Local Methods-I

CS771: Introduction to Machine Learning

Purushottam Kar

Please enroll on Piazza

http://tinyurl.com/ml17-18adf

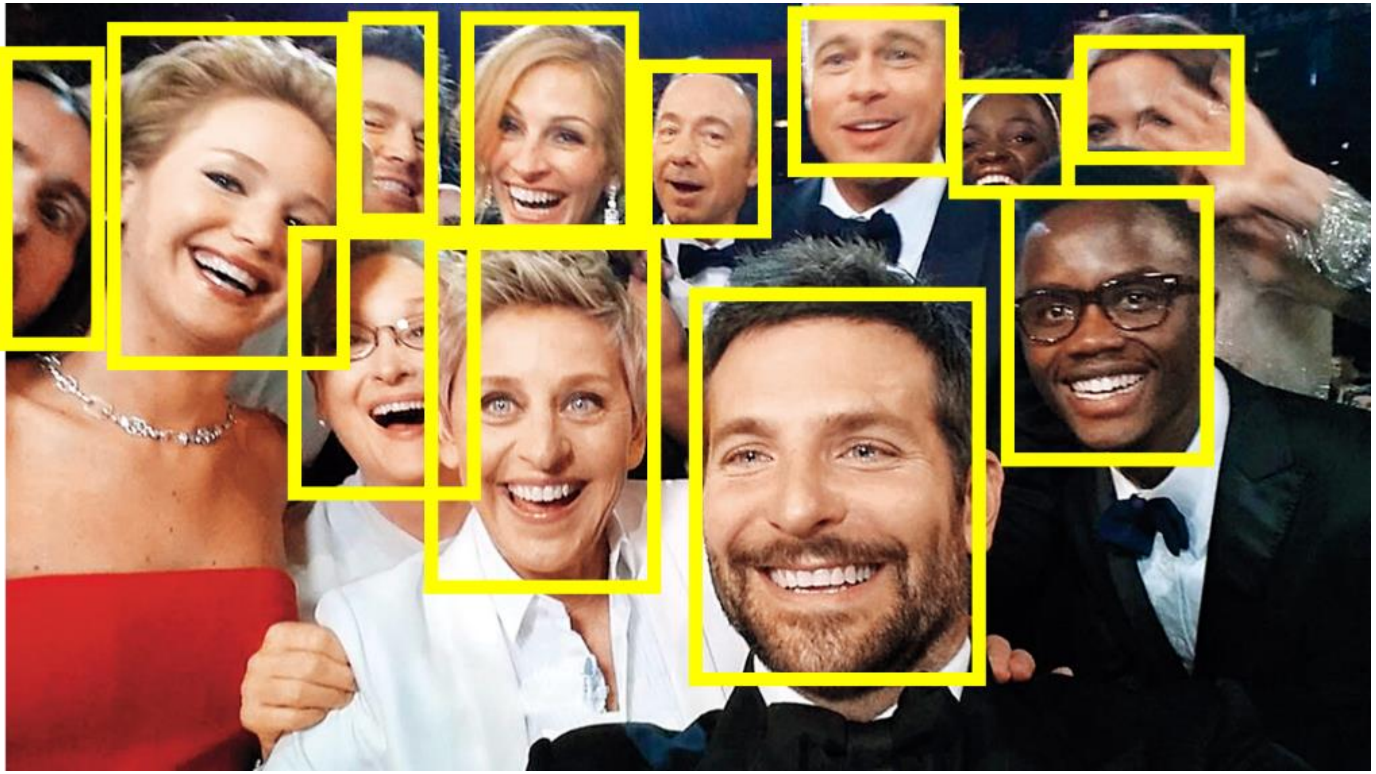
Internal Website up!

http://tinyurl.com/ml17-18ai

Enroll Project Groups

http://tinyurl.com/ml17-18apg

Recap: Problem Reductions in ML



Multi-Label Classification

Binary Classification

Multi-Classification

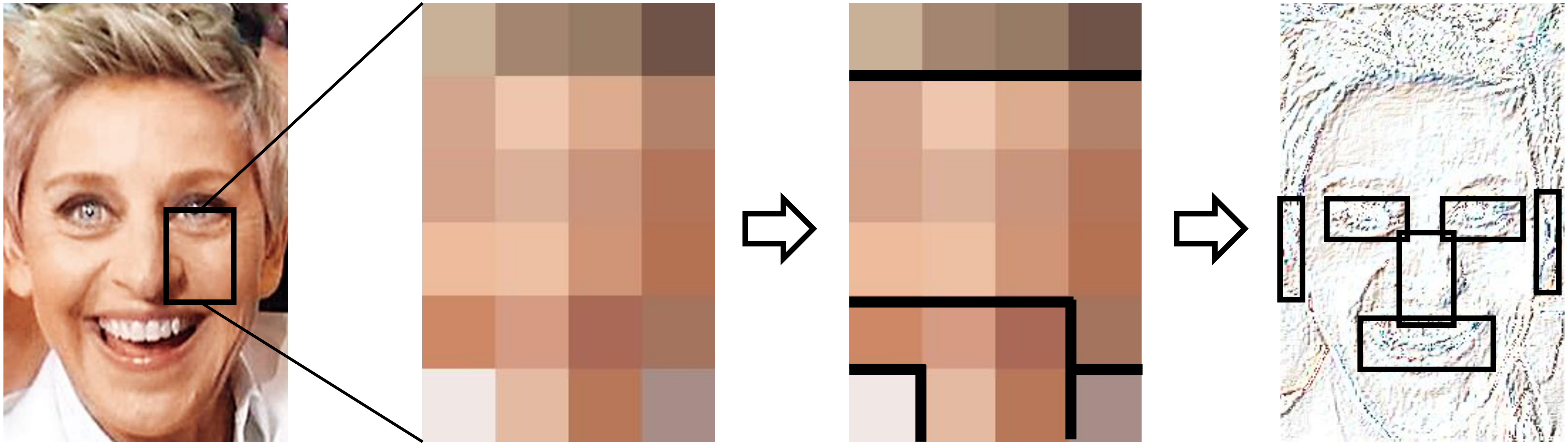
Regression

Ranking

Fantastic Features

... and how to find them

What is a feature?



Raw/ Low-level features

Derived/ High-level features

What is raw for you may have been derived by someone else

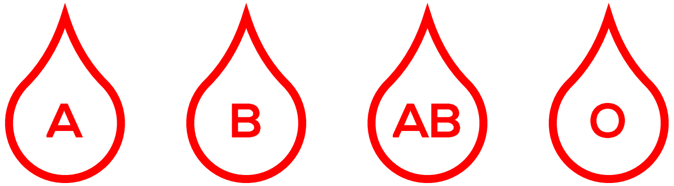
Types of Features

- Numerical Features

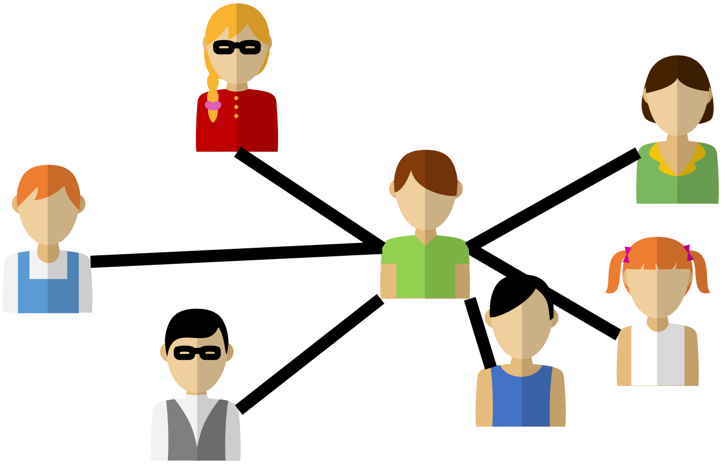
Dangal 

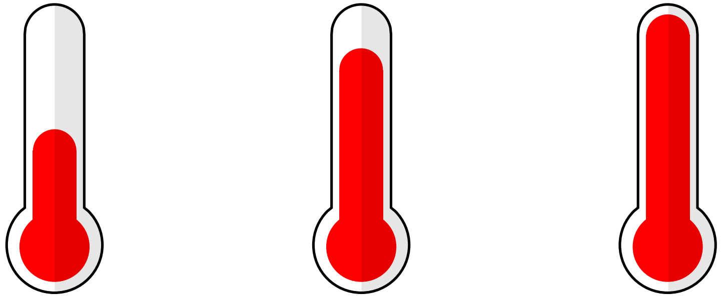
Dhoom 3 

- Categorical Features

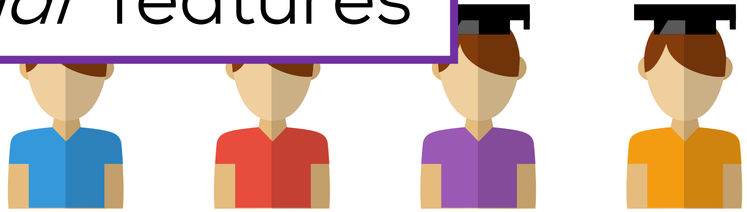


- Relational Features

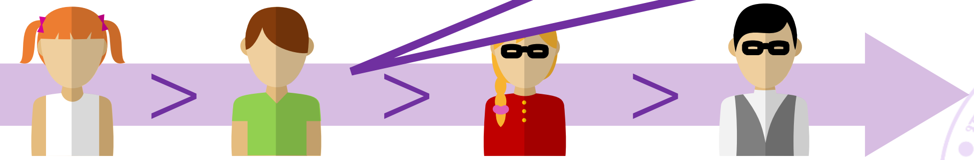




Ordinal features



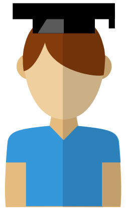
An ML problem for one person can be a feature for another



Academic Performance

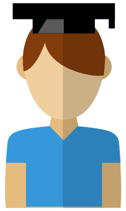
Derived Features

- Bagged/Binned Features



ESC101 (A), ESO207 (B), CS220 (B), CS340 (C), MSO201 (A), CS771 (A)

