2$ + $3(5-5)^2$

= 12 + 12 + 0

$$F = \frac{\text{variance among groups}}{\text{variance within groups}} = \frac{24/(3-1)}{18/(9-3)} = 4$$

$\bar{X} = 5$



Application: Feature Selection

- ANOVA can also be used for feature selection
- F-value score -> checks that when grouping the numerical feature by target vector, the means of each group is significantly different.

Application: Feature Selection

- ANOVA can also be used for feature selection
- F-value score -> checks that when grouping the numerical feature by target vector, the means of each group is significantly different.

- Examples of ANOVA for Feature Selection in Machine Learning:
  - https://chrisalbon.com/machine_learning/feature_selection/anova_f-value_for_feature_selection/
  - https://towardsdatascience.com/anova-for-feature-selection-in-machine-learning-d9305e228476

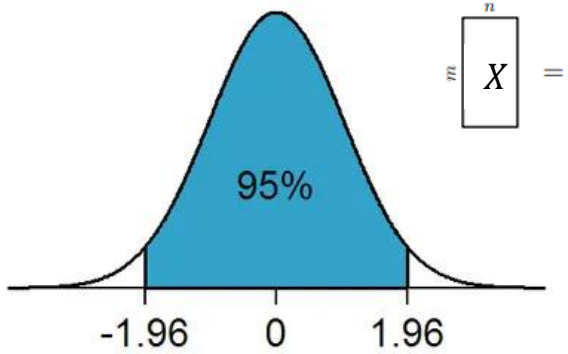
Programming Checklist

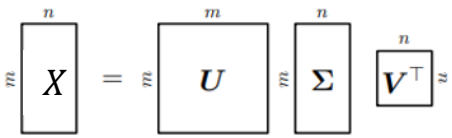
- Python Basics
  - int, float, bool, operations, display,
  - lists, tuples, dictionaries, sets and mutability
  - conditionals, loops, functions, list comprehension, files
  - object-oriented programming, class, methods
- Analysis of Algorithms and Asymptotic Notation
  - array search, sorting, hashing
- Common Data Science Packages
  - Jupyter notebook
  - pandas, numpy, scipy, matplotlib, scikit-learn
- Automatic Differentiation
- Github Basics

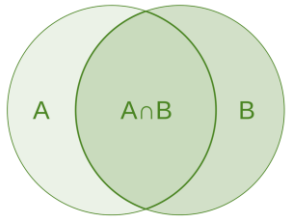
Mathematics Checklist

- Discrete Math
  - sets, union and intersection, combinations, permutations
- Probability Theory
  - summary statistics, expectation, covariance,
  - multivariate gaussians and basic distributions
  - confidence intervals, hypothesis testing, ANOVA
- Linear Algebra
  - change of variables, projections, data augmentation
  - determinant, eigenvalue decomposition, trace, rank
  - matrix inverse, PCA, SVD and dimensionality reduction
- Vector Calculus
  - Jacobians/gradients, matrix derivatives

$$B = \begin{bmatrix} -7 & -7 & 6 \\ 2 & 1 & -1 \\ 4 & 5 & -4 \end{bmatrix}$$





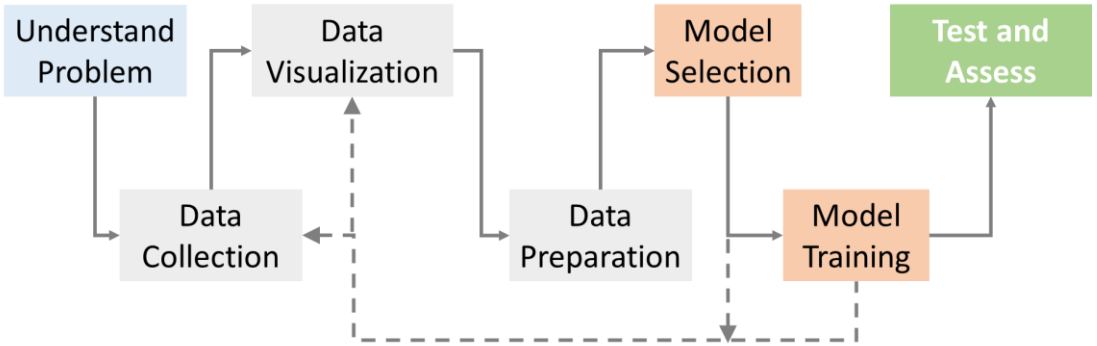


$$F_X(x) = \int_{-\infty}^{x} f_X(t)\,dt.$$

CDF PDF

Machine Learning and Analytics Checklist

- Instance Based Learning
  - KNN, decision trees, clustering strategies
- Model Based Learning
  - linear/logistic regression, MLE approach
  - feature mapping and polynomial regression
  - mixture of Gaussians for anomaly detection
  - deep learning overview and PyTorch
- Machine Learning Taxonomy
  - parametric/nonparametric,
  - classification/ regression,
  - supervised/unsupervised,
  - Model- and instance-based learning
- Discrete and Continuous Optimization
  - gradient descent, regularization and convexity
  - greedy algorithms and set cover
- Performance Metrics
  - generalization, overfitting, bias/variance
  - precision and recall, Confusion Matrix, ROC
  - cross-validation



