# CS5228: Knowledge Discovery and Data Mining

Lecture 5 — Classification & Regression I

# Course Logistics — Update

# Recap — Association Rules

- ## Association Rule Mining
  - Input: database of transactions (i.e., sets of items)
  - Output: rules predicting the occurrence of some items based on occurrence of other items

- ## Example: Market Basket Analysis
  - Item = product in supermarket
  - Transaction = products in basket at check-out
  - Interesting rules = customers who by X als tend to Y

| TID | Items |
|-----|-------|
| 1 | bread, yogurt |
| 2 | bread, milk, cereal, eggs |
| 3 | yogurt, milk, cereal, cheese |
| 4 | bread, yogurt, milk, cereal |
| 5 | bread, yogurt, milk, cheese |

# Recap — Association Rules

- **Interesting** association rules
  - Different definitions that quantify usefulness of a rule: support, confidence, lift, conviction, leverage, etc.

- Important observation
  - Relationship between support and confidence

  > ➜ Decoupling Support and Confidence — 2 steps:
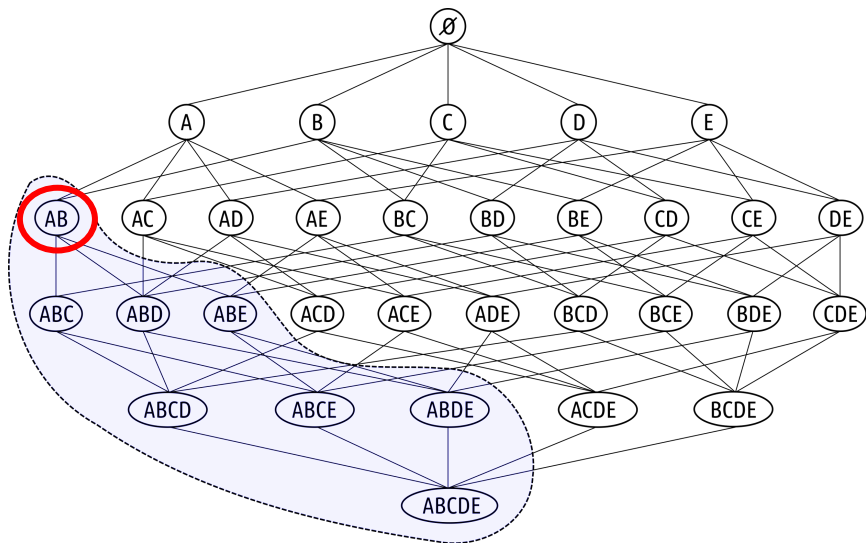  > 1) Frequent Itemset Generation
  > 2) Association Rule Generation

  $$C(X \rightarrow Y) = \frac{S(X \rightarrow Y)}{S(X)} = \frac{S(X \cup Y)}{S(X)}$$

  - **Anti-monotone** property of support and confidence

  > ➜ Apriori Algorithms for Frequent Itemset and Association Rule Generation
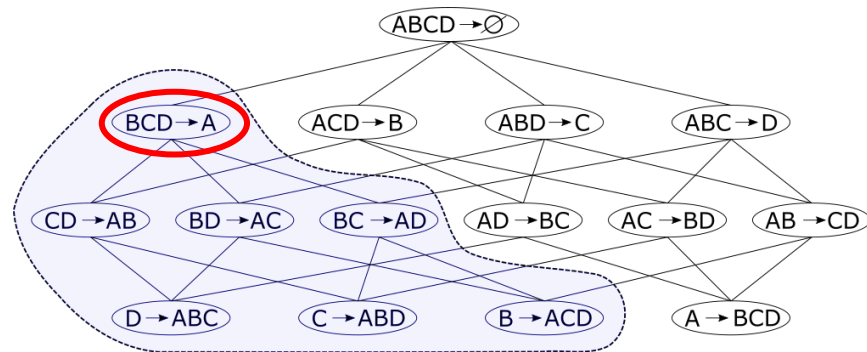
# Recap — Association Rules

## Itemset Lattice



If an itemset (e.g., {A,B}) is not frequent,
then any superset (e.g., {A,B,D}, {A,B,C,D})
is also not frequent

## Rule Lattice



If rule BCD➔A has a insufficient confidence,
so do all rules containing A in the consequent

# Outline

- **Classification & Regression**
  - **Overview & Examples**
  - Basic Setup
  - Evaluation

- Nearest Neighbor Methods
  - KNN — K-Nearest Neighbor Algorithm
  - Pros, Cons & Caveats

# Classification & Regression: Overview

- ## Classification **AND** Regression
  - Core tasks of supervised machine learning
  - Training: Find patterns in dataset between the values of **dependent variable(s)** given the values **independent variable(s) / features**
  - Prediction: Use patterns to assign values to dependent variables new/unseen data

- ## Classification **VS.** Regression
  - Classification: Dependent variable is categorical
  - Regression: Dependent variable is continuous

**Independent variables (features)**

| Age | Edu-cation | Marital Status | Income Level | Credit Approval | Credit Limit |
|-----|-----------|----------------|--------------|-----------------|--------------|
| 23  | Masters   | Single         | Mid          | No              | —            |
| 35  | College   | Married        | High         | Yes             | $S7,000      |
| 26  | Masters   | Single         | High         | No              | —            |
| 41  | PhD       | Single         | Mid          | Yes             | $S5,000      |
| 18  | Poly      | Single         | Low          | No              | —            |
| 55  | Poly      | Married        | High         | Yes             | $S10,000     |
| 30  | College   | Single         | High         | Yes             | $S8,000      |
| 35  | PhD       | Married        | High         | Yes             | $S10,000     |
| 28  | Masters   | Married        | Mid          | Yes             | $S5,000      |
| 45  | Masters   | Married        | Mid          | ???             | ???          |

**Dependent, categorical variable**

**Dependent, numerical variable**

# Application Examples

| month | flat_type | block | street_name | floor_area_sqm | flat_model | lease_commence_date | latitude | longitude | subzone | planning_area | resale_price |
|-------|-----------|-------|-------------|----------------|------------|---------------------|----------|-----------|---------|---------------|--------------|
| 2013-02 | 4 room | 4B | boon tiong road | 91.0 | model a | 2005 | 1.286589 | 103.831950 | tiong bahru | bukit merah | 663888.0 |
| 1997-08 | 3 room | 11 | holland drive | 65.0 | improved | 1975 | 1.309054 | 103.794063 | holland drive | queenstown | 228000.0 |
| 2007-08 | executive | 558 | jurong west street 42 | 157.0 | maisonette | 1985 | 1.354198 | 103.717344 | hong kah | jurong west | 305000.0 |
| 1996-02 | 4 room | 354 | hougang avenue 7 | 105.0 | new generation | 1986 | 1.372408 | 103.899433 | kangkar | hougang | 235000.0 |
| 2015-06 | 3 room | 99 | old airport road | 56.0 | standard | 1969 | 1.308590 | 103.888245 | aljunied | geylang | 320000.0 |

- Prediction of flat prices (regression task)
  - Help sellers and buyers to make better decisions
  - Understand the importance of attributes on flat prices

# Application Examples

| ID | age | sex | chest | rest_bp | chol | rest_ecg | max_hr | oldpeak | slope | num_vessels | thal | class |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 49.2 | 0 | 4.00 | 163.00 | 181.11 | 0 | 148.23 | 0.94 | 2 | 0 | 3 | 1 |
| 1 | 53.6 | 1 | 1.74 | 130.23 | 276.47 | 2 | 152.92 | 0.12 | 2 | 0 | 3 | 0 |
| 2 | 49.6 | 1 | 4.00 | 147.00 | 223.30 | 2 | 102.35 | 1.62 | 2 | 2 | 7 | 1 |
| 3 | 59.0 | 1 | 4.00 | 112.37 | 187.25 | 0 | 158.16 | 0.00 | 1 | 1 | 7 | 1 |
| 4 | 51.1 | 1 | 1.95 | 138.03 | 238.48 | 0 | 172.54 | 1.15 | 1 | 1 | 3 | 0 |

- Health analytics: prediction of heart disease
  - Diagnosis and therapy support systems
  - Identify most important risk factors

# Application Examples

- Sentiment Analysis / Opinion Mining (over text)

Regression



**94%**
TOMATOMETER

**90%**
AUDIENCE SCORE
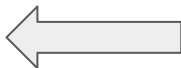


What you will be getting when you walk into an inevitably overstuffed movie theater is something singular that reflects our age in a way that none of the MCU films that preceded it have-indeed, very few Hollywood spectacles ever have.

May 1, 2019 | Rating: 3.5/4 | Full Review...

**Oliver Jones**
Observer
⭐ Top Critic

What's missing from "Endgame" is the free play of imagination, the liberation of speculation, the meandering paths and loose ends that start in logic and lead to wonder.

April 28, 2019 | Full Review...

**Richard Brody**
New Yorker
⭐ Top Critic

Endgame consists almost entirely of the downtime scenes that were always secretly everyone's favorite parts of these movies anyway.

April 26, 2019 | Full Review...