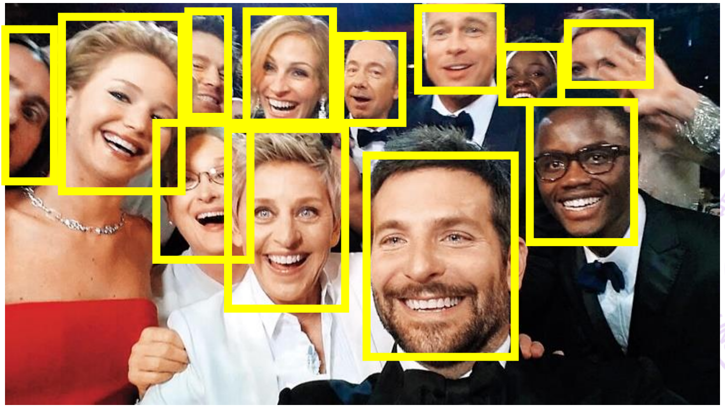
- Pooled/Aggregated Features



ESC101 (A), ESO207 (B), CS220 (B), CS340 (C), MSO201 (A), CS771 (A)

10 (max), 8.67 (avg)

- Latent Features



Feature Selection

Real-valued regression problem

- DIET (VEG/NON-VEG)
- EYE COLOR
- TOTAL INCOME
- EDUCATION LEVEL
- FAMILY SIZE
- HOUSE NO (ODD/EVEN)
- NO OF CHILDREN
- RENTED/SELF OWNED
- SURNAME LENGTH

(EXPENDITURE)

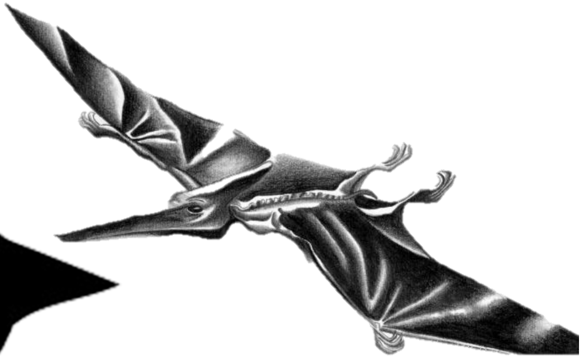
Challenge: Can we perform automated feature selection?

A little bit of Notation

- A data point (train/test) is represented as a column vector $\mathbf{x} \in \mathbb{R}^p$
- i-th coordinate of a vector $\mathbf{x}_i$
- The label of a data point (train/test) is represented as $y \in \mathbb{R}$
- Dataset (labelled) $(\mathbf{x}^1, y^1), (\mathbf{x}^2, y^2), \ldots, (\mathbf{x}^n, y^n)$
- Binary classification $y \in \{0, 1\}$
- Multi-classification $y \in \{1, 2, 3, \ldots, C\} \overset{\text{not}}{=} [C]$
- Multi-label classification $y \in 2^{[C]}$
- Real-valued regression $y \in \mathbb{R}$
- Vector-valued regression $\mathbf{y} \in \mathbb{R}^C$
- Ranking $\mathbf{y} \in \mathrm{Sym}([C]) \overset{\text{not}}{=} S_C$

Learning with Prototypes

Bird or Not!



Has wings

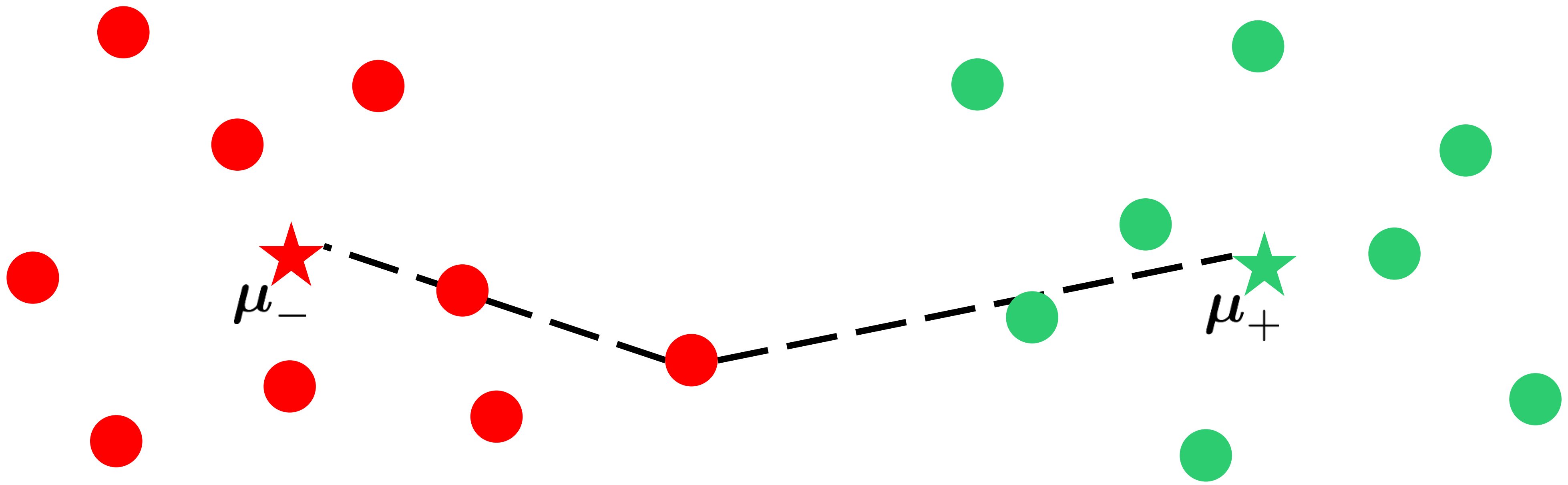
Can fly

Can sing

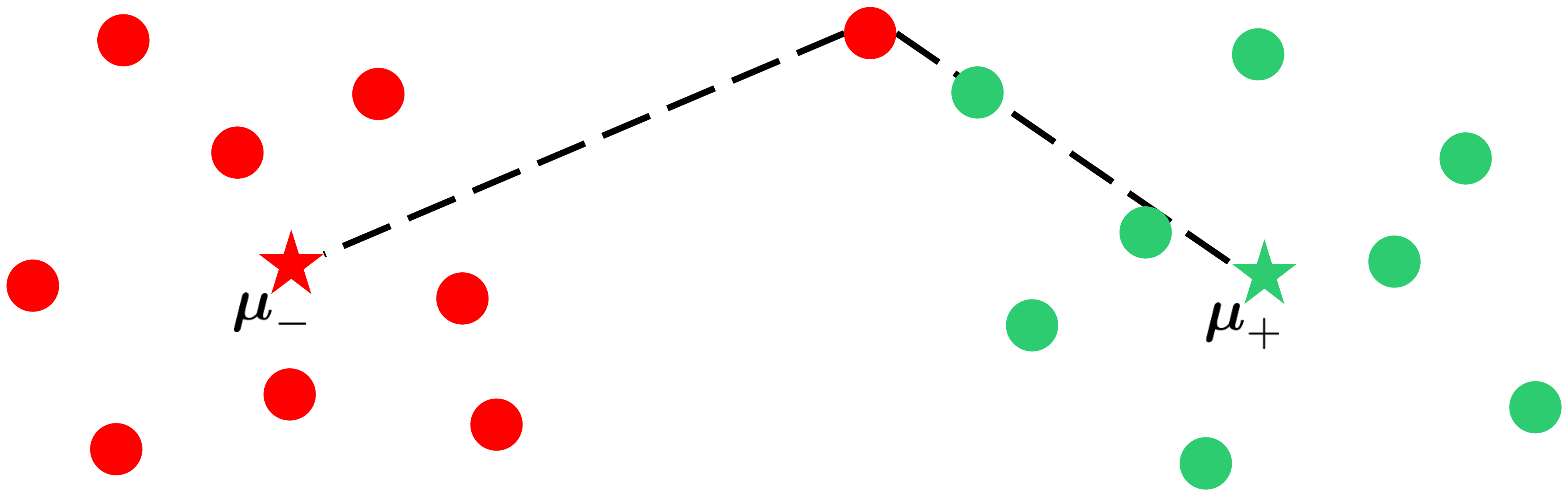
“Looks” like a bird

pixabay.com, independent.co.uk, youtube.com, newdinosaurs.com, wikipedia.com, clipartqueen.com