The End

or Just the Beginning…

- The MEng in MIE with Emphasis in Analytics builds on the foundations covered in APS1070.
- Courses to consider:
  - MIE1626 – Data Science Methods and Quantitative Analysis
  - MIE1517 – Introduction to Deep Learning
  - MIE1624 – Introduction to Data Science and Analytics
  - ECE1513 – Introduction to Machine Learning
  - MIE1628 – Big Data Science
  - APS1080 – Introduction to Reinforcement Learning
  - and many more…

Final Assessment Details

Final Assessment Details

- **When**
  - Crowdmark Opens: April 12 at 9am (Toronto Time)
  - Crowdmark Closes: April 13 at 3pm (Toronto Time)– latest time to start is April 13 at 11:30am

  Late submission: A grade of zero for the final assessment (as per syllabus)

  - Crowdmark – invites will be sent by email.
  - Submit answer to each question separately with legible handwriting.

- **Duration**
  - Limited window to start the exam and submit it (a countdown starts as soon as you access the assessment)
  - 2.5 hour for the exam + 30 minutes for taking pictures and uploading on crowdmark + 30 minutes for all contingencies = 3.5-hour in total

Final Assessment Details

- Material in the final exam:
  - Weeks 1 to 12 of lectures
  - Use course material + online resources with citation – NO HELP FROM OTHERS!!
- Sign Honour Code
  - **All work must be your own; all work must be completed without consulting with anyone else.**
  - Students suspected of plagiarism on a project, midterm or final assessment will be referred to the department for formal discipline for breaches of the Student Code of Conduct.
- Piazza will not be accessible during the exam.
- If needed, make assumptions and answer the questions. No need to contact us.
- In case of a logistic problem, you should email Samin + Sinisa + Ali immediately. There is no guarantee that we can respond to you before your time runs out.

Final Assessment Topics

- Analytical concepts in learning
- Comparison of ML models
- Dimensionality reduction methods
- Matrix operations and transformations
- Eigenvectors and eigenvalues
- Gaussian Elimination
- Solving systems of linear equations
- Linear and non-linear regression
- Decision trees
- Artificial neural networks
- Backpropagation
- Vector calculus
- Analytical geometry
- Model-based learning
- Gradient descent
- Polynomial regression
- Regularization
- Numerical optimization
- Logistic regression
- Binary classification
- Abstract data types
- Performance measures
- Confusion matrix
- Receiver Operating Characteristic
- Design and implementation of ML models

Final Assessment Format

- Multiple Choice and True /False
- Short Answer
- Calculations
- Descriptive and Analytical Answer
- Coding
  - You may be asked to interpret and explain concepts and models similar to what you would have seen in lectures.
  - **Google Colab** is available to you should you find it useful, but it does not count as showing your working.

Questions?

- Piazza

Sample Questions

Fall 2019 – Part I

Part I: Multiple Choice (Select only 1 answer per question)

1. Which of the following statements is true? **[2]**

   a) In *k*-fold cross-validation, training data is divided into *k folds*, then the model is trained on *k* folds and validated on 1 fold, for a total of *k-1* times (splits).
   b) In *k*-fold cross-validation, training data is divided into *k folds*, then the model is trained on *k* folds and validated on 1 fold, for a total of *k* times (splits).
   c) In *k*-fold cross-validation, training data is divided into *k folds*, then the model is trained on *k*-1 folds and validated on 1 fold, for a total of *k* times (splits).
   d) In *k*-fold cross-validation, training data is divided into *k folds*, then the model is trained on *k*-1 folds and validated on 1 fold, for a total of *k-1* times (splits).