**Dana Stevens**
Slate
⭐ Top Critic
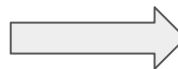
Classification



*Fresh* — Tomatometer

*Rotten* — Tomatometer

*Fresh* — Audience

*Rotten* — Audience

# Application Examples

- ## Self-driving vehicles
  - Input: image & sensor data

- ## Regression tasks, e.g.:
  - Acceleration
  - Steering angle
  - Event prediction

- ## Classification tasks, e.g.:
  - Obstacle detection
  - Street sign identification

# Outline

- **Classification & Regression**
  - Overview & Examples

  - **Basic Setup**

  - Evaluation


- Nearest Neighbor Methods
  - KNN — K-Nearest Neighbor Algorithm

  - Pros, Cons & Caveats

# Supervised Training/Learning — Basic Setup

### Training Data

$X_{train}$

| Age | Marital Status | Income Level |
|-----|----------------|--------------|
| 23 | Single | Mid |
| 35 | Married | High |
| 26 | Single | High |
| 41 | Single | Mid |
| 18 | Single | Low |
| 55 | Married | High |
| 30 | Single | High |
| 35 | Married | High |
| 28 | Married | Mid |
| 45 | Married | Mid |

$y_{train}$

| Credit Approval |
|-----------------|
| No |
| Yes |
| No |
| Yes |
| No |
| Yes |
| Yes |
| Yes |
| Yes |
| No |

### Test Data

$X_{test}$

| Age | Marital Status | Income Level |
|-----|----------------|--------------|
| 25 | Single | Low |
| 28 | Married | Mid |
| 45 | Married | High |
| 41 | Single | High |
| 21 | Single | Mid |
| 26 | Married | Mid |

### Predictions

$\hat{y}_{test}$

| Credit Approval |
|-----------------|
| No |
| Yes |
| No |
| Yes |
| No |
| Yes |

**Evaluation**

### Ground Truth

$y_{test}$

| Credit Approval |
|-----------------|
| No |
| Yes |
| Yes |
| Yes |
| No |
| No |

**Model**

**(classification / regression)**

13
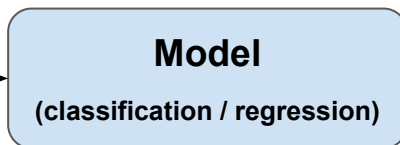
# Supervised Training/Learning — Training & Test Data

- Given: Dataset D of pairs {(x, y)}
  - x — features (independent variables)
  - y — label (dependent variable)

- Split dataset D into:
  - $D_{train}$ — data for training the model
  - $D_{test}$ — data for evaluating the model
  - Important: $D_{train} \cap D_{test} = \varnothing$
    (the test data is not allowed to be used during training)

| Age | Education | Marital Status | Income Level | Credit Approval |
|-----|-----------|----------------|--------------|-----------------|
| 23 | Masters | Single | Mid | No |
| 35 | College | Married | High | Yes |
| 26 | Masters | Single | High | No |
| 41 | PhD | Single | Mid | Yes |
| 18 | Poly | Single | Low | No |
| 55 | Poly | Married | High | Yes |
| 30 | College | Single | High | Yes |
| 35 | PhD | Married | High | Yes |
| 28 | Masters | Married | Mid | Yes |
| 45 | Masters | Married | Mid | No |

$D_{train}$ (rows 1–6), $D_{test}$ (rows 7–10)

features | labels

# Supervised Training/Learning

- Model — Parameterized function $h(x, \Theta) = y$
  - Maps input space (features) to the output space (labels)

  - $\Theta$ — trainable/learnable parameters of the model
    (note: not all models are parameterized)

- Model selection — Selection of a "family" of functions $h(x, \Theta) = y$
  - K-Nearest Neighbor, Decision Trees, Linear Models, Neural Networks, …

- Training / Learning
  - Process of systematically finding the best values for $\Theta$

  - $\Theta_{best}$ ⇔ best mapping between features and labels

# Outline

# Classification & Regression — Evaluation

**Regression**

$\hat{y}_{test}$       $y_{test}$

| Credit Limit |
|:---:|
| $S9,000 |
| $S4,000 |
| $S5,800 |
| $S10,000 |
| $S11,000 |
| $S3,500 |

| Credit Limit |
|:---:|
| $S10,000 |
| $S4,000 |
| $S6,000 |
| $S12,000 |
| $S10,000 |
| $S3,000 |
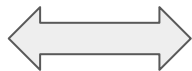
Direct comparison of numerical values

Common: Root Mean Squared Error (RSME):

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(\hat{y}_i - y_i)^2}{N}}$$

**Classification**

$\hat{y}_{test}$       $y_{test}$

| Credit Approval |
|:---:|
| No |
| Yes |
| No |
| Yes |
| No |
| Yes |

| Credit Approval |
|:---:|
| No |
| Yes |
| Yes |
| Yes |
| No |
| No |
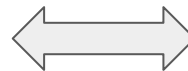
Question: How to compare classification results?

# Classification: Evaluation

- Base case: Binary classification (2 labels: 0 or 1)

- Example: COVID-19 test

**test result**

$\hat{y}_{test}$

| |
|---|
| 0 |
| 0 |
| 0 |
| 1 |
| 0 |
| 1 |

**infected**

$y_{test}$

| |
|---|
| 0 |
| 0 |
| 1 |
| 1 |
| 0 |
| 0 |
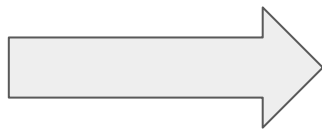
## Confusion Matrix

ground truth label $y$

| | **1** | **0** |
|---|---|---|
| **1** | 1 | 1 |
| **0** | 1 | 3 |

predicted label $\hat{y}$

# Classification: Evaluation — Confusion Matrix

ground truth label $y$

| predicted label $\hat{y}$ | 1 | 0 |
|---|---|---|
| **1** | True Positives (TP) | False Positives (FP) |
| **0** | False Negatives (FN) | True Negatives (TN) |
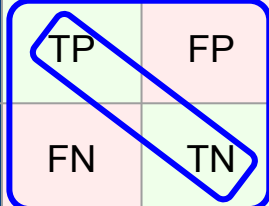
**True Positives (TP)**:     Number of positive classes that have been correctly predicted as positive

**True Negatives (TN)**:     Number of negative classes that have been correctly predicted as negative

**False Positives (FP):**     Number of negative classes that have been incorrectly predicted as positive

**False Negatives (FN):**     Number of positive classes that have been incorrectly predicted as negative

# Classification: Evaluation — Popular Metrics

- ## Accuracy

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$
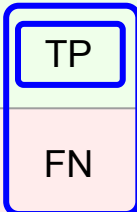


- ## Sensitivity, Specificity

$$Sensitivity = \frac{TP}{TP + FN}$$



$$Specificity = \frac{TN}{TN + FP}$$

# Classification: Evaluation — Popular Metrics

- Precision, Recall, F1 Score

**Harmonic Mean** of Precision and Recall

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

# Classification: Evaluation — Why so Many Measures?

- Problem: (Highly) imbalanced datasets

- Example use case: COVID-19 test
    - Most people in a population are not infected
    - Assume a test that always(!) returns "negative"

ground truth label $y$

|  | **1** | **0** |
|---|---|---|
| **1** | 0 | 0 |
| **0** | 200 | 10,000 |

predicted label $\hat{y}$

$$Accuracy = \frac{0 + 10000}{0 + 0 + 10000 + 200} = 98\%$$

$$Specificity = \frac{10000}{10000 + 0} = 100\%$$

➔ Very high values despite "useless" test

# Classification: Evaluation — Why so Many Measures?

- **Observation: FP and FN not always equally problematic**

$$Precision = \frac{TP}{TP + FP} \qquad Recall = \frac{TP}{TP + FN}$$

- **Example: Heart disease prediction**
  - BAD: misclassifying a high-risk person
  - OK-ish: misclassifying a healthy person

  **Recall > Precision**

- **Example: News article classification**
  (e.g., for search engines such as Google News)
  - BAD: showing article of wrong category
  - OK: missing a relevant article in result

  **Recall < Precision**

# Classification: Evaluation — Numerical Class Scores

- Assumption so far:
  $\hat{y}$ are **class labels**

- Output of many models:
  $\hat{y}$ are **class scores**
  - e.g., probability of a class

→ **Thresholding**
  - Use threshold to convert $\hat{y}$ to a binary variable 0/1

|  | $\hat{y}_{0.5}$ | $\hat{y}_{0.43}$ | $y$ |
|---|---|---|---|
| 0.45 | 0.45 → **0** | 0.45 → **1** | 1 |
| 0.30 | 0.30 → **0** | 0.30 → **0** | 0 |
| 0.55 | 0.55 → **1** | 0.55 → **1** | 0 |
| 0.25 | 0.25 → **0** | 0.25 → **0** | 0 |
| 0.35 | 0.35 → **0** | 0.35 → **0** | 1 |
| 0.55 | 0.55 → **1** | 0.55 → **1** | 1 |
| 0.30 | 0.30 → **0** | 0.30 → **0** | 0 |
| 0.60 | 0.60 → **1** | 0.60 → **1** | 1 |
| 0.40 | 0.40 → **0** | 0.40 → **0** | 0 |
| 0.45 | 0.45 → **0** | 0.45 → **1** | 1 |
| class scores | threshold=0.5 | threshold=0.43 | ground truth |

# Classification: Evaluation — Numerical Class Scores

- Different threshold yield different results



$$F1 = 0.66$$

$$F1 = 0.80$$

➜ Question: What threshold to use? — Answer: Try them all!

# Receiver Operating Characteristic (ROC)

- Plot **True Positive Rate** (Sensitivity) against **False Positive Rate** (1-Specificity)
  - Plot values for each meaningful threshold $t$ (meaningful = has effect on result)



class scores    ground truth

both sorted w.r.t. scores
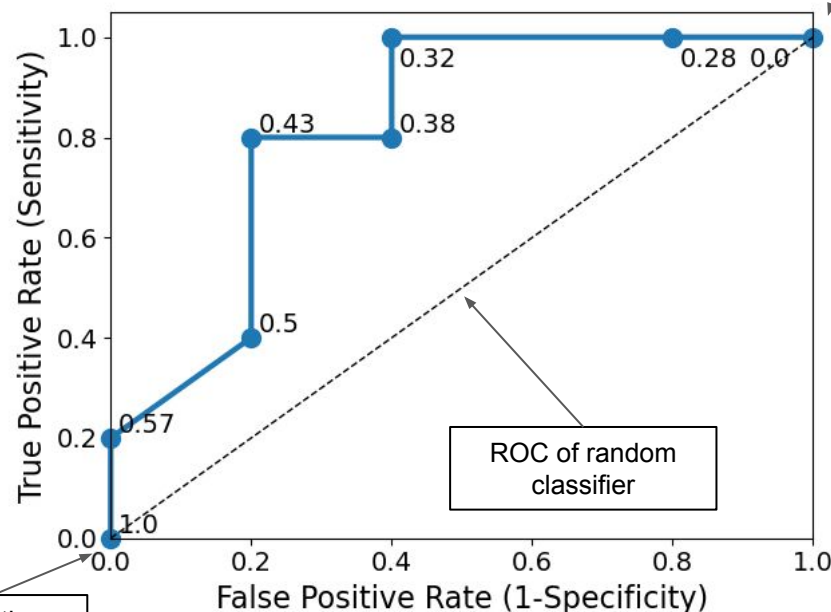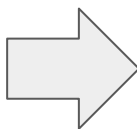
# Receiver Operating Characteristic (ROC) — Comparison