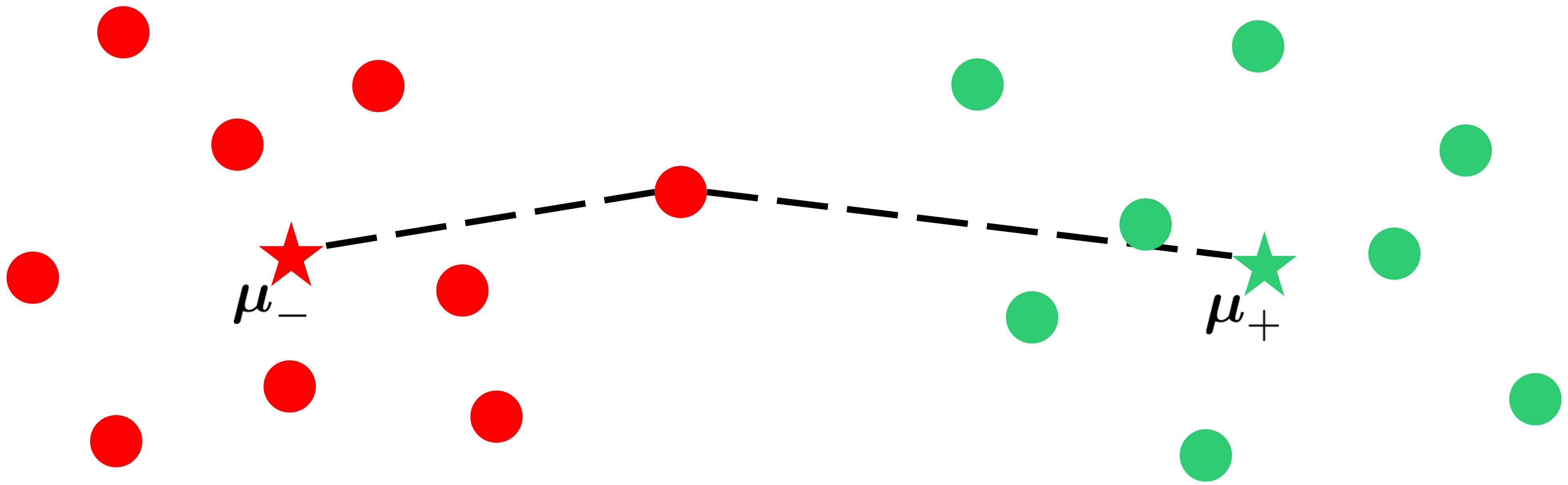
Learning with Prototypes



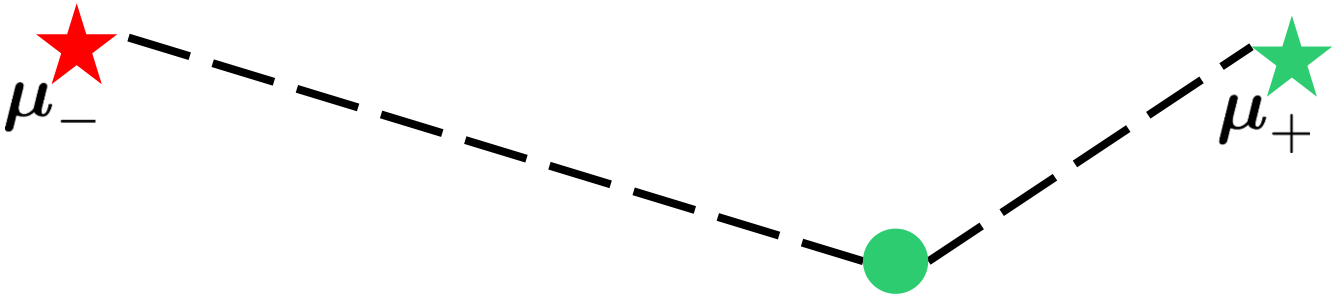
Learning with Prototypes

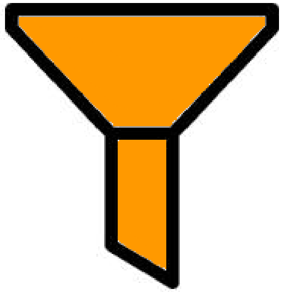


Learning with Prototypes



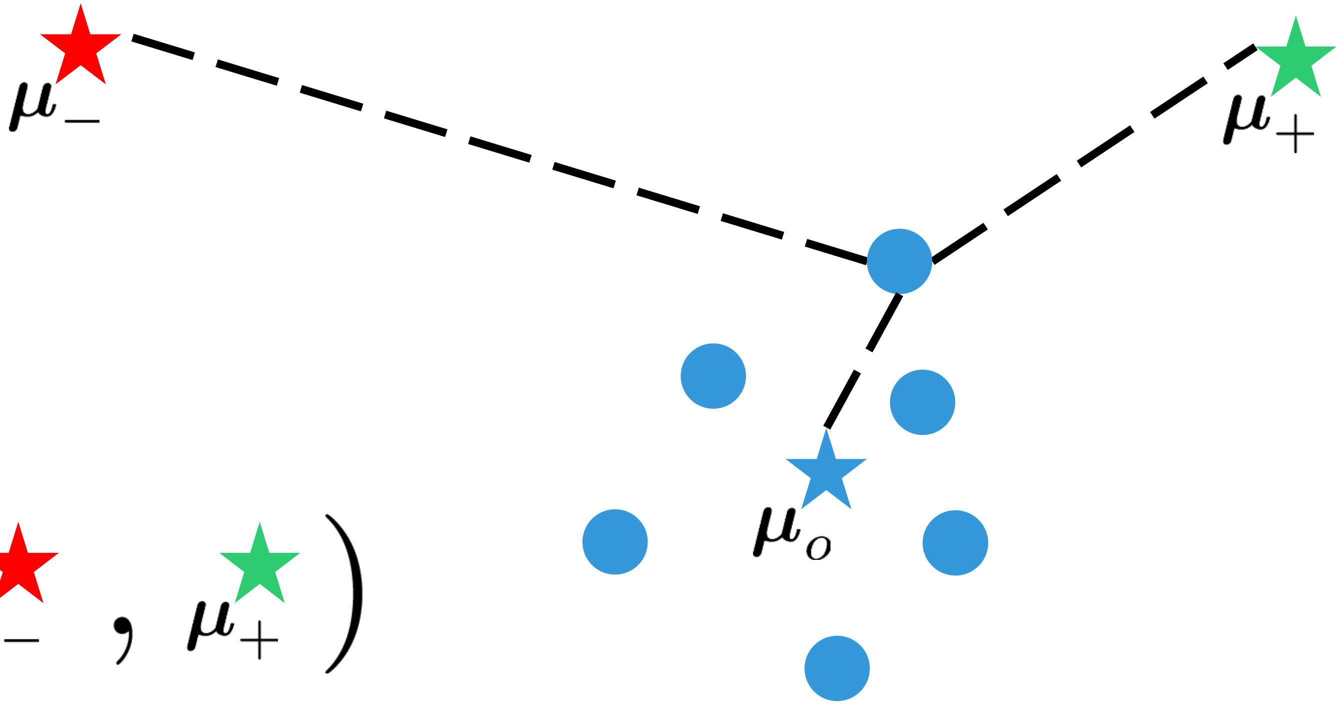
Learning with Prototypes

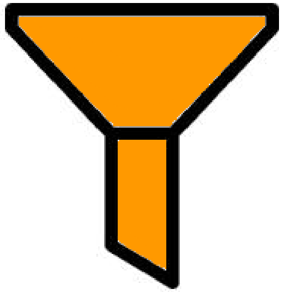




$$= (\mu_- , \mu_+)$$

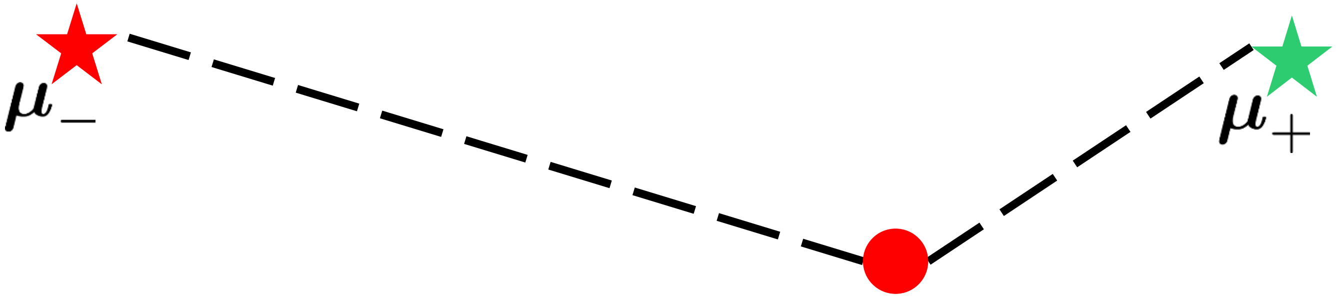
Learning with Prototypes

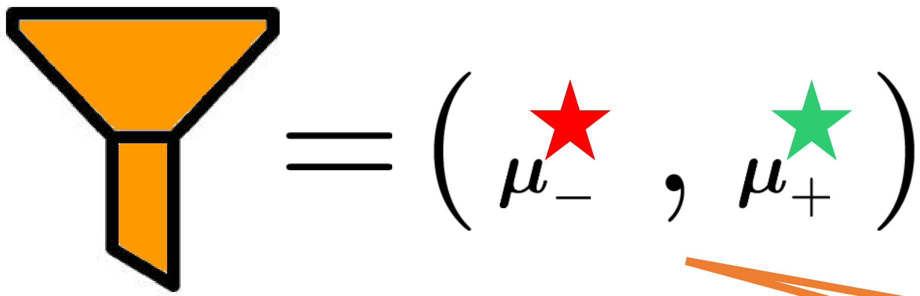




$$= (\mu_- \ , \ \mu_+)$$

Learning with Prototypes





- How to measure “closeness”?
- How to identify the prototypes?
- Why just one prototype per class?

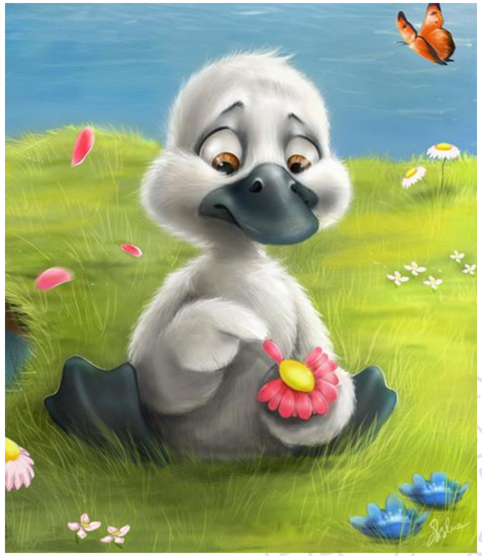
Parametric model

The Inductive Bias

> "Assumptions made by an ML algorithm on what aspects of the input data influence the desired label and how …"

- Underlies all decisions taken while building an ML algorithm
- Correct assumptions give models with good performances
- Incorrect assumptions may give poor models
- Impossible to completely eliminate inductive bias