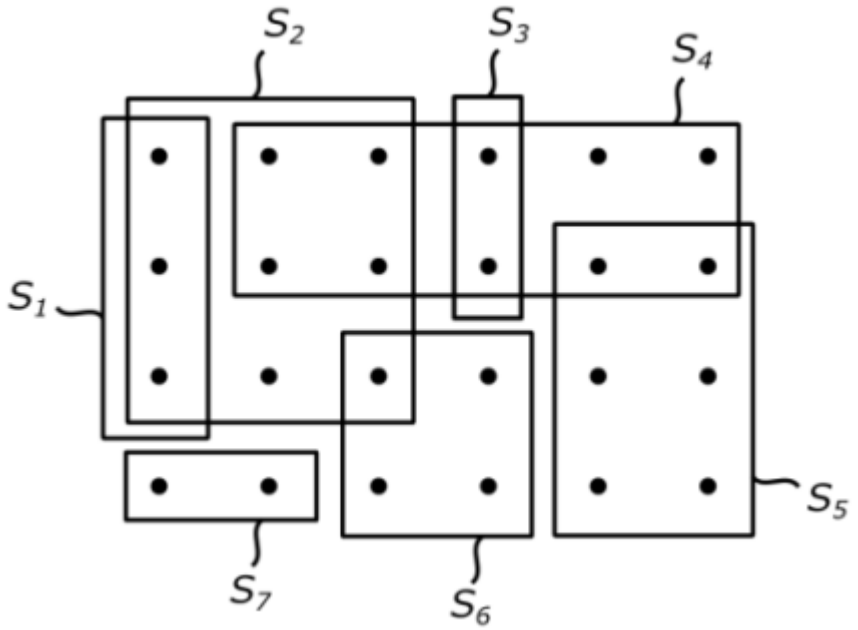
Fall 2019 – Part I

2. When performing PCA, the goal is to accomplish which of the following? **[2]**

   a) Maximize the variance of the primary components and maximize the residuals.
   b) Minimize the variance of the primary components and minimize the residuals.
   c) Maximize the variance of the primary components and minimize the residuals.
   d) Minimize the variance of the primary components and maximize the residuals.

Fall 2019 – Part II

3. The schematic to the right has sets S_1, S_2, S_3, S_4, S_5, S_6 and S_7. What sets, and in what order, would a greedy algorithm select to cover the universe (i.e., cover each point) if all sets are weighted equally? **[2]**



Answer

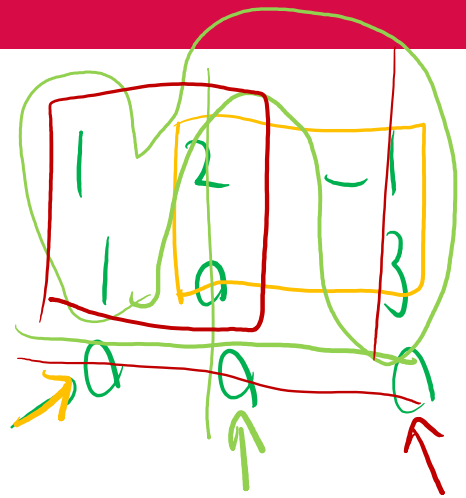
In order of selection: S4, S2, S5, S6, S7

We exclude S1, S3

Fall 2019 – Part II

4. Is the matrix $A = \begin{pmatrix} 1 & 2 & -1 \\ 1 & 0 & 3 \\ -2 & -4 & 2 \end{pmatrix}$ invertible? **[2]**

$2R_1 + R_3$



Answer

For $n = 3$ (known as Sarrus' rule),

$$\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix} = a_{11}a_{22}a_{33} + a_{21}a_{32}a_{13} + a_{31}a_{12}a_{23}$$

$$- a_{31}a_{22}a_{13} - a_{11}a_{32}a_{23} - a_{21}a_{12}a_{33}\,.$$

=1*0*2+1*(-4)*(-1)+(-2)*2*3-(-2)*0*(-1)-1*(-4)*3-1*2*2=0+4-12+0+12-4=0

No -> determinant is 0

Fall 2019 – Part II

5. Given that $\binom{n}{r} = \frac{n!}{r!(n-r)!}$, prove that $\binom{n}{r} = \binom{n}{n-r}$ **[2]**

$\binom{a}{b} = \frac{a!}{b!(a-b)!}$

Answer

$$= \frac{n!}{(n-r)!\,(n-(n-r)!}$$

$$= \frac{n!}{(n-r)!\,r!}$$

$n-n+r$

Fall 2019 – Part II

6. We have $x = [1,1,2]^T \in \mathbb{R}^3$ and $y = [1,0,3]^T \in \mathbb{R}^3$. What is the angle between vectors? **[2]**

$x \cdot y = 1 + 0 + 6 = 7$

$x \cdot y = |x||y| \cos(\theta)$

$|x| = \sqrt{6}$ $|y| = \sqrt{10}$

$\cos(\theta) = \frac{7}{\sqrt{6}\sqrt{10}} \rightsquigarrow \theta = \cos^{-1}\left(\frac{7}{\sqrt{60}}\right)$

Answer

$$\cos(w) = \frac{x^T y}{\sqrt{x^T x y^T y}} = \frac{7}{\sqrt{6*10}} \rightarrow w = 25.4\ degrees$$

Fall 2019 – Part II

7. Calculate the Jacobian $J_F(x_1, x_2, x_3)$ of the function $F : \mathbb{R}^3 \to \mathbb{R}^4$, which has components: **[2]**

$$y_1 = x_1 + 2x_3^2 \; ; \; y_2 = x_1 \sin x_2 \; ; \; y_3 = x_2 \exp(x_3) \; ; \; y_4 = x_1 + x_2$$