Class scores from 3 different classifiers
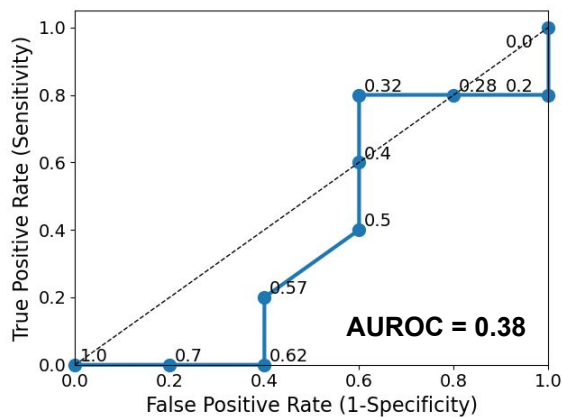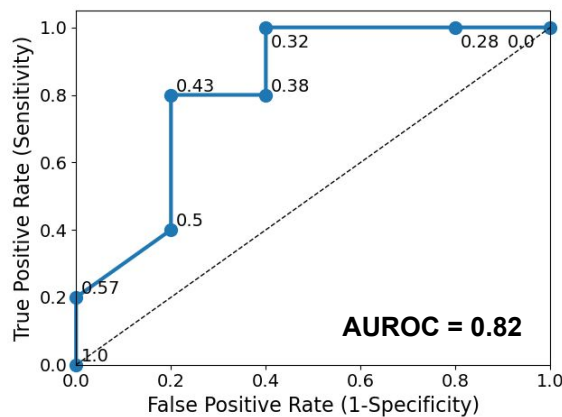


Poor classifier          "Decent" classifier          Perfect classifier
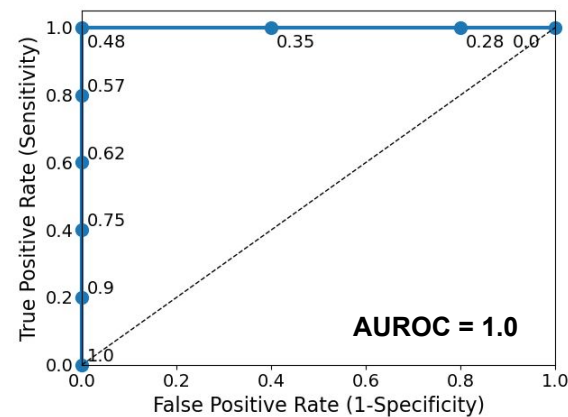
- Quantify quality of classifiers using **AUROC** (or AUC)
  - Area Under Receiver Operating Characteristic
  - AUROC of random classifier: 0.5

# ROC / AUROC — Practical Example

- **IRIS dataset**
  - 3 classes of Iris plants
  - 50 samples per class
  - 4 continuous features
    (sepal/petal length/width)

- **ROC / AUROC**
  - More samples, more thresholds
  - Smoother ROC
  - Thresholds typically omitted
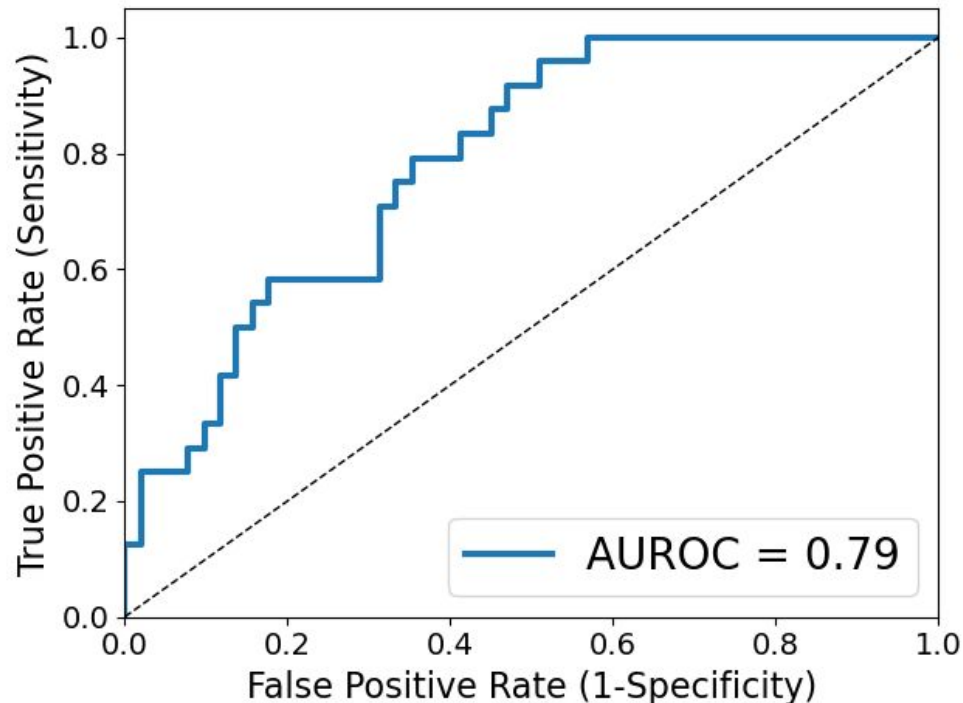
# Quick Quiz

# Quick Quiz
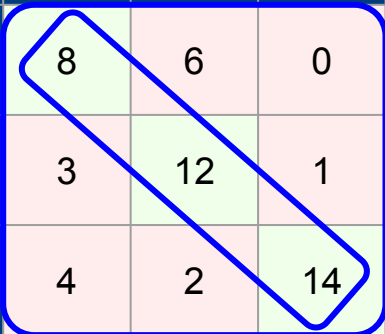
# Classification: Evaluation — Beyond 2 Classes
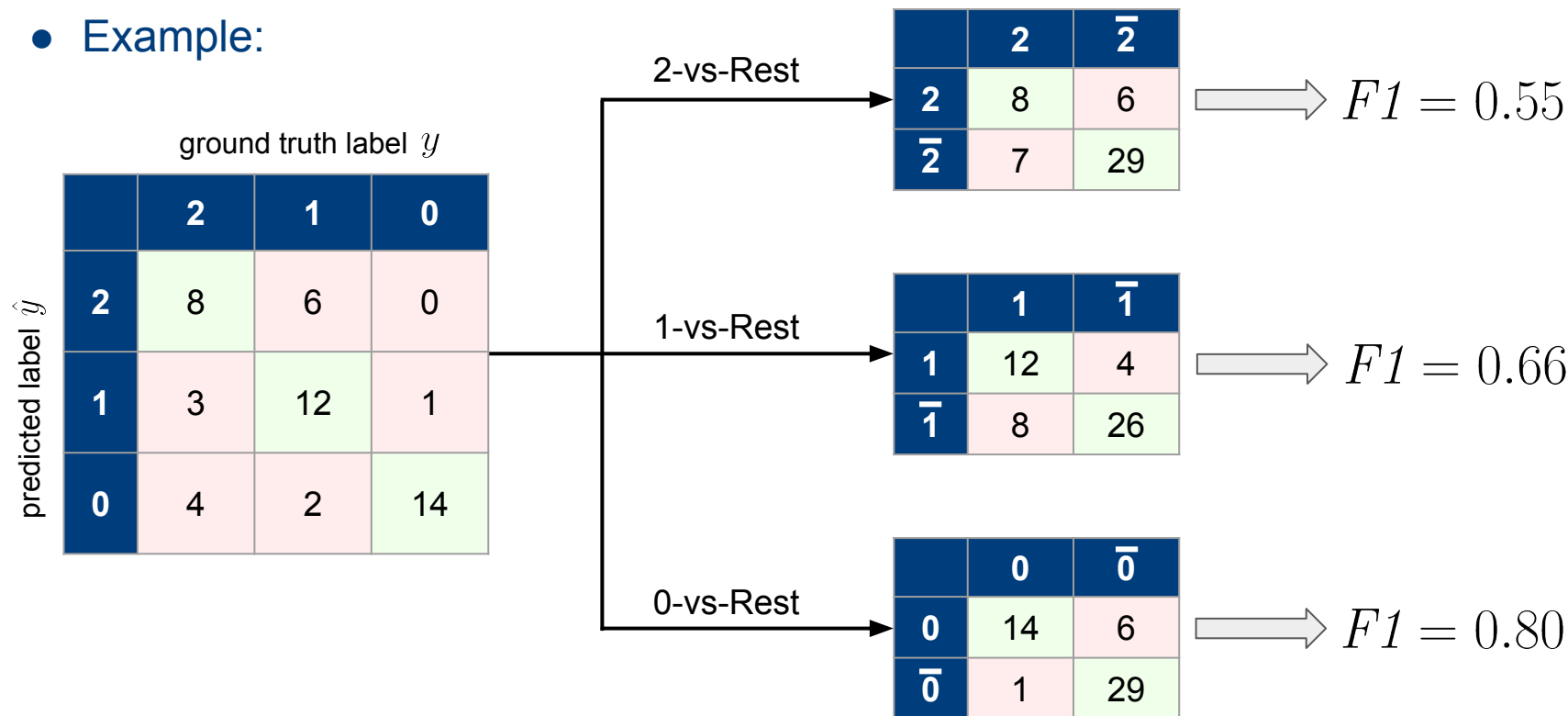
- Example: 3 classes, 50 samples

ground truth label $y$

| | 2 | 1 | 0 |
|---|---|---|---|
| **2** | 8 | 6 | 0 |
| **1** | 3 | 12 | 1 |
| **0** | 4 | 2 | 14 |

predicted label $\hat{y}$

$$Accuracy = \frac{8 + 12 + 14}{8 + 12 + 14 + 6 + 3 + 1 + 4 + 2} = 0.68$$

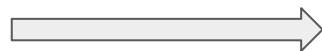# Multiclass Evaluation — One-vs-Rest Confusion Matrices

- Example:

ground truth label $y$

| | **2** | **1** | **0** |
|---|---|---|---|
| **2** | 8 | 6 | 0 |
| **1** | 3 | 12 | 1 |
| **0** | 4 | 2 | 14 |

predicted label $\hat{y}$

**2-vs-Rest**

| | **2** | **$\overline{2}$** |
|---|---|---|
| **2** | 8 | 6 |
| **$\overline{2}$** | 7 | 29 |

$\Longrightarrow$ $F1 = 0.55$

**1-vs-Rest**

| | **1** | **$\overline{1}$** |
|---|---|---|
| **1** | 12 | 4 |
| **$\overline{1}$** | 8 | 26 |