The Ugly Duckling Theorem

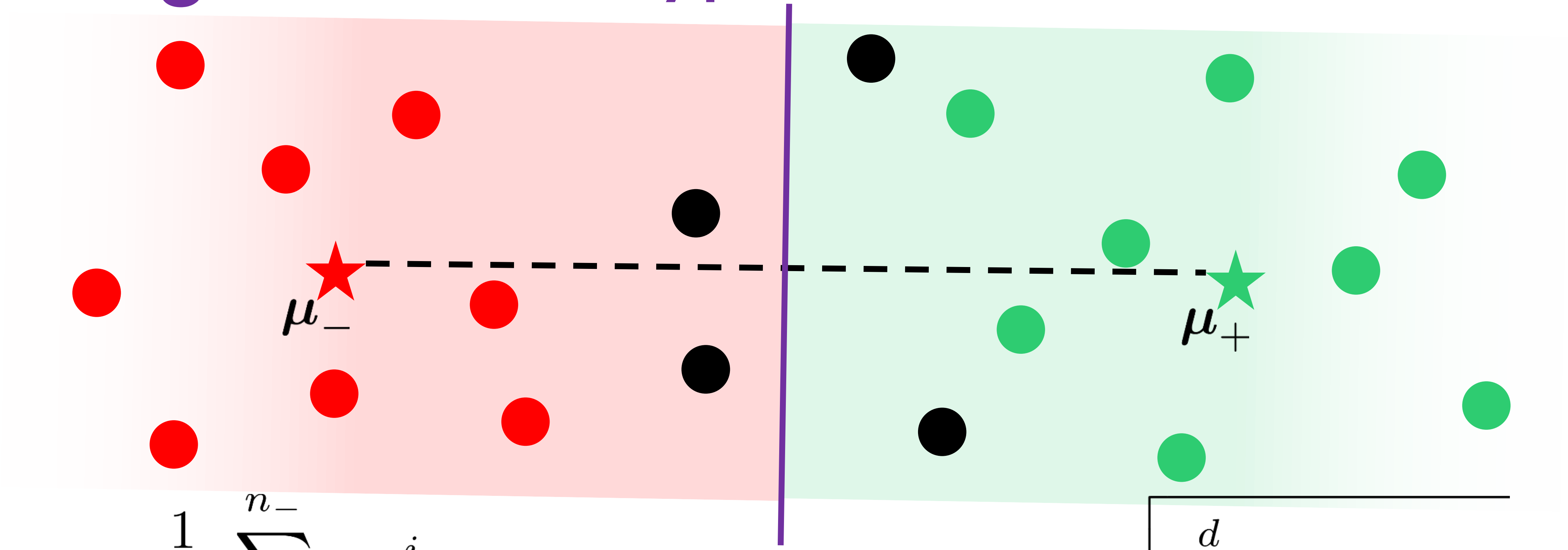


Metrics and Norms

- A metric on a set $\mathcal{X}$ is a bivariate function $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$
  - is symmetric $d(\mathbf{x}, \mathbf{y}) = d(\mathbf{y}, \mathbf{x})$
  - satisfies the triangle inequality $d(\mathbf{x}, \mathbf{y}) \leq d(\mathbf{y}, \mathbf{z}) + d(\mathbf{z}, \mathbf{x})$
  - separates discernibles $d(\mathbf{x}, \mathbf{y}) \neq 0 \Leftrightarrow \mathbf{x} \neq \mathbf{y}$
- A norm on a vector space V is a univariate function $\|\cdot\| : V \to \mathbb{R}$
  - separates discernible $\mathbf{v} \neq \mathbf{0} \Rightarrow \|\mathbf{v}\| \neq 0$
  - satisfies the triangle inequality $\|\mathbf{u} + \mathbf{v}\| \leq \|\mathbf{u}\| + \|\mathbf{v}\|$
  - satisfies positive homogeneity $\|a \cdot \mathbf{v}\| = |a| \cdot \|\mathbf{v}\|,\ \text{for } a \in \mathbb{R}$

Exercise: Show that all norms are non-negative valued.
Exercise: Show that every norm induces a metric $d(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|$.

Learning with Prototypes

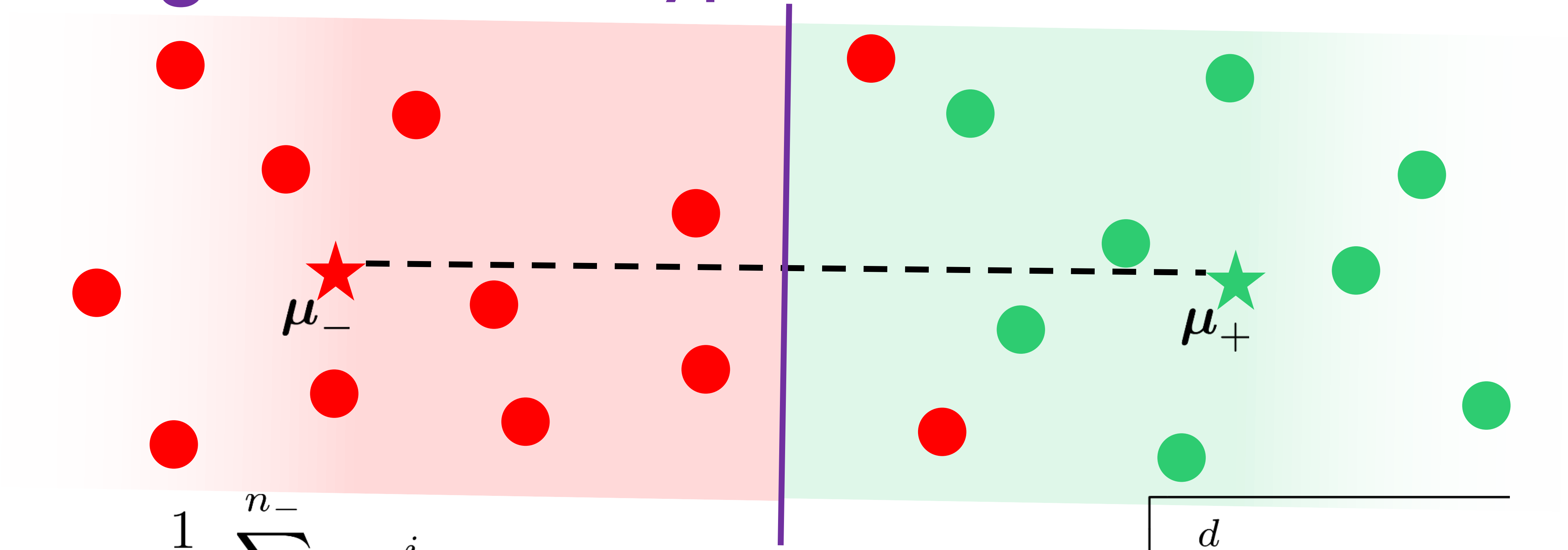


$$\boldsymbol{\mu}_- = \frac{1}{n_-}\sum_{i=1}^{n_-}\mathbf{x}^{-,i}$$

$$\boldsymbol{\mu}_+ = \frac{1}{n_+}\sum_{j=1}^{n_+}\mathbf{x}^{+,j}$$

$$d(\mathbf{x}, \boldsymbol{\mu}) = \|\mathbf{x} - \boldsymbol{\mu}\|_2 = \sqrt{\sum_{k=1}^{d}(\mathbf{x}_k - \boldsymbol{\mu}_k)^2}$$

Learning with Prototypes

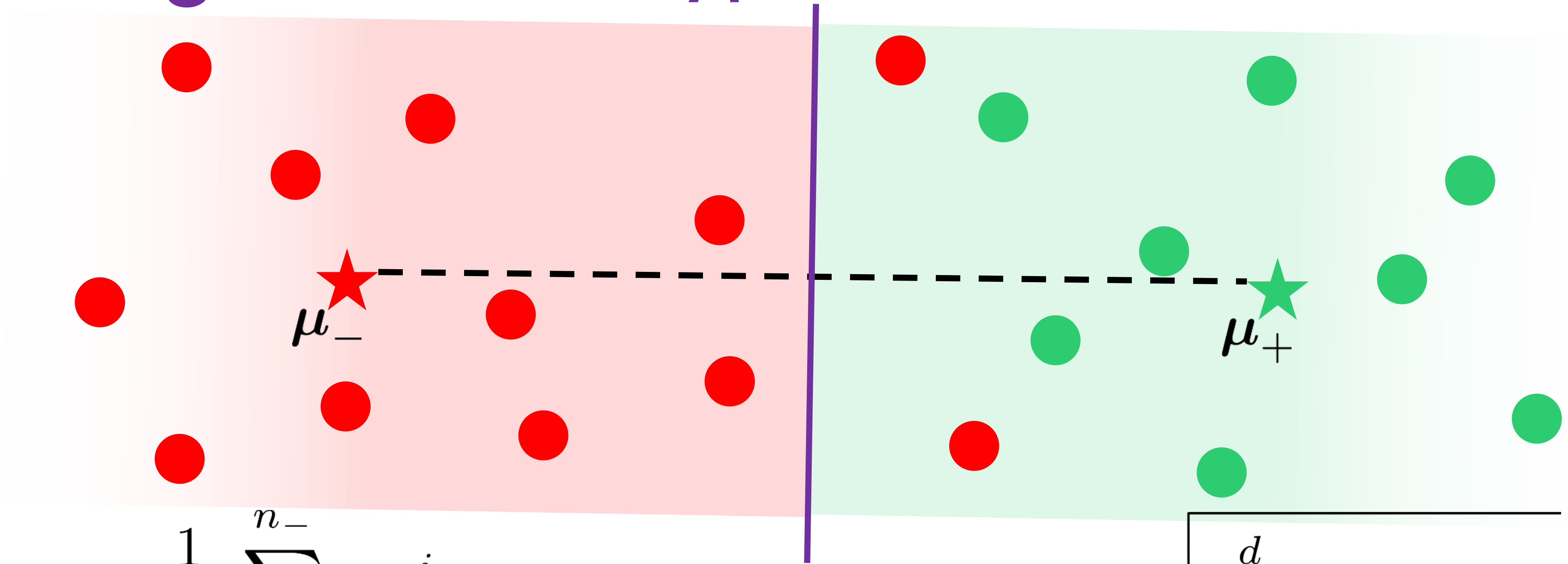


$$\boldsymbol{\mu}_- = \frac{1}{n_-}\sum_{i=1}^{n_-}\mathbf{x}^{-,i}$$

$$\boldsymbol{\mu}_+ = \frac{1}{n_+}\sum_{j=1}^{n_+}\mathbf{x}^{+,j}$$

$$d(\mathbf{x},\boldsymbol{\mu}) = \|\mathbf{x}-\boldsymbol{\mu}\|_2 = \sqrt{\sum_{k=1}^{d}(\mathbf{x}_k-\boldsymbol{\mu}_k)^2}$$