Answer

$$\begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_1}{\partial x_2} & \frac{\partial y_1}{\partial x_3} \\ \frac{\partial y_2}{\partial x_1} & \frac{\partial y_2}{\partial x_2} & \frac{\partial y_2}{\partial x_3} \\ \frac{\partial y_3}{\partial x_1} & \frac{\partial y_3}{\partial x_2} & \frac{\partial y_3}{\partial x_3} \\ \frac{\partial y_4}{\partial x_1} & \frac{\partial y_4}{\partial x_2} & \frac{\partial y_4}{\partial x_3} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 4x_3 \\ \sin(x_2) & x_1\cos(x_2) & 0 \\ 0 & \exp(x_3) & x_2\exp(x_3) \\ 1 & 1 & 0 \end{bmatrix}$$

4x3

Fall 2019 – Part II

8. Consider functions $f : \mathbb{R}^3 \to \mathbb{R}^1$ and $\boldsymbol{x} : \mathbb{R}^1 \to \mathbb{R}^3$, where:

$$f(\boldsymbol{x}) = x_1 + 2x_2^2 + x_3 \; ; \boldsymbol{x}(t) = \begin{bmatrix} t \\ 3t \\ \exp(t) \end{bmatrix}$$

Calculate the gradient $\frac{df}{dt}$ using the chain rule. **[2]**

Answer

$$\begin{bmatrix} \frac{df}{dx_1} & \frac{df}{dx_2} & \frac{df}{dx_3} \end{bmatrix} \begin{bmatrix} \frac{dx_1}{dt} \\ \frac{dx_2}{dt} \\ \frac{dx_3}{dt} \end{bmatrix} = \begin{bmatrix} 1 & 12t & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 3 \\ \exp(t) \end{bmatrix} = 1 + 36t + \exp(t)$$

Fall 2019 – Part III

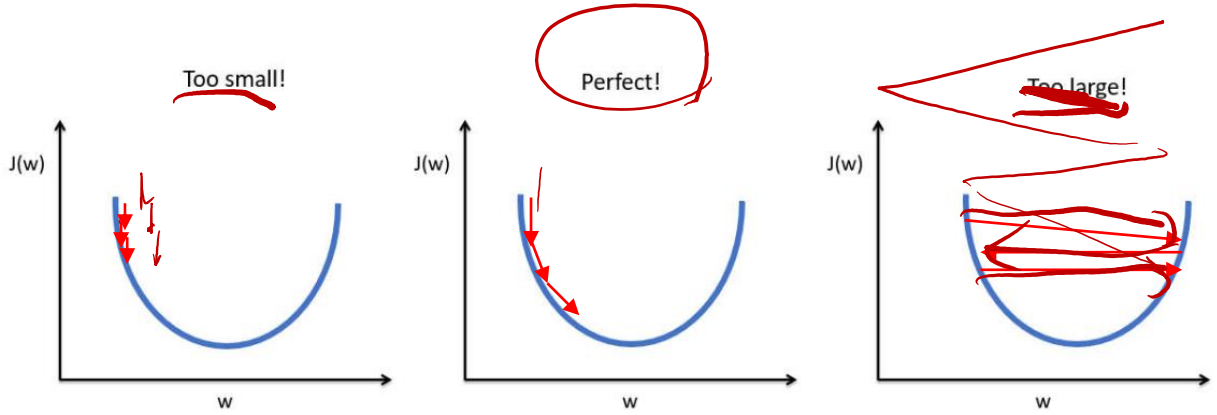
9. In general, is using a gradient descent algorithm a good choice for finding optimal hyperparameters? Why or why not? **[2]**

Answer

No. Gradient descent requires a smooth function to optimize, and ideally a convex function in order to find the global minimum. This is usually not the case for hyperparameters.

10. What is a learning rate? Explain what happens if it is set too high or too low (you may use a schematic/drawing if helpful). **[2]**

Answer



Learning rate is the step size taken in the opposite direction of the gradient.