$\Longrightarrow$ $F1 = 0.66$

**0-vs-Rest**

| | **0** | **$\overline{0}$** |
|---|---|---|
| **0** | 14 | 6 |
| **$\overline{0}$** | 1 | 29 |

$\Longrightarrow$ $F1 = 0.80$

# One-vs-Rest — Micro Averaging

|  | 2 | $\overline{2}$ |
|---|---|---|
| **2** | 8 | 6 |
| **$\overline{2}$** | 7 | 29 |

|  | 1 | $\overline{1}$ |
|---|---|---|
| **1** | 12 | 4 |
| **$\overline{1}$** | 8 | 26 |

|  | 0 | $\overline{0}$ |
|---|---|---|
| **0** | 14 | 6 |
| **$\overline{0}$** | 1 | 29 |

Average over all
TP, FP, FN, TN

|  | C | $\overline{C}$ |
|---|---|---|
| **C** | 11.33 | 5.33 |
| **$\overline{C}$** | 5.33 | 28 |
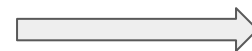
$$F1 = 0.68$$

# One-vs-Rest — Macro Averaging

|  | 2 | $\overline{2}$ |
|---|---|---|
| **2** | 8 | 6 |
| $\overline{2}$ | 7 | 29 |

$\Longrightarrow \quad F1 = 0.55$

|  | 1 | $\overline{1}$ |
|---|---|---|
| **1** | 12 | 4 |
| $\overline{1}$ | 8 | 26 |

$\Longrightarrow \quad F1 = 0.66$

|  | 0 | $\overline{0}$ |
|---|---|---|
| **0** | 14 | 1 |
| $\overline{0}$ | 6 | 29 |

$\Longrightarrow \quad F1 = 0.80$

Average over
all metrics

$\Longrightarrow \quad F1 = 0.67$

# One-vs-Rest — Macro vs. Micro Averaging

- Both methods use One-vs-Rest confusion matrices
  - All introduced metrics applicable (incl. ROC/AUROC)

- Micro-averaging
  - Averaging over TP, FP, FN, TN values of all One-vs-Rest confusion matrices
  - Favors bigger classes (since average over counts)

- Macro-averaging
  - Averaging over metrics derived from each One-vs-Rest confusion matrix
  - Treats all class equally (since metrics are normalized)

# ROC / AUROC — Practical Example (Multiclass)

- IRIS dataset
  - 3 classes of Iris plants
  - 50 samples per class
  - 4 continuous features
    (sepal/petal length/width)



ROC curves:
- micro-average ROC curve (area = 0.73)
- macro-average ROC curve (area = 0.78)
- ROC curve of Class 0 (area = 0.91)
- ROC curve of Class 1 (area = 0.60)
- ROC curve of Class 2 (area = 0.79)

X-axis: False Positive Rate (1-Specificity)
Y-axis: True Positive Rate (Sensitivity)

Adopted from https://scikit-learn.org/stable/auto_examples/model_selection/plot_roc.html
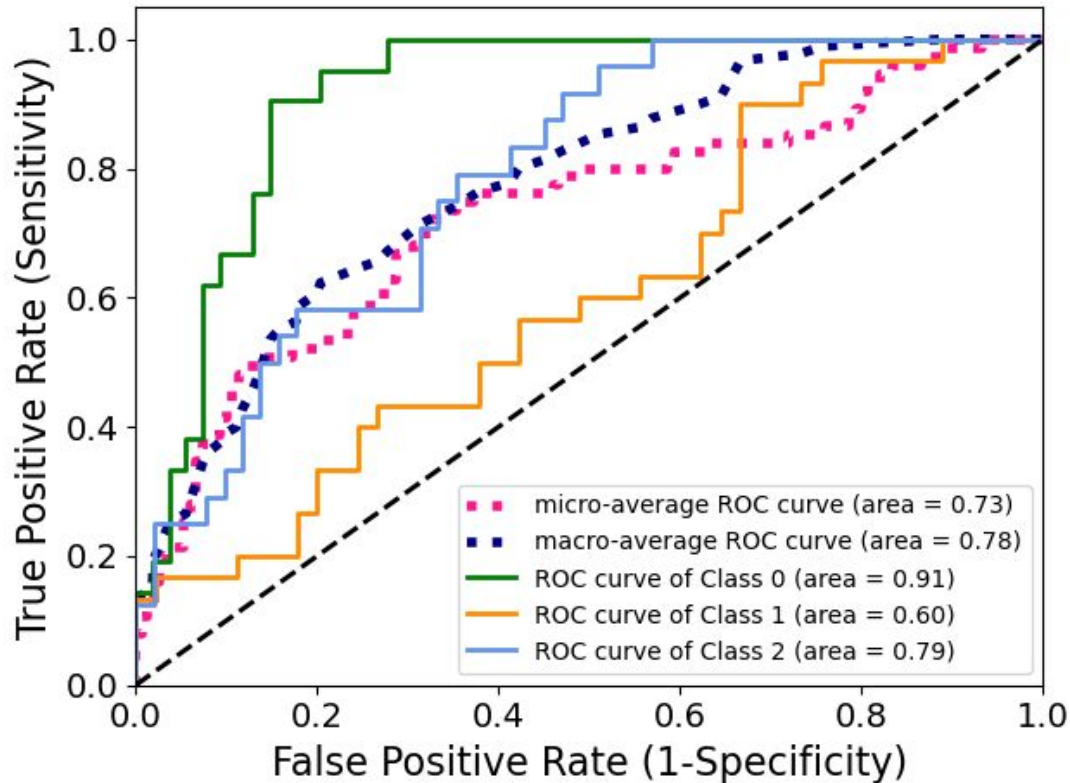
# Quick Quiz

# Outline

- Classification & Regression
  - Overview & Examples

  - Basic Setup

  - Evaluation

- **Nearest Neighbor Methods**
  - **KNN — K-Nearest Neighbor Algorithm**

  - Pros, Cons & Caveats

# K-Nearest Neighbor Algorithm (KNN)

- Intuition behind KNN:
  - Label of an unseen data point $x$ derives from the labels of the k-nearest neighbors of $x$
  - Similar data points ➔ similar labels

  Required: notion of similarity/distance

- Example
  - What should be the label/color of the unseen (gray) data point?
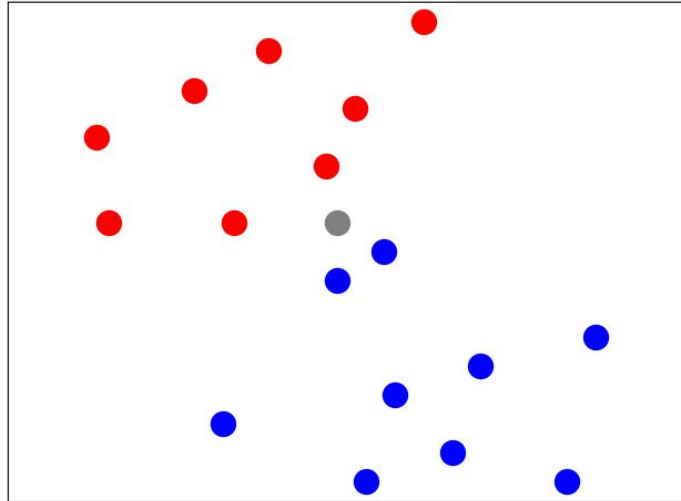
# KNN for Classification

- "Training"
  - Remember training data

- Prediction for unseen point $x_i$
  - Calculate distances to all
    training data points $x_j$, e.g.:

    $$d(p, q) = \sqrt{\sum_{i}^{d} (q_i - p_i)^2}$$

    Euclidean distance, d = #features

  - Get the k-nearest neighbors

  - **Label of $x_i$ = most frequent label
    among all k-nearest neighbors**
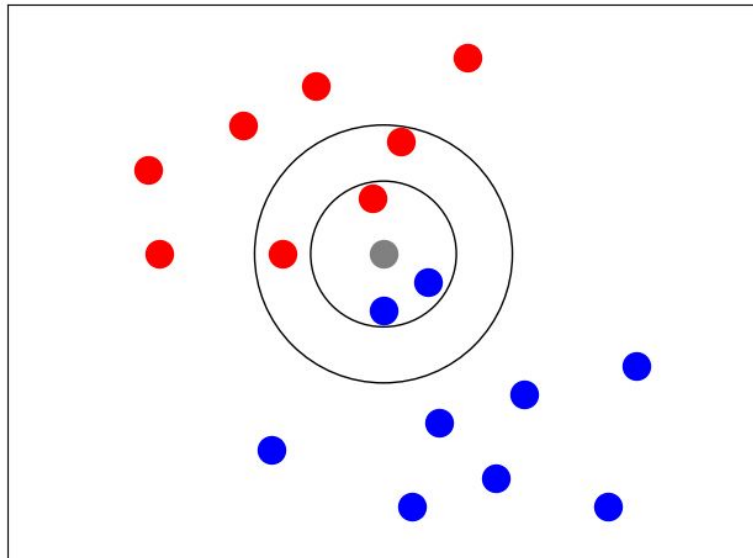


$k$ is typically odd to minimize the chance of ties

# Quick "Quiz"

# KNN for Classification — Distance Metrics

- Example for different distance metrics
  - **Euclidean distance**
  - **Manhattan distance**
  - **Chebyshev distance**

- Other metrics, e.g.:
  - Cosine similarity
  - Jaccard similarity
  - ...
  - User-defined metrics
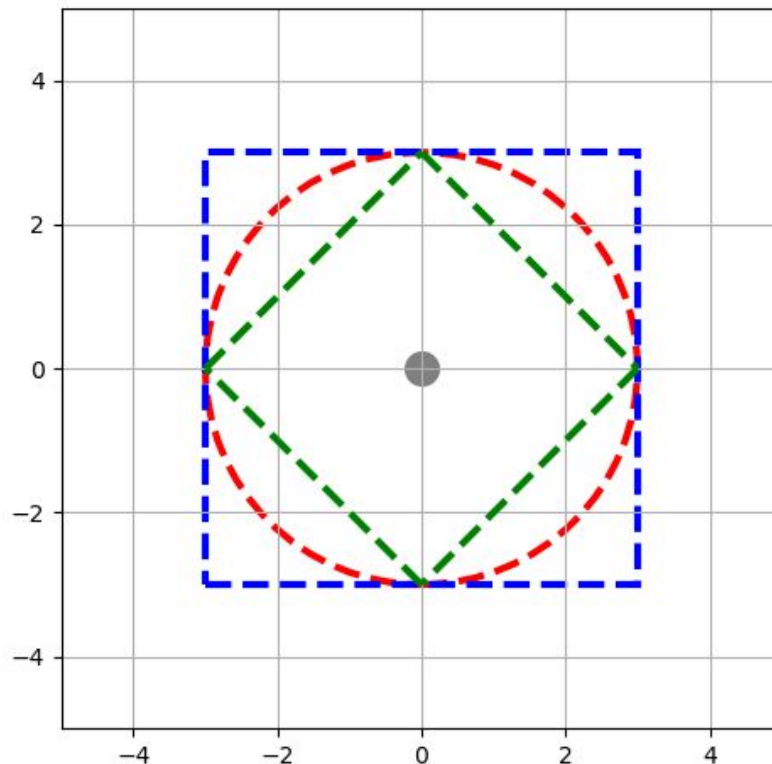
# KNN for Classification — Example