Learning with Prototypes



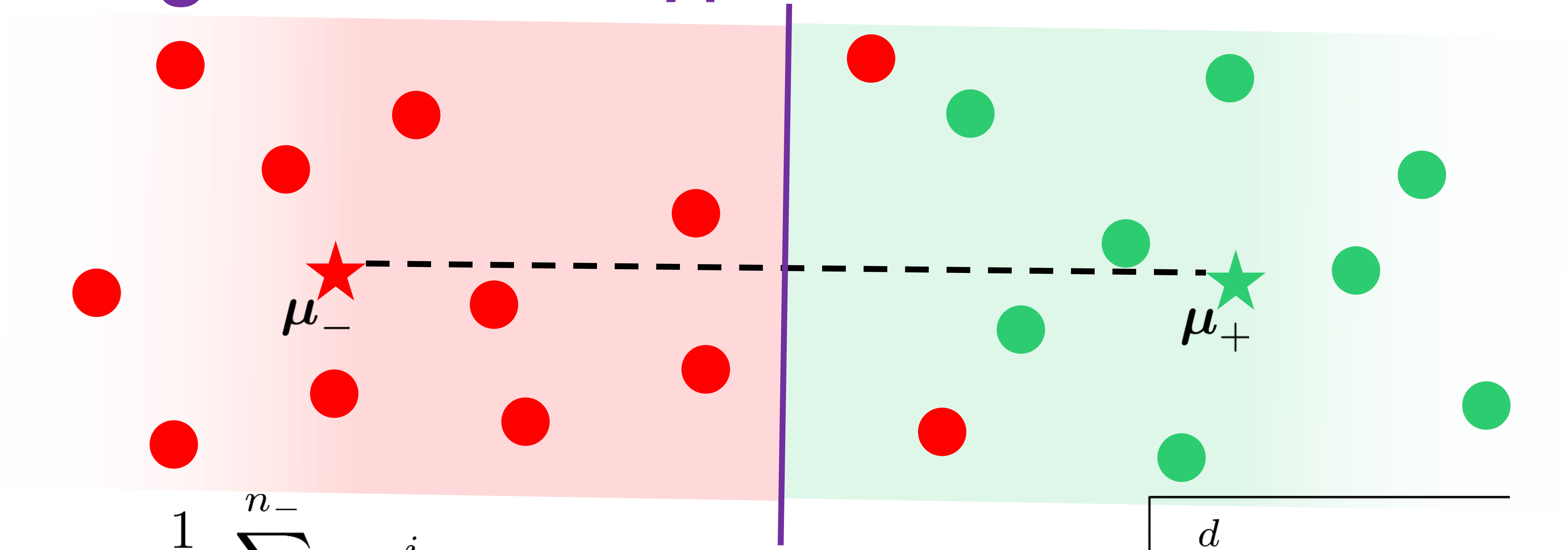
$$\boldsymbol{\mu}_- = \frac{1}{n_-}\sum_{i=1}^{n_-}\mathbf{x}^{-,i}$$

$$\boldsymbol{\mu}_+ = \frac{1}{n_+}\sum_{j=1}^{n_+}\mathbf{x}^{+,j}$$

$$d(\mathbf{x}, \boldsymbol{\mu}) = \|\mathbf{x} - \boldsymbol{\mu}\|_2 = \sqrt{\sum_{k=1}^{d}(\mathbf{x}_k - \boldsymbol{\mu}_k)^2}$$

Linear Decision Boundary

Learning with Prototypes



$$\boldsymbol{\mu}_- = \frac{1}{n_-}\sum_{i=1}^{n_-}\mathbf{x}^{-,i}$$

$$\boldsymbol{\mu}_+ = \frac{1}{n_+}\sum_{j=1}^{n_+}\mathbf{x}^{+,j}$$

$$d(\mathbf{x},\boldsymbol{\mu}) = \|\mathbf{x}-\boldsymbol{\mu}\|_2 = \sqrt{\sum_{k=1}^{d}(\mathbf{x}_k-\boldsymbol{\mu}_k)^2}$$

Linear Decision Boundary

$$f(\mathbf{x}) = \mathrm{sign}(\langle\mathbf{w},\mathbf{x}\rangle + b)$$

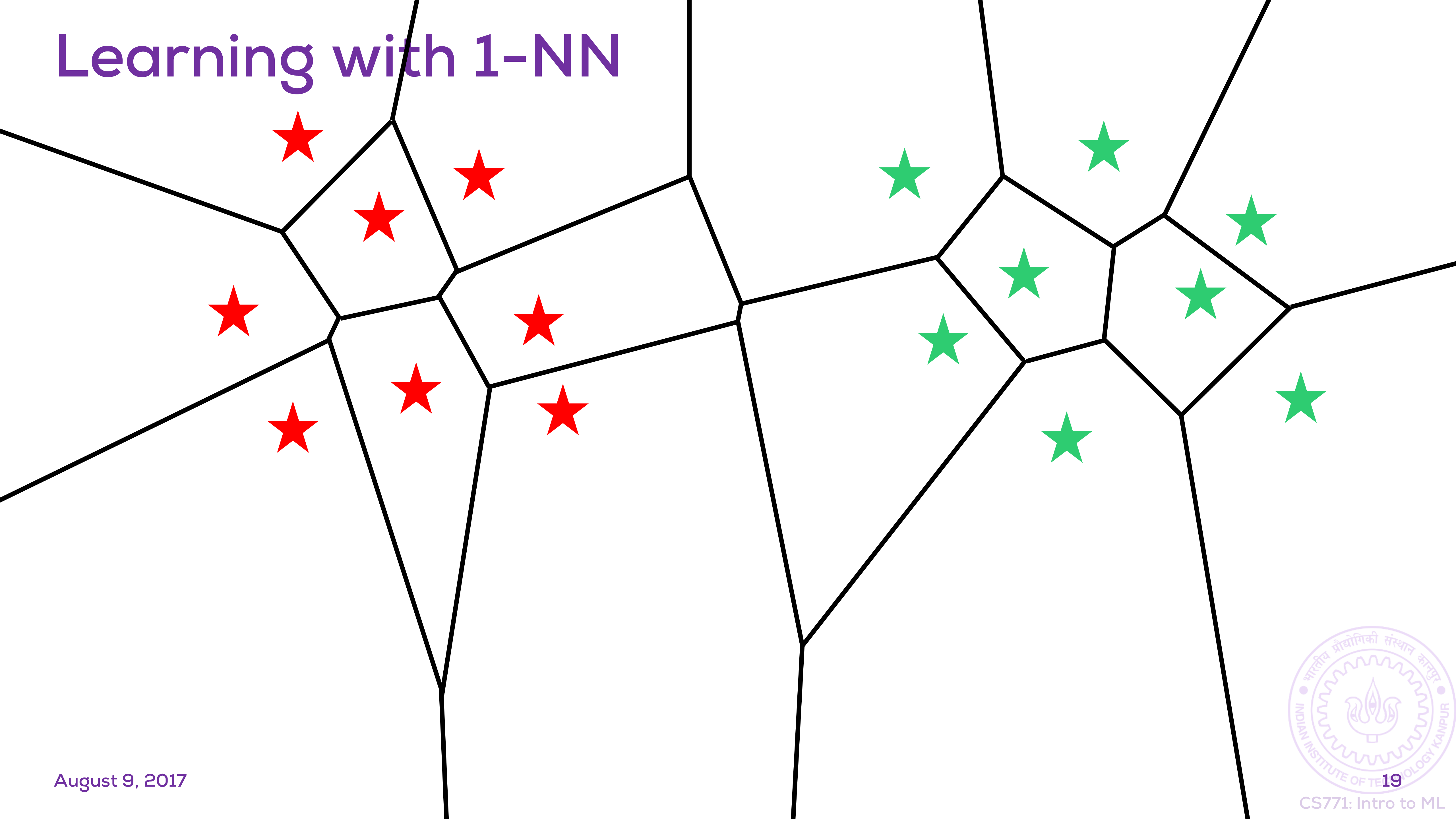
Exercise!