Fall 2019 – Part IV

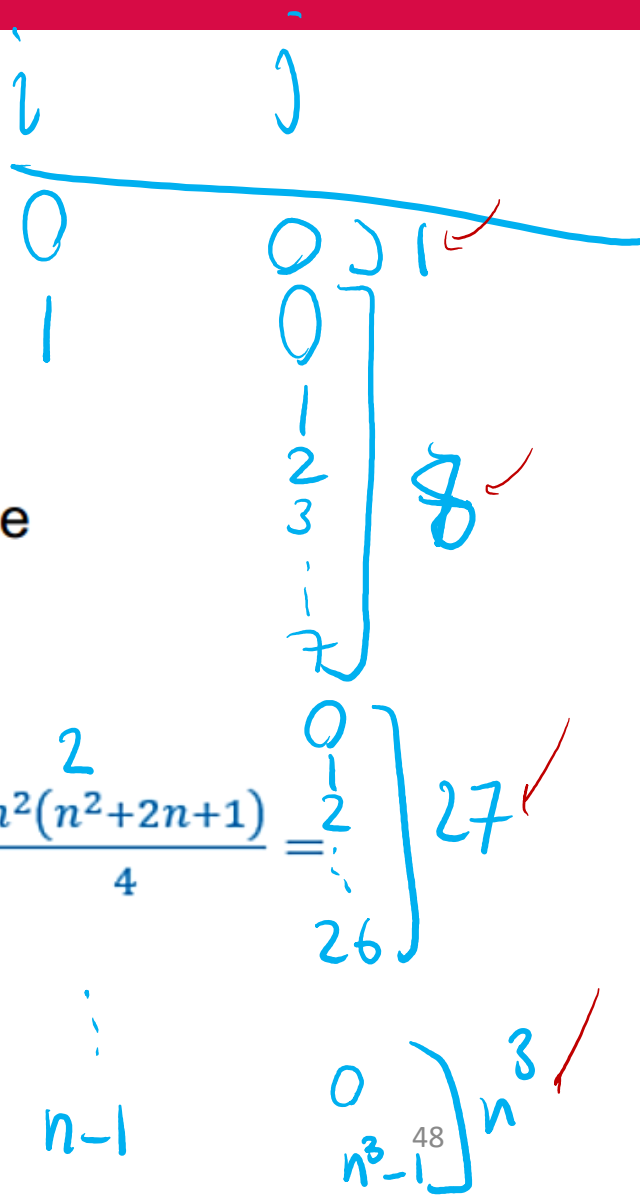
11. You have the following code:

```
for i in range (n):
    for j in range ((i+1)**3):
        x = x + 1
        y = y + 1
```

How efficient is this code in terms of big O notation (what is the order of the algorithm's runtime)? [hint: $\sum_{k=1}^{n} k^3 = \frac{n^2(n+1)^2}{4}$] **[2]**

Answer

The loop is performed 1, 8, 27, … , n^3 times, $\sum_{k=1}^{n} k^3 = \frac{n^2(n+1)^2}{4} = \frac{n^2(n^2+2n+1)}{4} = \frac{n^4+2n^3+n^2}{4} \rightarrow O(n^4)$



Fall 2019 – Part IV

12. The input parameter *x* of "my_function" is training data with features as columns and examples as rows.

```
def my_function(x):
    return (x - x.mean()) / x.std
```

a) What is the purpose of this function, specifically? **[2]**

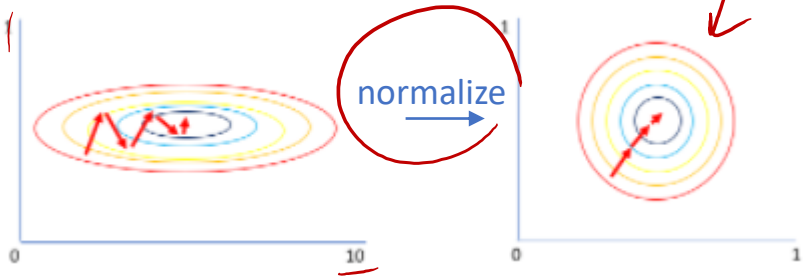
b) Describe why this type of process is important to gradient descent and K-Nearest Neighbor algorithms. **[2]**

Answer

a. Normalize each dimension by zero mean and unit variance.

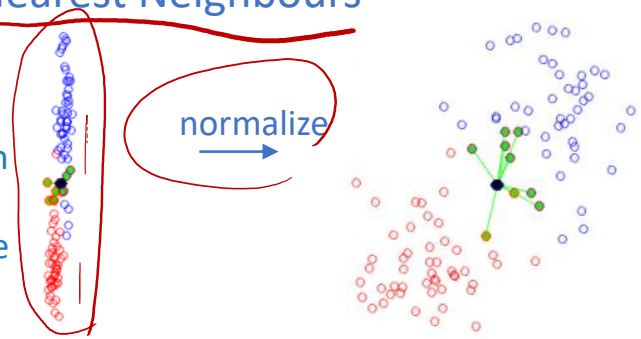
b. Gradient descent

gradient does not point towards minimum, hence takes longer to converge



K-Nearest Neighbours

All nearest neighbours align in direction of the axis with the smaller range.