- IRIS dataset
  - 3 classes, 50 samples per class
  - 4 continuous features, but only 2 used: (sepal length & width)

- Basic EDA
  - Class "Red" well separated
  - A lot of overlap between Classes "Green" and "



120 training samples: 40 Red, 39 Green, 41 Blue
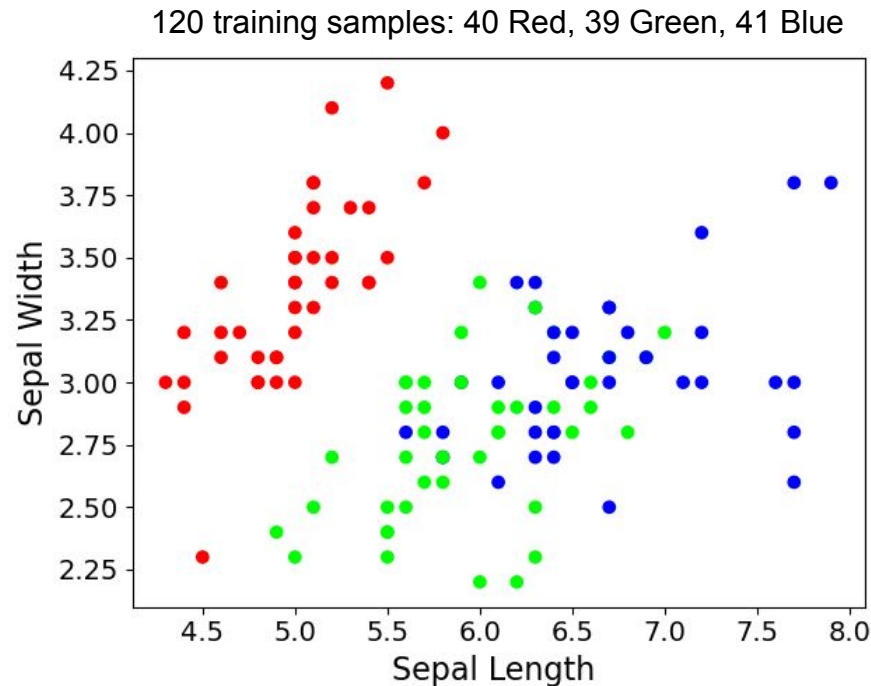
# KNN for Classification — Example

- Common outcomes
  - Different *k* yield different results

  - (Very) small *k* — results very sensitive to noise and outliers

  - Large *k* — insufficient capacity to properly sep

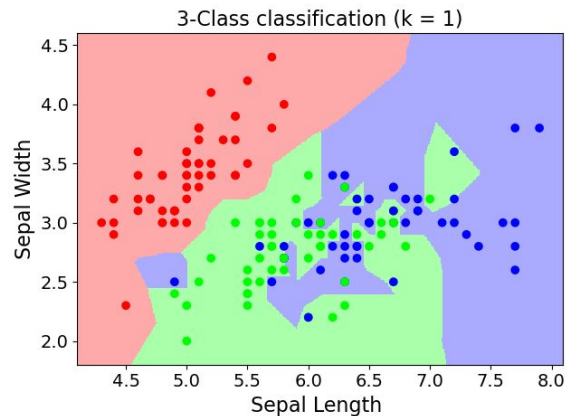  - Very large *k* — most frequent class in training data starts to dominate

30 test samples
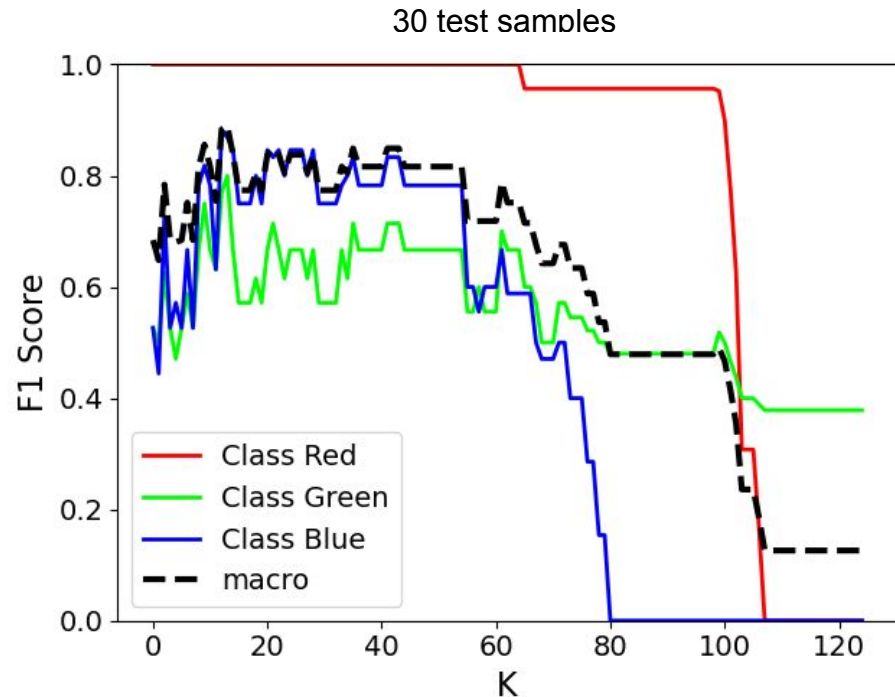
# 1-Nearest Neighbor: Voronoi Tessellation

- Decision boundaries for 1-NN =derived from Voronoi Tessellation
  - Separation of space into cells
  - Cell: set of points nearest to a single data point



Decision boundary between "Red" and "Blue" class
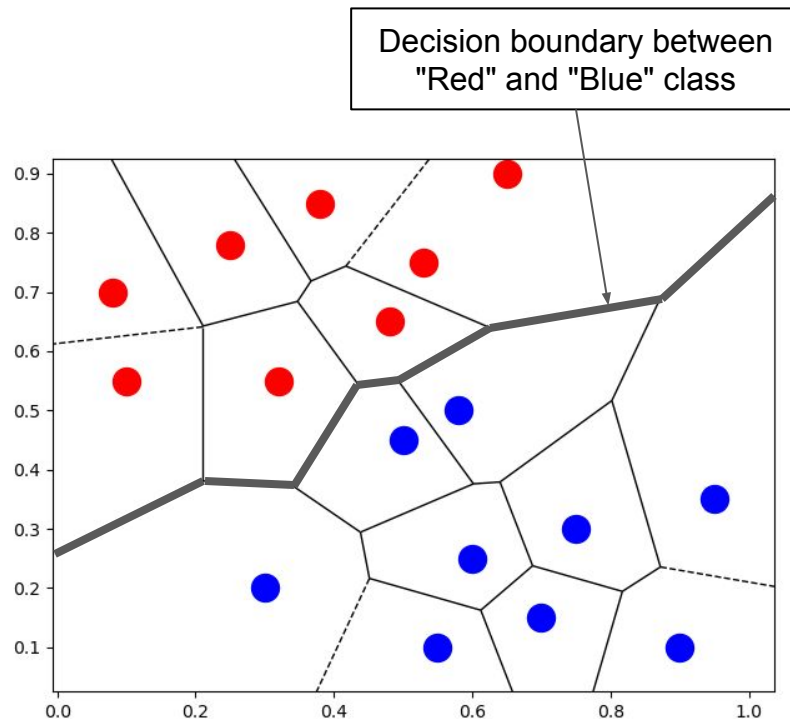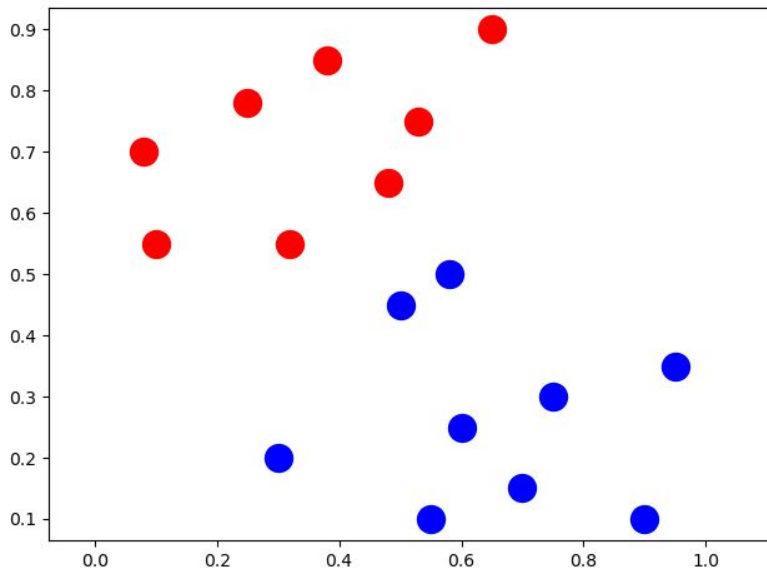
# KNN for Classification — Example