Learning with the Nearest Neighbors

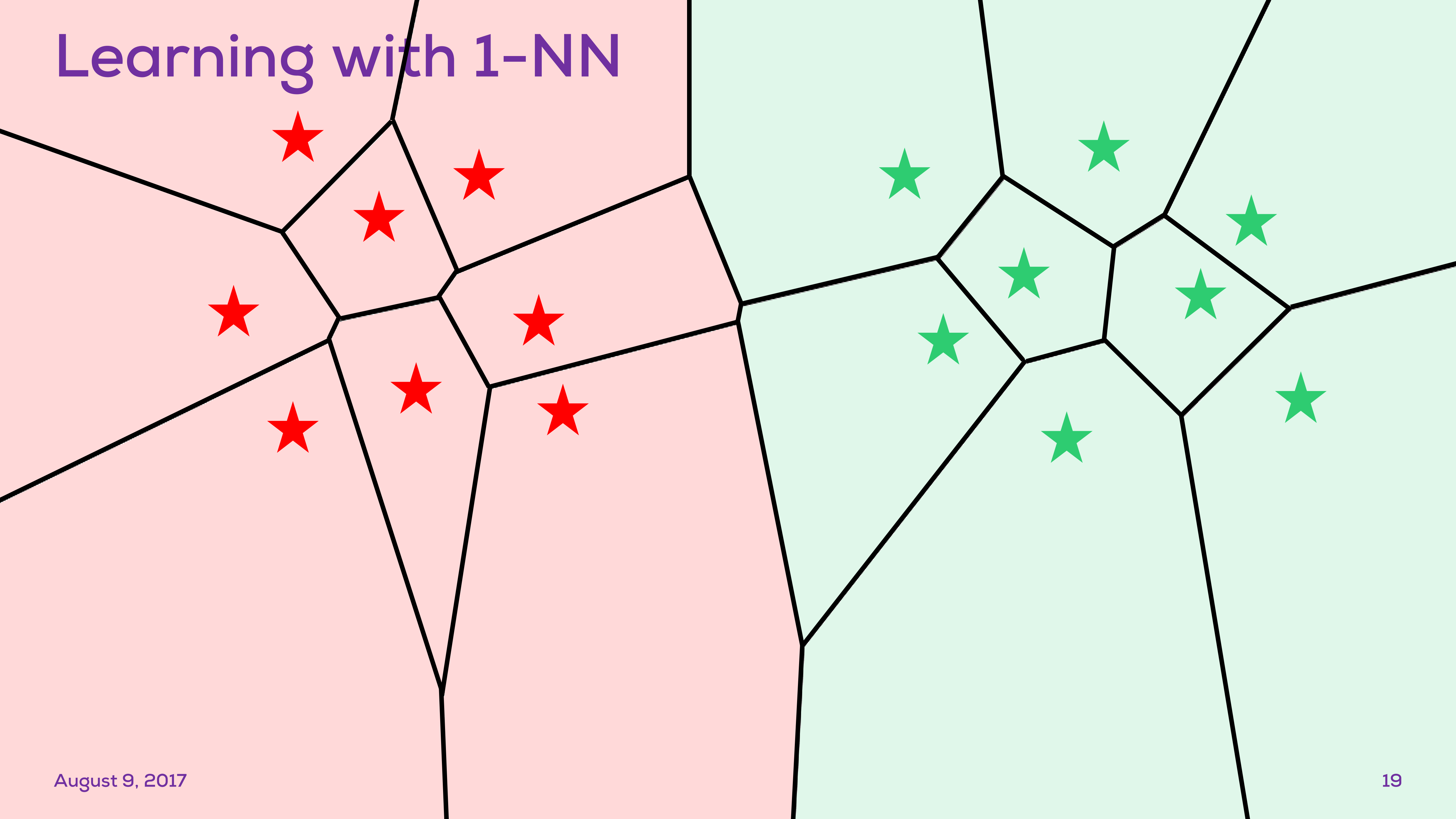
Learning with 1-NN

Learning with 1-NN

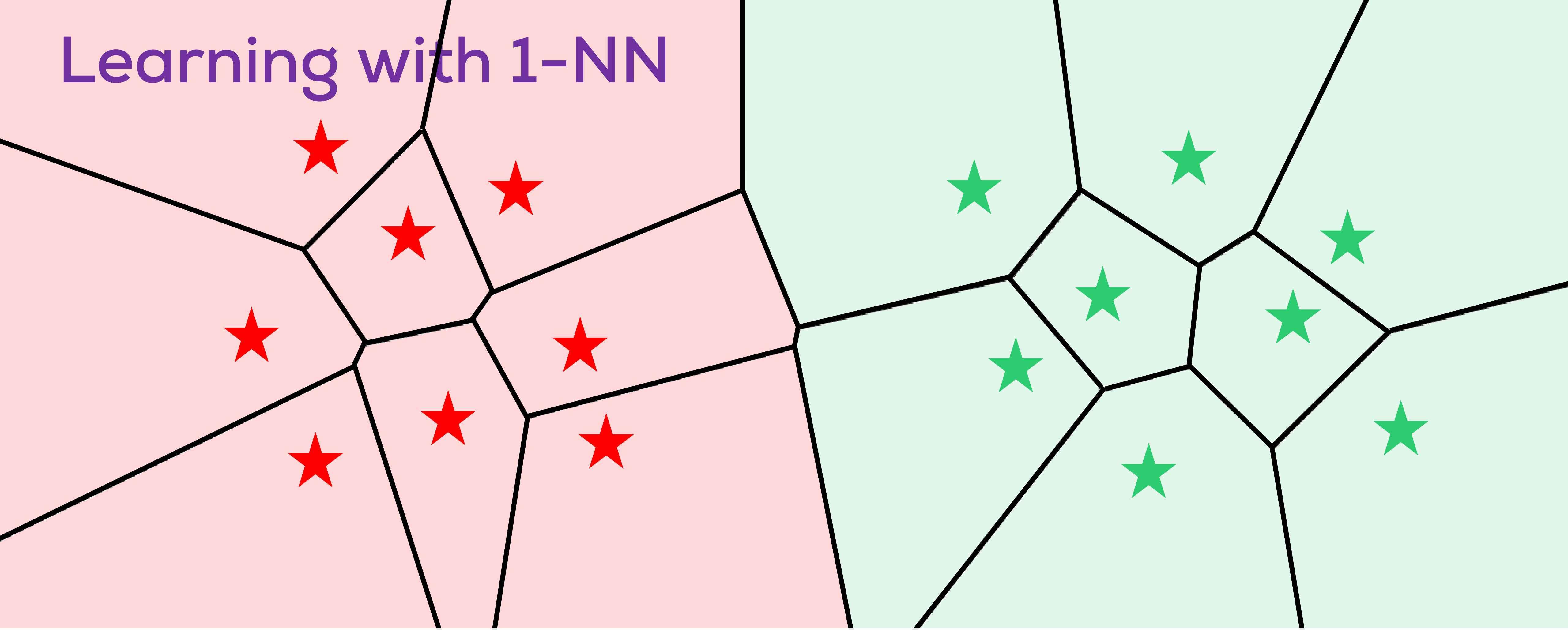
Learning with 1-NN



Learning with 1-NN



Learning with 1-NN



Voronoi Tesselization ⇒ Non-linear Decision Boundary

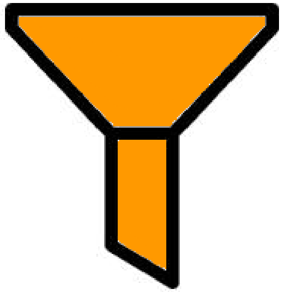
Learning with 1-NN

Advantages

- One of the oldest “learning” algorithms Fix and Hodges (1951)
- Also theoretically one of the “best” possible!
- In practice, performs really well given enough training data

Disadvatages

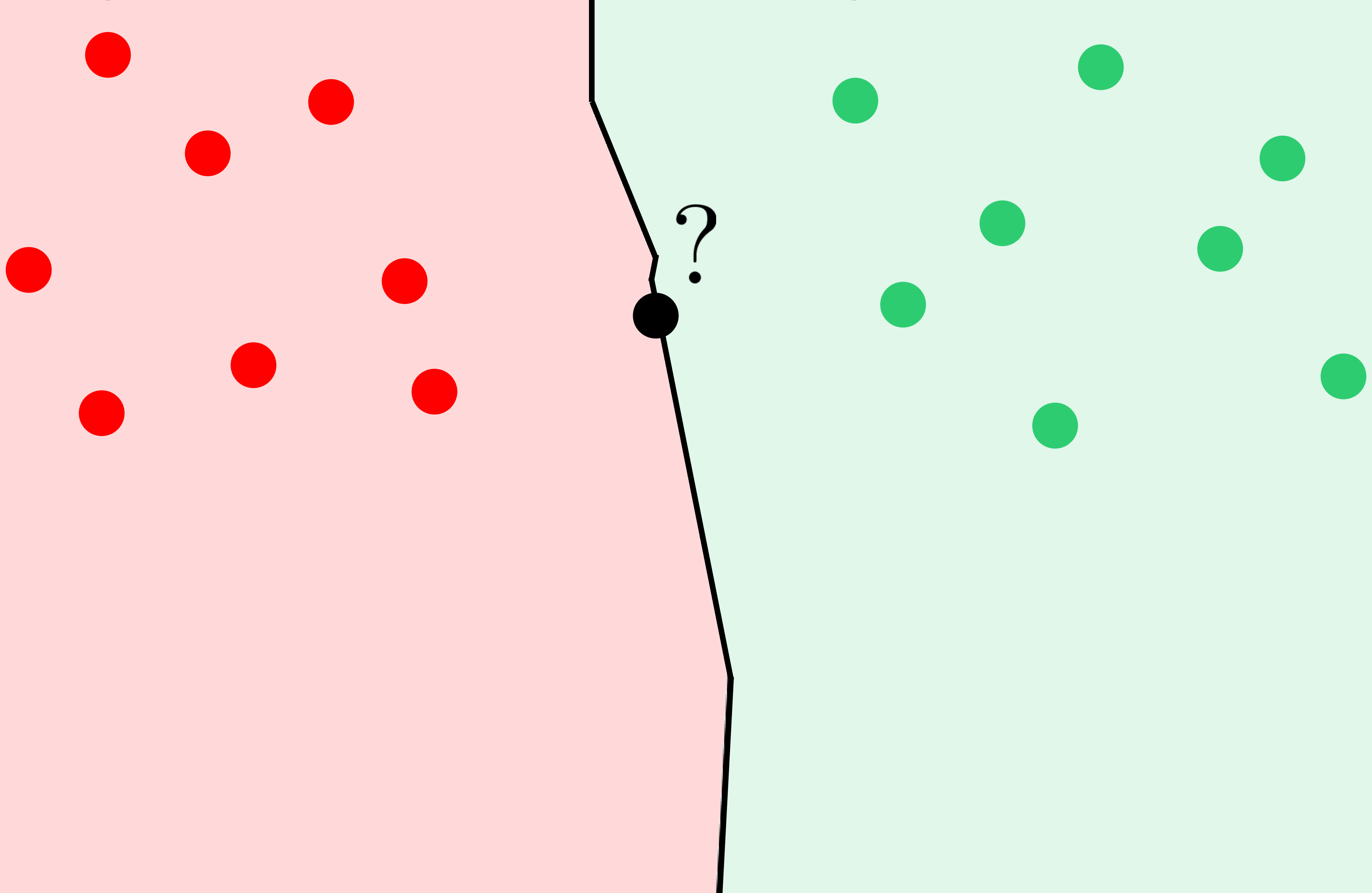
- Prone to noise, “overfits” the data
- Very expensive in terms of storage and computation