Fall 2019 – Part IV

13. Consider a differentiable loss function $l(\boldsymbol{w}, \boldsymbol{x}, y)$ and a dataset D. You're asked to optimize the average loss $\frac{1}{N}\sum_{i=1}^{N} l\left(\boldsymbol{w}, \boldsymbol{x}^{(i)}, y^{(i)}\right)$ with respect to $\boldsymbol{w}$. Write pseudo-code implementing mini-batch gradient descent optimization using a decaying learning rate (i.e., a learning rate that is reduced by α_D every epoch), with the following inputs: **[4]**

- Differentiable loss $l(\boldsymbol{w}, \boldsymbol{x}, y)$ with gradient $\nabla_{\boldsymbol{w}} l(\boldsymbol{w}, \boldsymbol{x}, y)$.
- Dataset $D = \left(\boldsymbol{x}^{(1)}, y^{(1)}\right), \ldots \left(\boldsymbol{x}^{(N)}, y^{(N)}\right)$.
- Mini-batch size m, initial weights $\boldsymbol{w}_0$ and number of steps T.
- Initial learning rate α_0 and learning rate decay α_D.

Note that your pseudo-code is just a representation of your algorithm written in plain English. Keep your pseudo-code simple and concise. Please use indentation and control structures. State any assumptions you make.

Fall 2019 – Part IV

Answer

```
Initial weight w = w_0
Initial alpha = a_0
For each EPOCH
        shuffle the training data
        For each Iteration (T)
                pick a mini batch
                compute the gradients: delta (w,x,y)
                update the weights: w = w - alpha * delta
        Update the LR: alpha = alpha * a_d
```

$\alpha_0 = 0.01$

$\alpha_d = 0.99$

$\alpha' = 0.01 \times 0.99$

$= 0.0099$

Short Break

Summer 2020

2. [2] We have that

$$f(\boldsymbol{x}) = f(x_1, x_2, x_3) = ln(x_1 x_2) - \frac{x_3}{2}$$

$$\boldsymbol{x}(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{bmatrix} = \begin{bmatrix} e^t \\ -cos(t) \\ 2t \end{bmatrix}$$

Calculate the gradient $\frac{df}{dt}$. Show all your calculations.

Answer

Given that $ln(x_1 x_2) = ln(x_1) + ln(x_2)$

$$\frac{df}{dt} = \frac{df}{dx}\frac{dx}{dt} = \begin{bmatrix} \frac{df}{dx_1} & \frac{df}{dx_2} & \frac{df}{dx_3} \end{bmatrix} \begin{bmatrix} \frac{dx_1}{dt} \\ \frac{dx_2}{dt} \\ \frac{dx_3}{dt} \end{bmatrix}$$

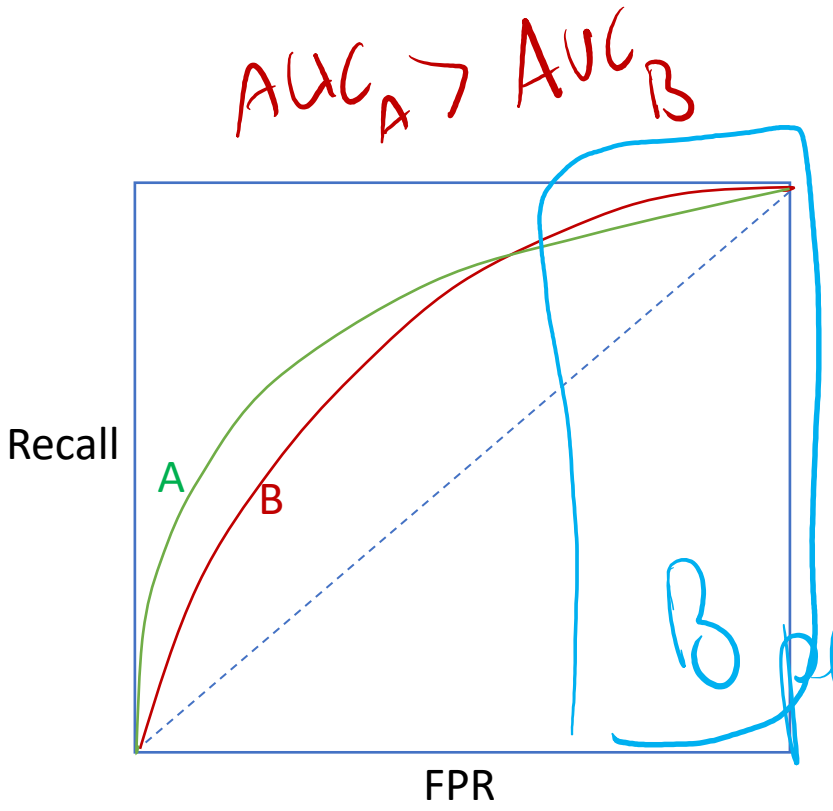
$$= \begin{bmatrix} \frac{1}{x_1} & \frac{1}{x_2} & -\frac{1}{2} \end{bmatrix} \begin{bmatrix} e^t \\ sin(t) \\ 2 \end{bmatrix}$$

$$= \begin{bmatrix} \frac{1}{e^t} & -\frac{1}{cos(t)} & -\frac{1}{2} \end{bmatrix} \begin{bmatrix} e^t \\ sin(t) \\ 2 \end{bmatrix} = 1 - \tan(t) - 1$$

$$= -tan(t)$$

Summer 2020

3. [3] Binary classification Model A achieves a top F1 score of $F1_A$ and an area under the ROC curve (AUC) of AUC_A, whereas binary classification Model B achieves a top F1 score of $F1_B$ and an AUC of AUC_B, on the same dataset. If $AUC_A > AUC_B$., is it implied that $F1_A > F1_B$? Why?