# Quick Quiz

# KNN for Regression

- "Training"
  - Remember training data

- Prediction for unseen point $x_i$
  - Calculate distances to all training data points $x_j$, e.g.:

    $$d(p,q) = \sqrt{\sum_i^d (q_i - p_i)^2}$$

    Euclidean distance, d = #features

  - Get the k-nearest neighbors

  - **Label of $x_i$ = mean of scores of all k-nearest neighbors**



Mean of 5-NN

Mean of 3-NN
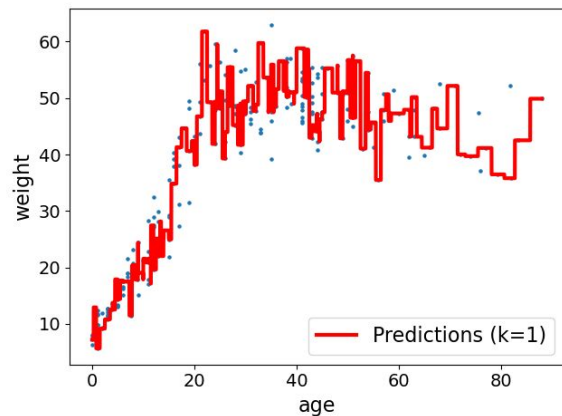
$x_i$
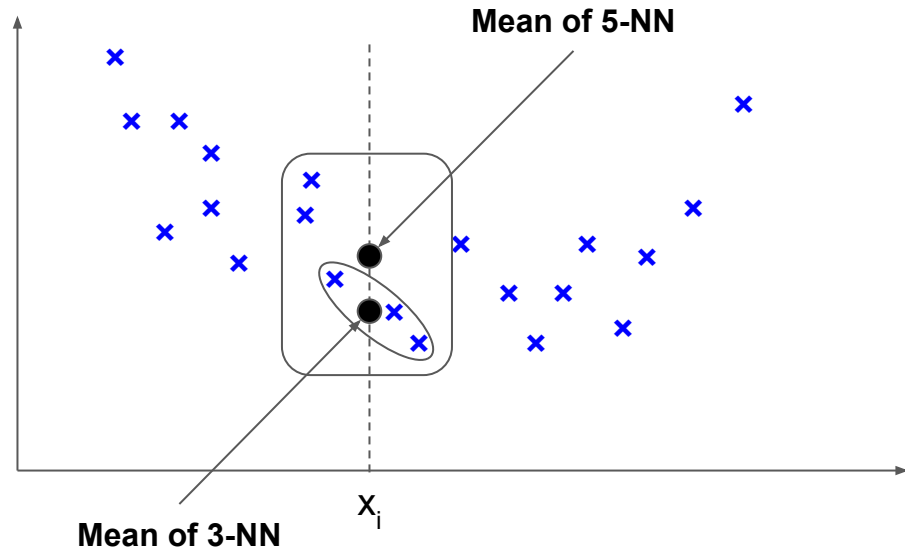
# KNN for Regression — Example

- "age vs. weight" dataset
  - age and weight of 257 males
  - 200 training samples, 57 test samples

- Common outcomes (again)
  - Different $k$ yield different scores
  - (Very) small $k$ — predicted scores very sensitive to noise and outliers
  - Large $k$ — mean over too many neighbors
  - Very large $k$ — predicted scores converge to the mean over
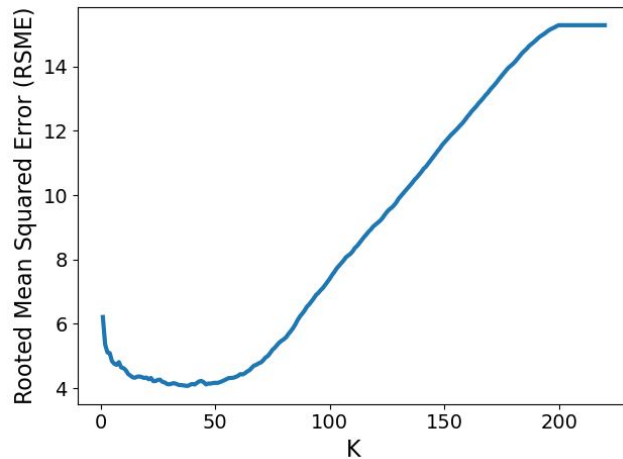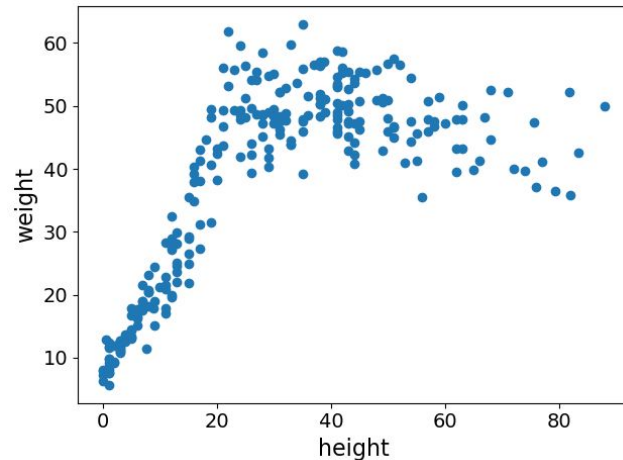
# KNN for Regression — Example
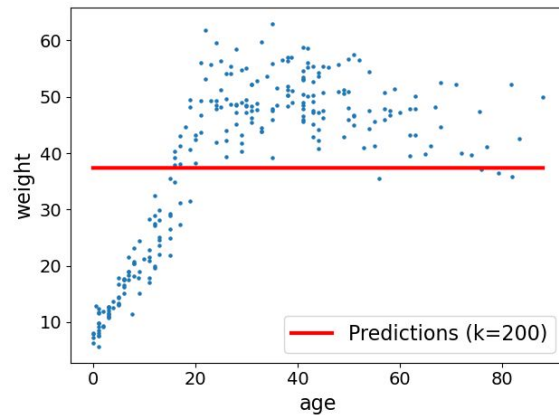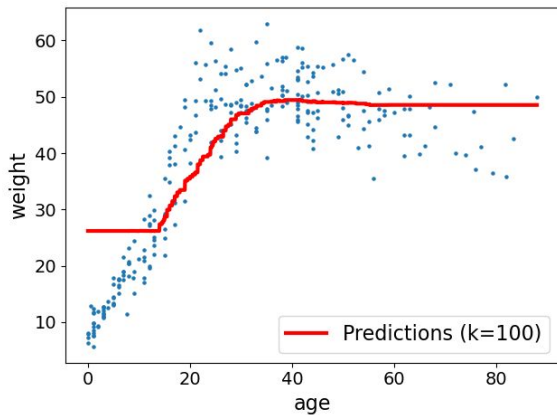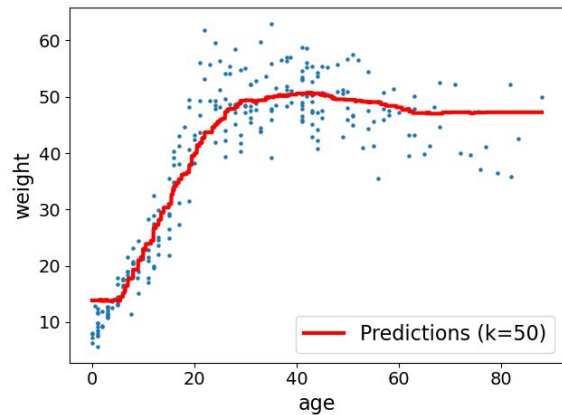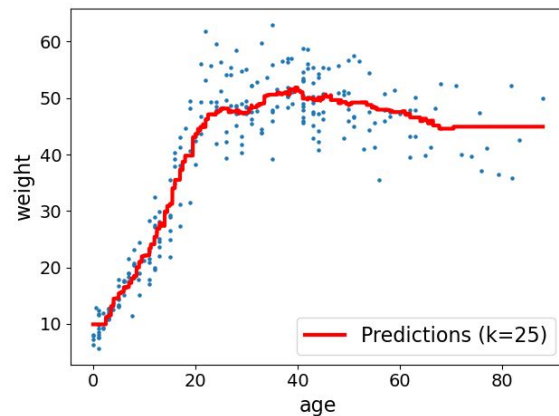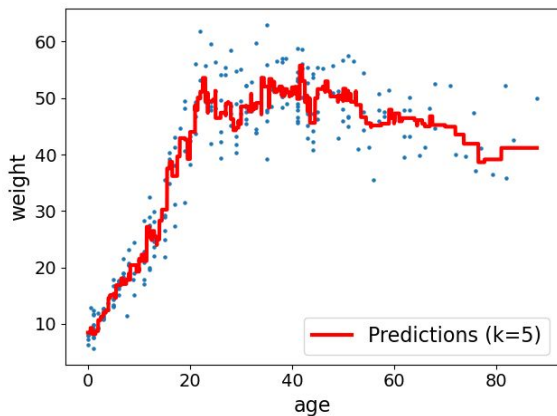
# Choice of *k* — Summary

- *k* too small
  - Predictions sensitive to noise/outliers
  - Very uneven decision boundaries (or regression lines)

  Risk of **overfitting**

- *k* too large
  - Unable to capture local patterns
  - Very smooth decision boundaries (or regression lines)

  Risk of **underfitting**

# Outline

- **Classification & Regression**
  - Overview & Examples
  - Basic Setup
  - Evaluation

- **Nearest Neighbor Methods**
  - KNN — K-Nearest Neighbor Algorithm
  - **Pros, Cons & Caveats**

# Pros & Cons...and Caveats

- ## Pros
  - Very simple and intuitive algorithm — but often surprisingly good performance!
  - Generic algorithm — applicable as long as distance between points can be "meaningfully" measured
  - Can produce arbitrarily shaped decision boundaries
  - No training time

- ## Cons
  - Finding neighbors at test time may be slow
    (in practice: data structures and auxiliary algorithms to speed up k-NN search)
  - All training data need to be stored

- ## Caveats
  - *"...applicable as long as distance between points can be "meaningfully" measured"*

# Caveat 1: Feature Values (Range, Magnitude)

- Euclidean distance sensitive to range and magnitude of attribute values

| id | weight (kg) | height (cm) | sex |
|----|-------------|-------------|-----|
| A | 80 | 165 | male |
| B | 55 | 180 | female |
| C | 70 | 180 | **???** |

| id | weight (kg) | height (m) | sex |
|----|-------------|------------|-----|
| A | 80 | 1.65 | male |
| B | 55 | 1.80 | female |
| C | 70 | 1.80 | **???** |

$$d(A,C) = \sqrt{(70-80)^2 + (180-165)^2} = 18.0$$

$$d(B,C) = \sqrt{(70-55)^2 + (180-180)^2} = 15.0$$

➜ C classified as female

$$d(A,C) = \sqrt{(70-80)^2 + (1.80-1.65)^2} = 10.0$$

$$d(B,C) = \sqrt{(70-55)^2 + (1.80-1.80)^2} = 15.0$$

➜ C classified as male

# Data Normalization

- Z-score normalisation, e.g.:

$$x_i^{weight} = \frac{x_i^{weight} - \mu^{weight}}{\sigma^{weight}}$$

- Min-max normalization, e.g.:

$$x_i^{weight} = \frac{x_i^{weight} - min(x^{weight})}{max(x^{weight}) - min(x^{weight})}$$

| id | weight (kg) | height (m) | sex |
|----|-------------|------------|--------|
| A  | 80          | 1.65       | male   |
| B  | 55          | 1.80       | female |
| C  | 70          | 1.80       | **???** |