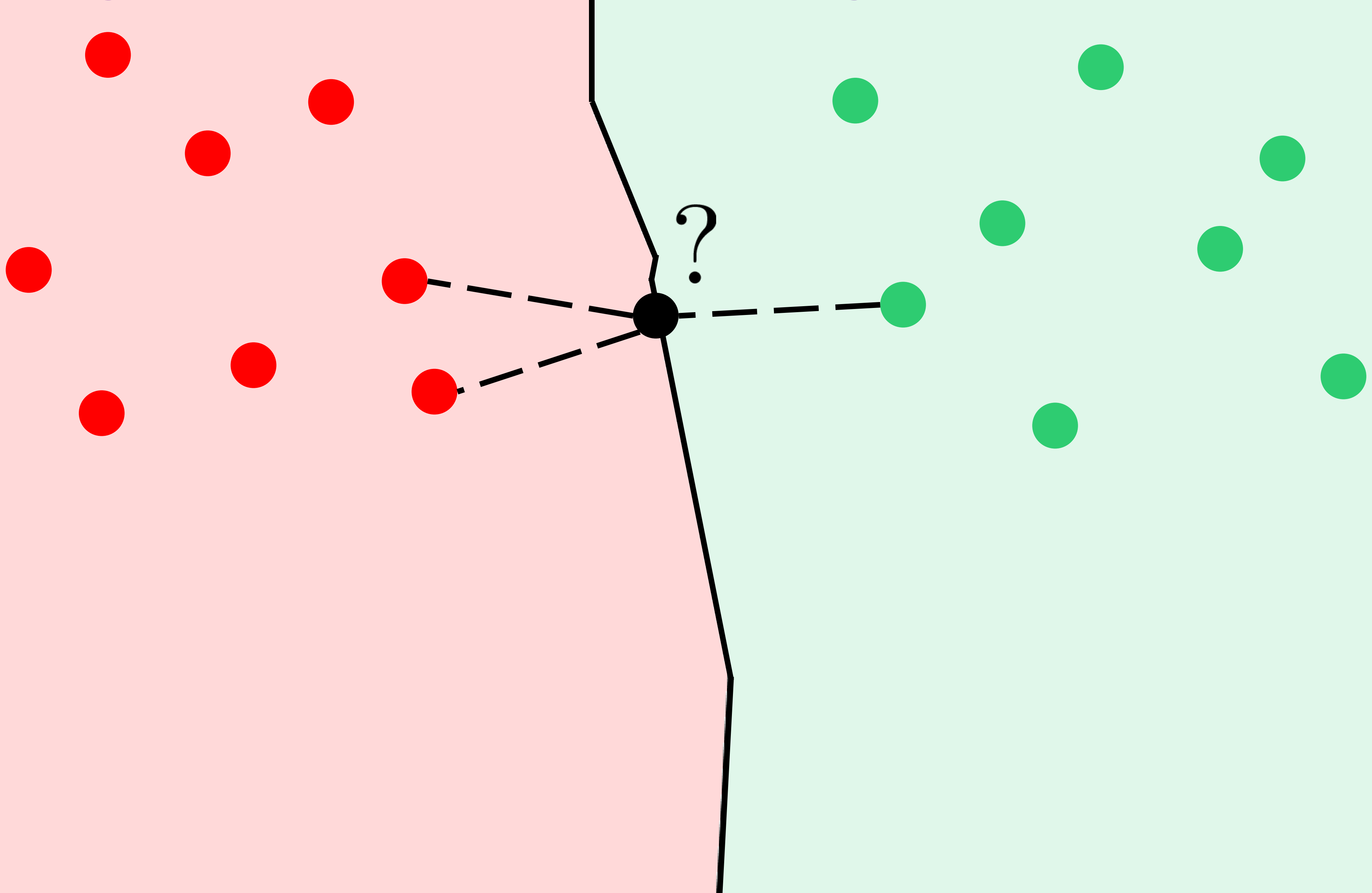
$= ?$

Non-parametric model

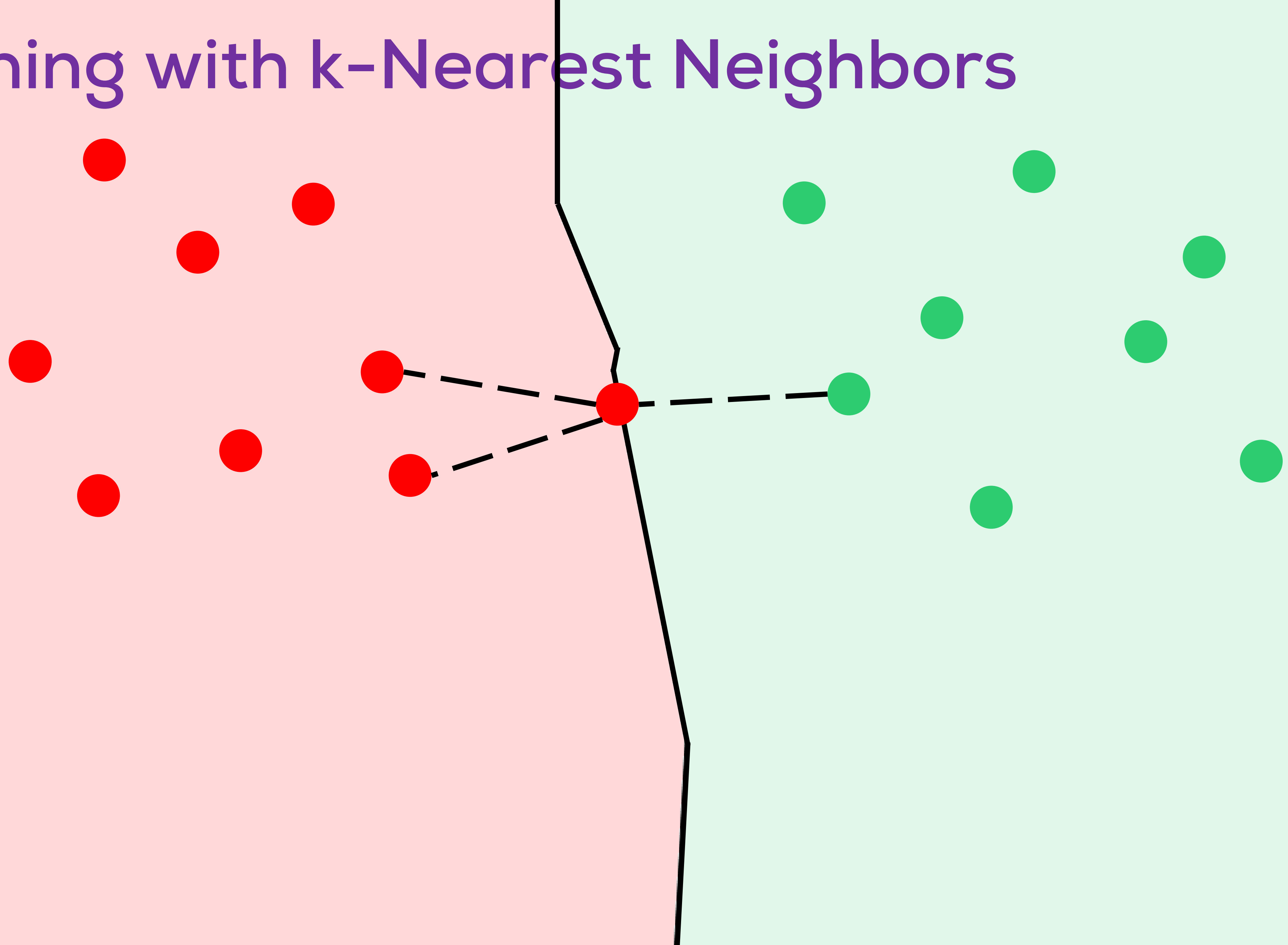
Learning with k-Nearest Neighbors



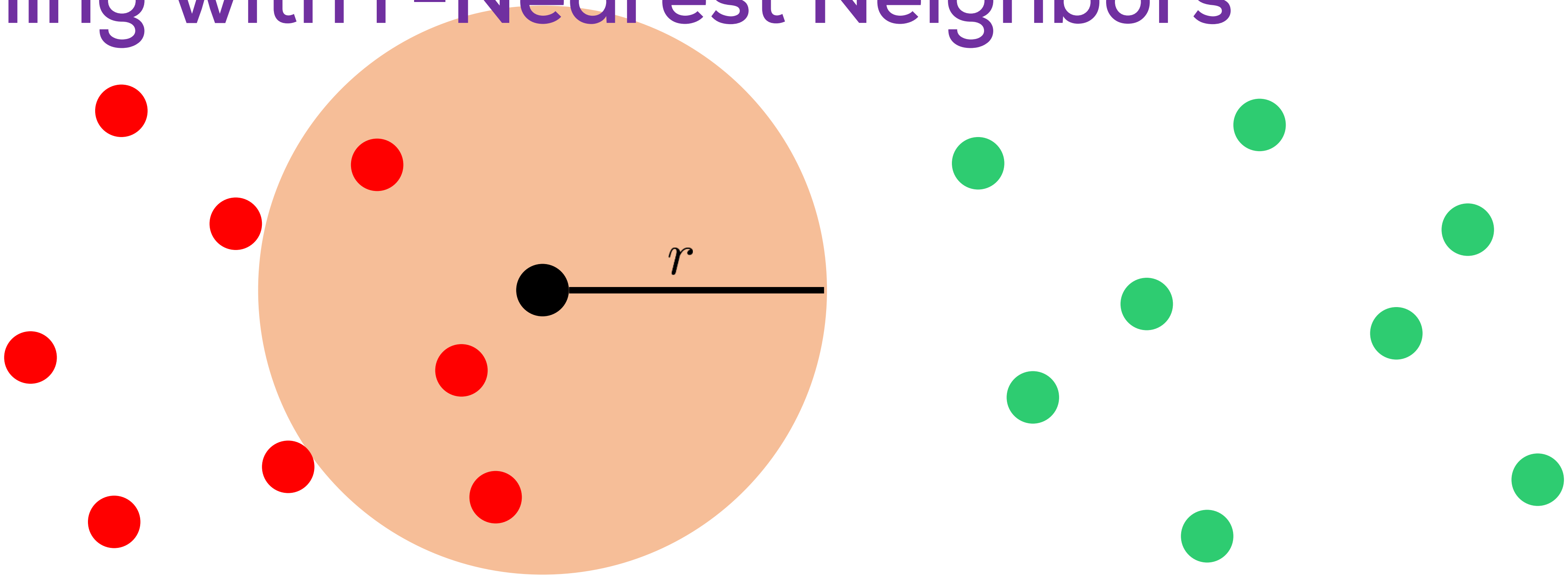
Learning with k-Nearest Neighbors



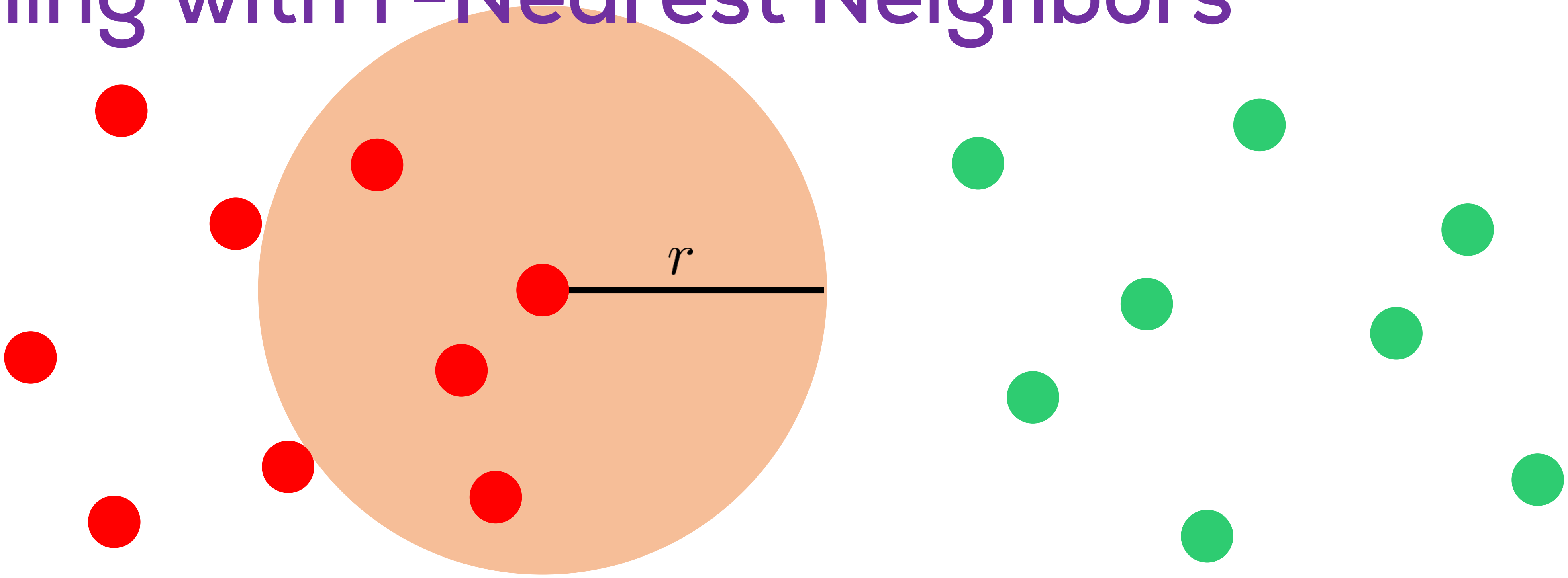
Learning with k-Nearest Neighbors



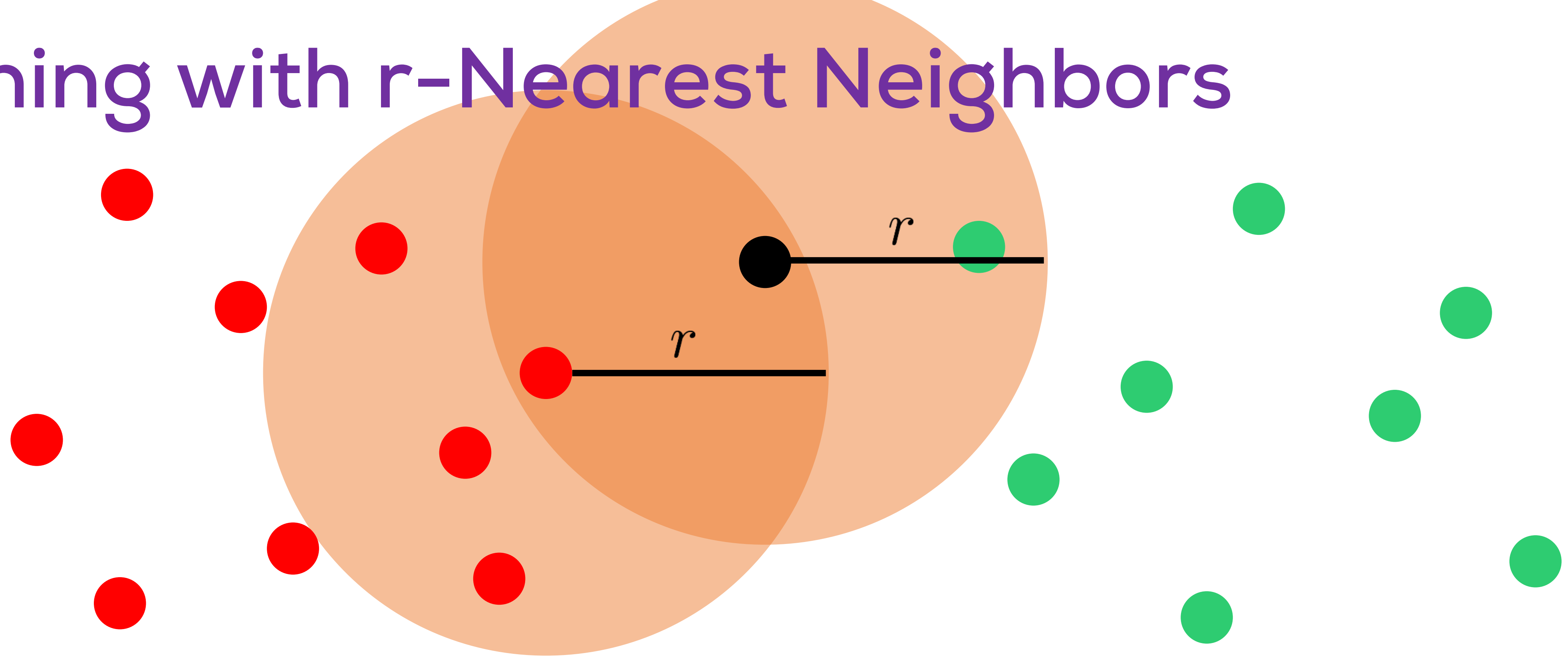
Learning with r-Nearest Neighbors



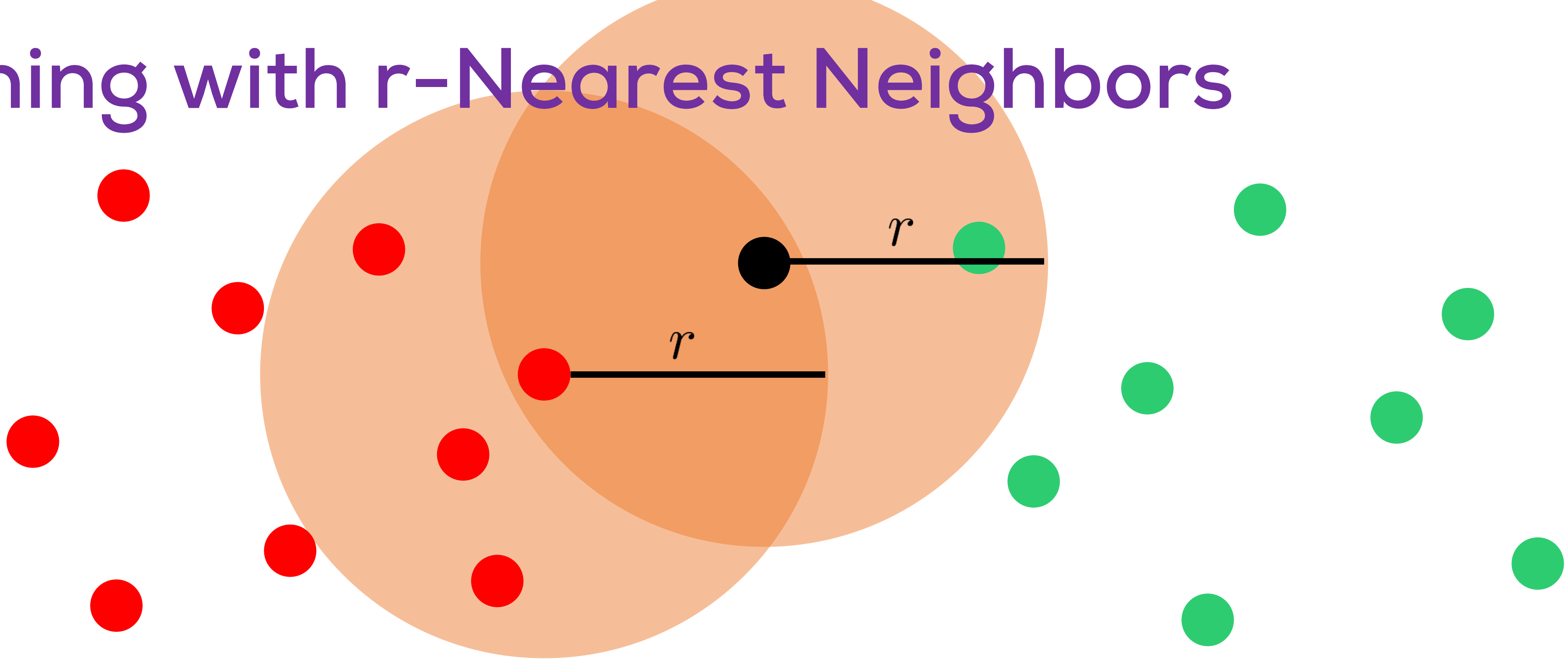