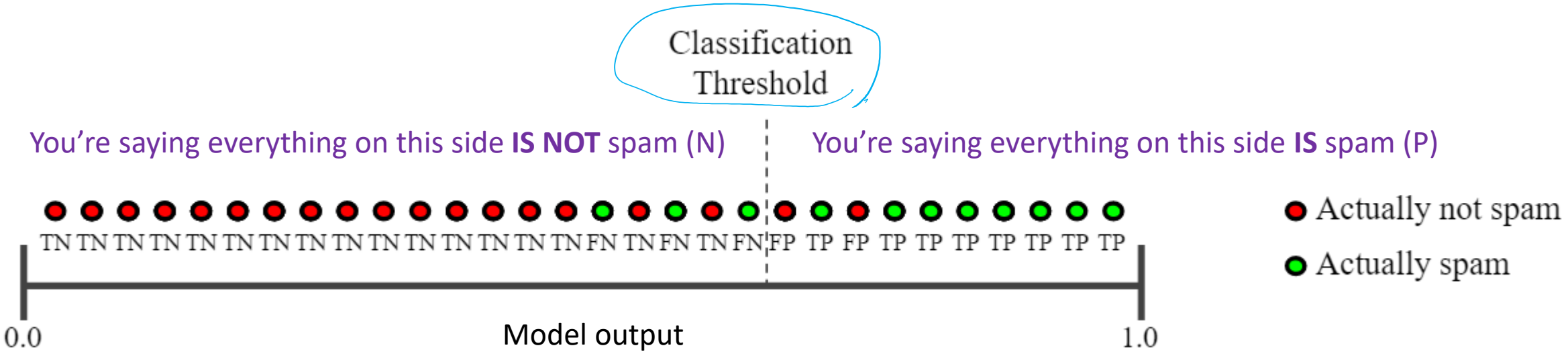


Answer

No, $F1_A > F1_B$ is not implied. AUC is the area under the ROC curve, which plots TPR ($TPR = \frac{TP}{TP+FN}$) vs FPR ($FPR = \frac{FP}{FP+TN}$). F1 score is ($\frac{2TP}{2TP+FP+FN}$) calculated at a single threshold. ***Top F1 score is sensitive to the "best" threshold, whereas AUC is an aggregate measure across all possible thresholds .***

A model has multiple F1's depending on the threshold

- Application – spam detection
- Improving precision usually reduces recall, vice-versa



Precision tp/(tp+fp) = **8/(8+2) = 0.8**

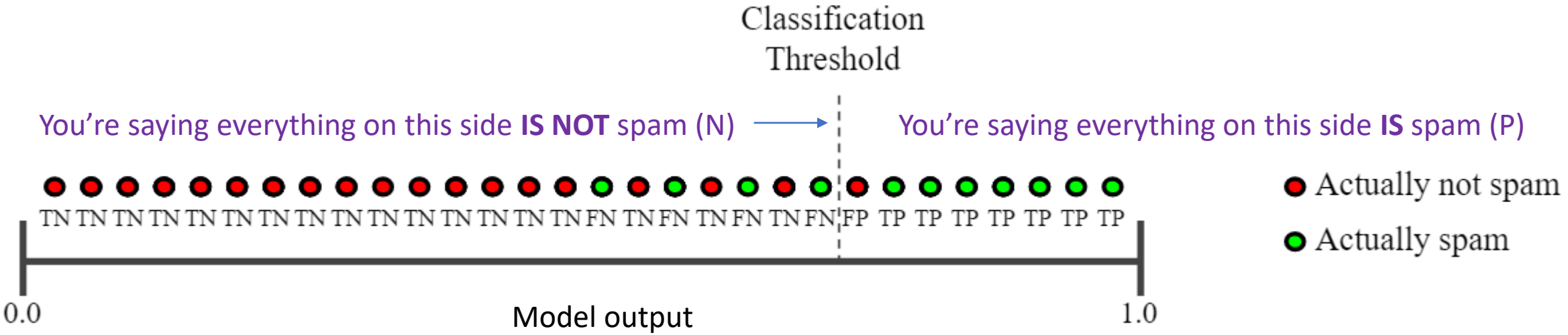
Recall tp/(tp+fn) = **8/(8+3) = 0.73**

F1 Score = **2*0.8*0.73/(0.8+0.73) = 0.76**

Source: ML Crash Course

A model has multiple F1's depending on the threshold

- Application – spam detection
- Improving precision usually reduces recall, vice-versa