Z-score normalisation

| id | weight | height | sex |
|----|--------|--------|--------|
| A  | 1.14   | -1.41  | male   |
| B  | -1.30  | 0.71   | female |
| C  | 0.16   | 0.71   | **???** |

$$d(A, C) = 2.33, \quad d(B, C) = 1.46$$

→ C classified as female

# Data Normalization — Visualized

- (weight, height) data points for 187 females (red) and 165 males (blue)
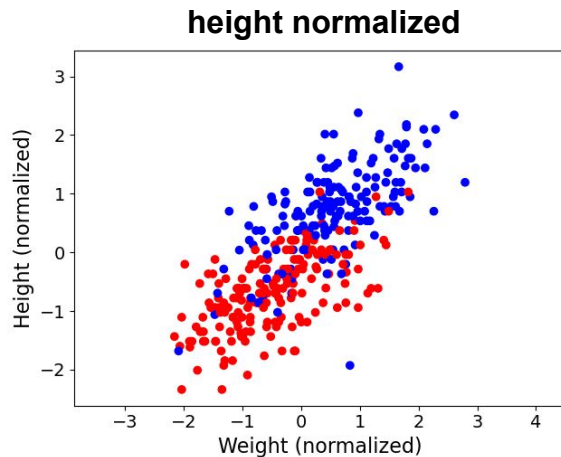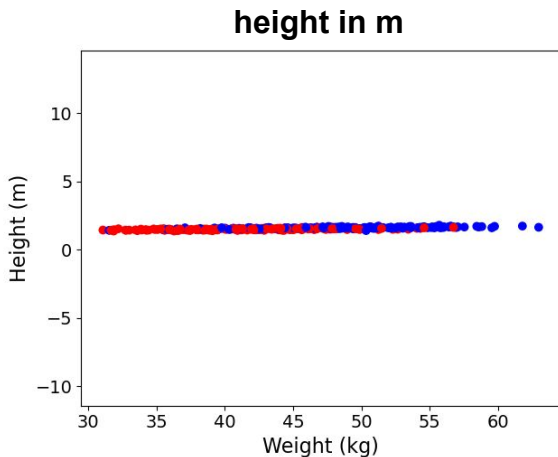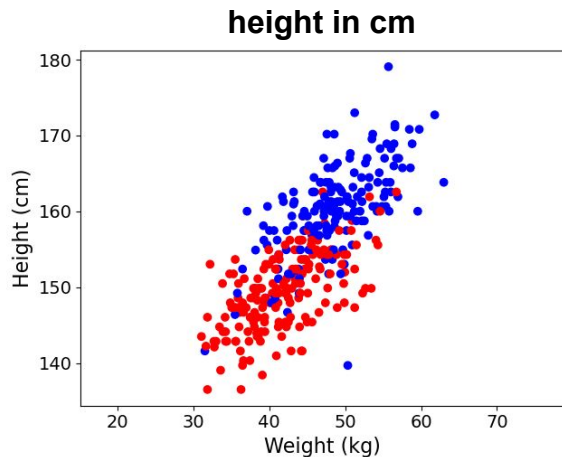


**Note:** Normalization assumes that all features are equally important. This may not always be true!
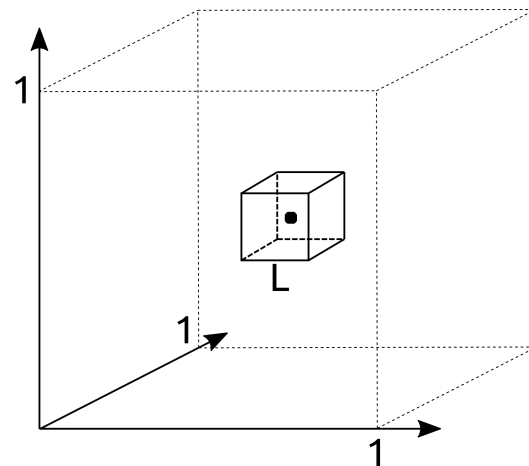
# Caveat 2: Curse of Dimensionality



- Effect of high dimensions (i.e., many features)
  - Data points tend to never be close together

  - Average distance between points converges

- Intuition
  - Assume *N* data points uniformly distributed within a unit cube with *d* dimensions

  - Let *L* be the length of the smallest cube containing the *k*-NN of a data point

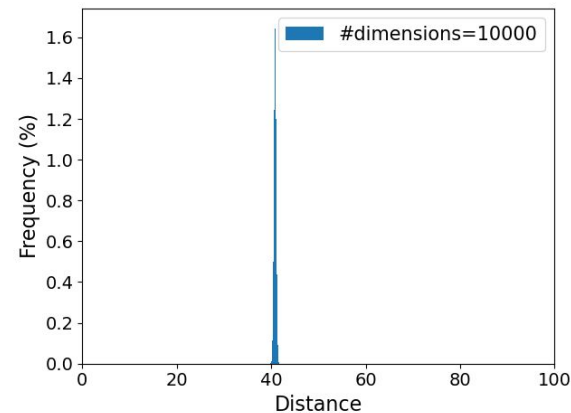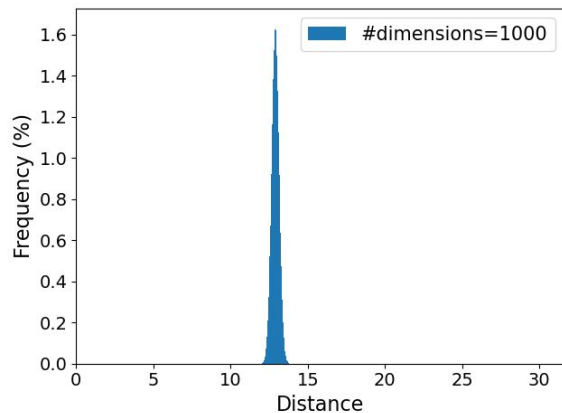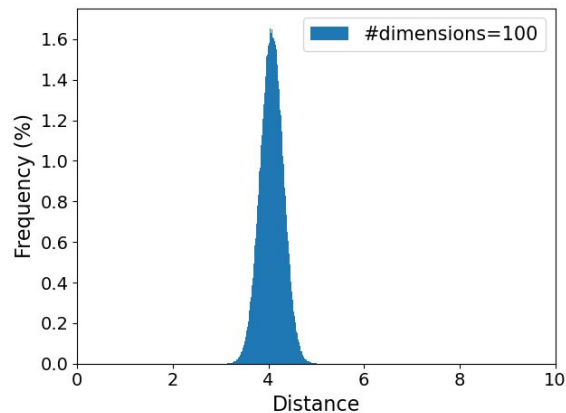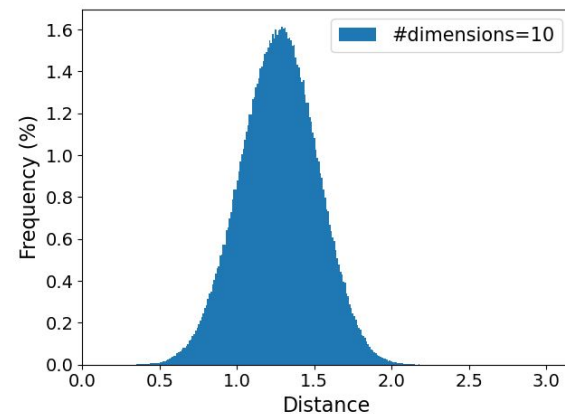$$ L^d \approx \frac{k}{N} \;\Rightarrow\; L \approx \left( \frac{k}{N} \right)^{\frac{1}{d}} $$

N=1,000, k=10

| d | L |
|---|---|
| 2 | 0.100 |
| 3 | 0.215 |
| 10 | 0.631 |
| 100 | 0.955 |
| 1,000 | 0.995 |
| 10,000 | 0.999 |

**The cube with the k-NN is almost the whole unit cube!**

# Curse of Dimensionality

Distribution of pairwise distances between 1,000 random data points and different number of dimensions
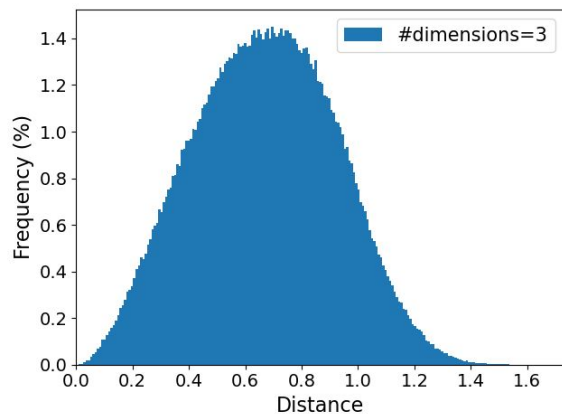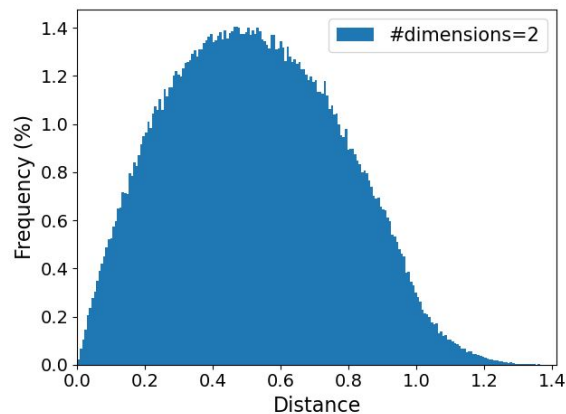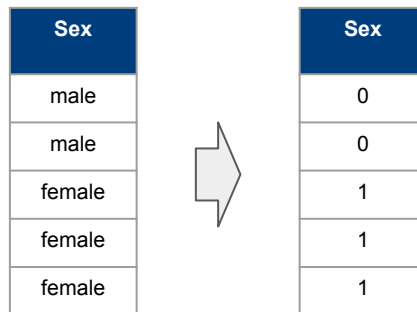
# Caveat 3: Non-Numerical Data

- In practice: Features often categorical
  - Ordinal — education level, grades, etc.
  - Nominal — sex, marital status, etc.