Precision **= 7/(7+1) = 0.88**

Recall **= 7/(7+4) = 0.64**

F1 Score **=2*0.88*0.64/(0.88+0.64)=0.74**

Source: ML Crash Course

Summer 2020

4. [3] *StandardScaler* (in *sklearn*) is often used in linear regression models.
   a. What does *StandardScaler* do?
   b. Is it best practice to apply StandardScaler before or after splitting data into training and testing groups? Why?
   c. Typically, is it useful to apply StandardScaler on targets, feature data, or both?

Answer

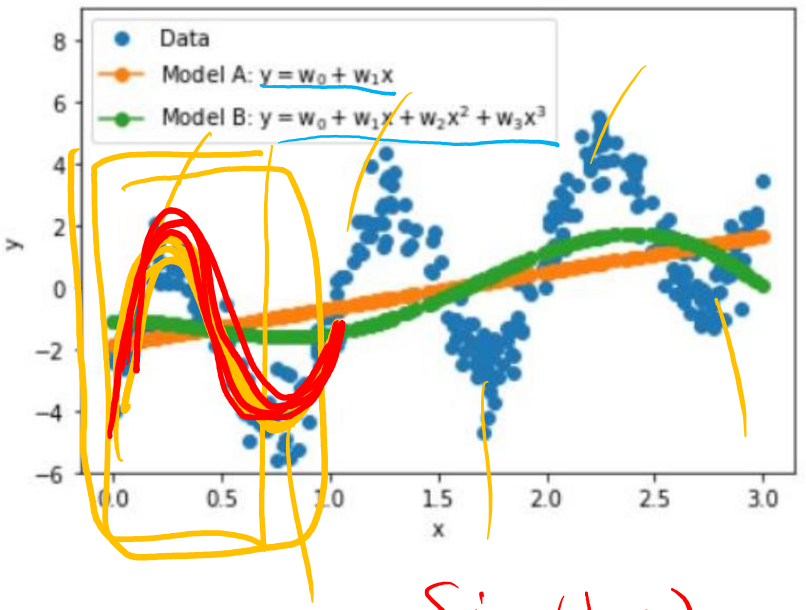
a. StandardScaler scales features to 0 mean and unit variance.

b. It's best to apply StandardScaler after splitting data, since you want to keep test data completely separate. If you scale before splitting, you're incorporating knowledge of test data properties into your model.

c. StandardScaler is typically applied only on features (so any one feature does not dominate the others). There's usually no benefit to applying it on targets.

Summer 2020

5. [3] Sam applies linear regression to data that has a single feature, *x*. A first model, Model A, makes predictions *y* using only a bias (w_0) and a weighted feature (w_1x), i.e. $y = w_0 + w_1x$. Unsatisfied, Sam then creates new features that are the *x* feature squared and cubed, yielding the model $y = w_0 + w_1x + w_2x_2^2 + w_3x_3^3$ (Model B), which still isn't a great fit.



a. What minimum degree polynomial is needed to fit the data reasonably well? Why? [hint: count turning points (extremums) of polynomial]

b. Can you conceive of an alternate feature mapping scheme, where a single additional feature is added to Model A, that would fit the data well? Explain your reasoning.

a sin(bx)

Answer

a. You'll need a polynomial of at least degree 7 to fit the data well – there's an explanation here – 1 more degree than the number of bumps.

b. The idea here is to add a sinusoidal feature to Model A. The general equation for a sine function is:

$$y = a * sin(b * x)$$

Here the period is clearly 1, so $1 = \frac{2\pi}{b}$ and $b = 2\pi$

We could thus add a feature sin(2πx), and expect a good fit.

Summer 2020

6. [3] This term, several approaches to combat overfitting were discussed.

   a. What is overfitting, and how do we know if a model is overfitted?
   b. Describe two methods that are used to prevent overfitting. To what models do these methods apply? (ie are they general, or specific to certain types of models?)
   c. What are the major advantages and disadvantages of these methods?

Answer

a. Model has fit the subtleties in the training data that does not generalize to the test data. The model has low training error, but on a validation set, its error is high.

b. Cross-validation and regularization are the most obvious methods here, but you could also discuss the removal of features, adding more data, etc. These generalize to all models.

c. Cross-validation: allows you to use your data more effectively, but at the price of increased computation
Regularization: limits the flexibility of the parameters which can be good and bad. Good in that it will limit the model's propensity to overfit, but bad in that sometimes it may be too restrictive.

APS1070 course evaluations

Please consider completing the course evaluation (4-11 April).