- Basic approach: Convert categorical features into numerical features
  - Allows direct use of common distance metrics
  - Does not automatically yield good results

| Age | Sex | Education | Marital Status |
|-----|------|-----------|----------------|
| 23 | male | Masters | Single |
| 35 | male | College | Married |
| 26 | female | Masters | Single |
| 41 | female | PhD | Single |
| 18 | female | Poly | Single |
| 55 | male | Poly | Divorced |
| 30 | female | College | Single |
| 35 | make | PhD | Married |
| 28 | male | Masters | Married |
| 45 | female | Masters | Married |

# Caveat 3: Non-Numerical Data

- ## Binary features
  - Special case of categorical data
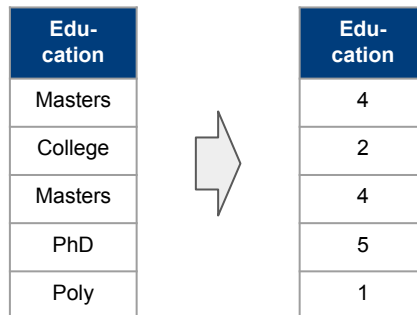  - Convert into binary variable 0/1 to indicate absence/presence

| Sex |
|---|
| male |
| male |
| female |
| female |
| female |

| Sex |
|---|
| 0 |
| 0 |
| 1 |
| 1 |
| 1 |

- ## Ordinal features
  - Utilize natural order of feature values
  - Convert into numerical values while preserving their original order

| Edu-cation |
|---|
| Masters |
| College |
| Masters |
| PhD |
| Poly |

| Edu-cation |
|---|
| 4 |
| 2 |
| 4 |
| 5 |
| 1 |

# Caveat 3: Non-Numerical Data

- Nominal features (with $n$ different values)
  - **One-hot encoding**: convert nominal feature into $N$ binary features

  - Convert each new binary feature into binary variable 0/1 to indicate absence/presence

  - Increases dimensionality!
    (particularly if N is very large)

| Marital Status |
| --- |
| Single |
| Married |
| Single |
| Single |
| Single |
| Divorced |

| Single | Married | Divorced |
| --- | --- | --- |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 0 | 0 |
| 1 | 0 | 0 |
| 0 | 0 | 1 |

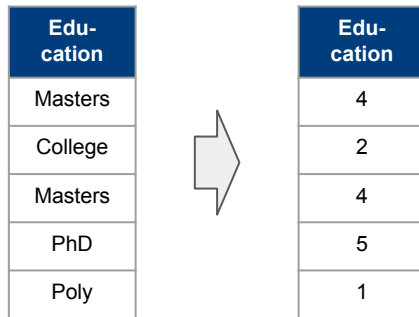# Caveat 3: Non-Numerical Data

- ## Discussion / Limitations

  - One-hot encoding increases dimensionality of data

  - Distance between ordinal values often not intuitive

    $$5 \text{ (PhD)} - 4 \text{ (Masters)} \overset{?}{=} 2 \text{ (College)} - \text{Poly (1)}$$

  - Typically requires data normalization
    (particularly when combined with numerical features)


- ## Alternative approach: Custom distance metric

  - Design metric that appropriately quantifies distance between categorical values

  - Typically very specific to given dataset and application context.

| Edu-cation |
| --- |
| Masters |
| College |
| Masters |
| PhD |
| Poly |

| Edu-cation |
| --- |
| 4 |
| 2 |
| 4 |
| 5 |
| 1 |

# Caveat 4: Semantic vs. Low-Level Similarity

- **Unstructured data**
  (text, image, video, audio)
  - Object is often more than just the sum of its parts

- **Example** (dog food ad)
  - Pixelwise similarity large; easy to compute
  - Semantic similarity small

# Supervised Training/Learning — Extended Setup

- Recall basic setup: training data + test data

- Building a good classifier or regressor
  - Find the best data preprocessing steps
  - Find the best model (model selection)          → Iterative evaluation required
  - Find the best hyperparameter values

- **Important:** Not allowed to use test data for this!
  - Test data is supposed to contain truly unseen data
  - Test data no longer unseen when used to optimize/tune model
    (for example, results might be different for a different training-test split)

# Training & Evaluation Process Using Validation Data

- Common data split to ensure generalizability of results
  - Use test data only at the very end to evaluate performance of final model
  - Use validation data for model selection and hyperparameter tuning
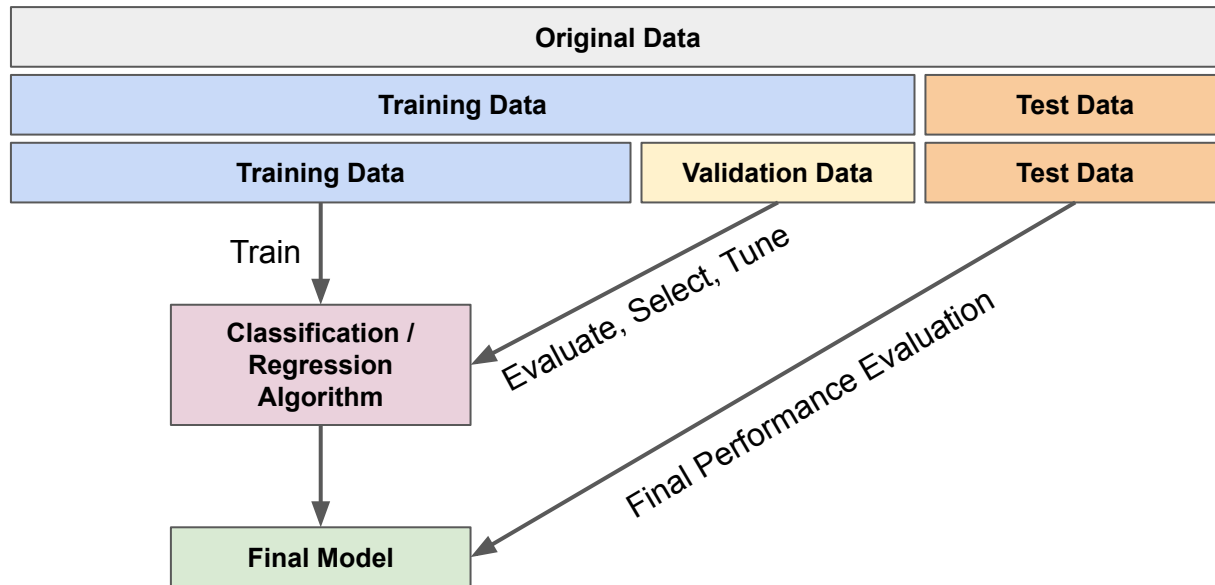
# K-Fold Cross Validation (just one of many validation techniques)

- Core idea of *k*-fold cross validation
  - Split training data into k blocks of equal size (e.g., *k*=10)
  - Use (*k*-1) blocks for training and remaining block for evaluation
  - Repeat for *k* rounds with different permutations

| | Round 1 | Round 2 | Round 3 | Round 10 |
|---|---|---|---|---|
| Validation Score: | 0.93 | 0.90 | 0.91 | 0.95 |
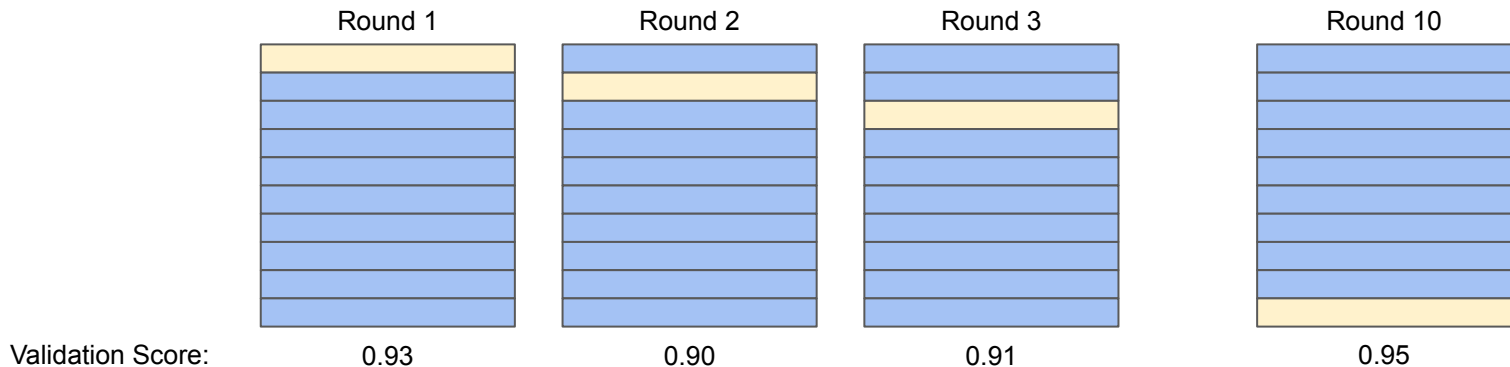
- Advantages
  - Averaged results more reliable
  - Variance between results useful indicator

# Information Leakage through Normalization

- Important guidelines
  - Do not normalize before splitting dataset into training and test data!
  - Normalize training and test data but only based on training data!

  **Why?**

- Example (pseudo) code using scikit-learn

```python
# Split input data into training and test set
X_train, X_test = split(X, 0.2)

# Fit StandardScaler (i.e., calculate mean and variance)
scaler = preprocessing.StandardScaler().fit(X_train)   # CORRECT!
#scaler = preprocessing.StandardScaler().fit(X)         # WRONG!!!

# Fit both training and test data
X_train_transformed = scaler.transform(X_train)
X_test_transformed = scaler.transform(X_test)
```

# Summary

- ## Evaluation of classifiers (straightforward for regressors)
  - Different metric with different interpretations

  - Applicability of metrics depending on data and task


- ## K-Nearest Neighbor (KNN) classifier/regressor
  - Very intuitive supervised learning model

  - Limited applicability for (very) large datasets
    (heavy lifting done during prediction time not the training time)

  - Choice of hyperparameter $k$ important to address
    risk of overfitting and underfitting

# Solutions to Quick Quizzes