Learning with r-Nearest Neighbors



Learning with r-Nearest Neighbors



Learning with r-Nearest Neighbors

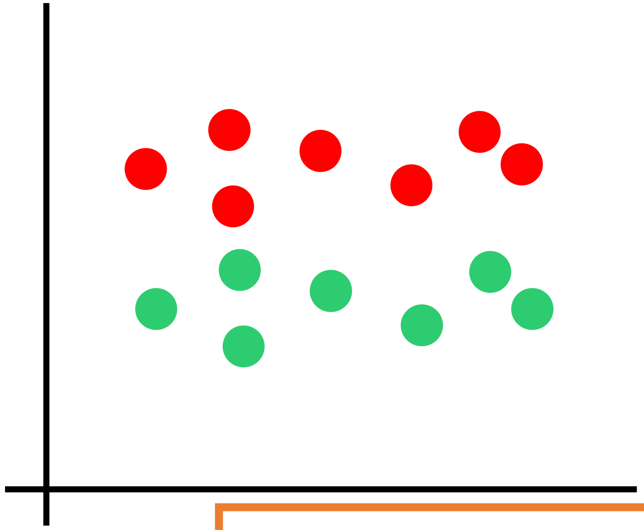


- k, r, metric are **hyper-parameters** of the algorithm
- Usually tuned using cross validation
- Can have *weighted* versions of k-NN, r-NN too!

Exercise: How will you extend this to multiclass problems?
Exercise: How will you set weights in weighted k-NN?

Some practical considerations

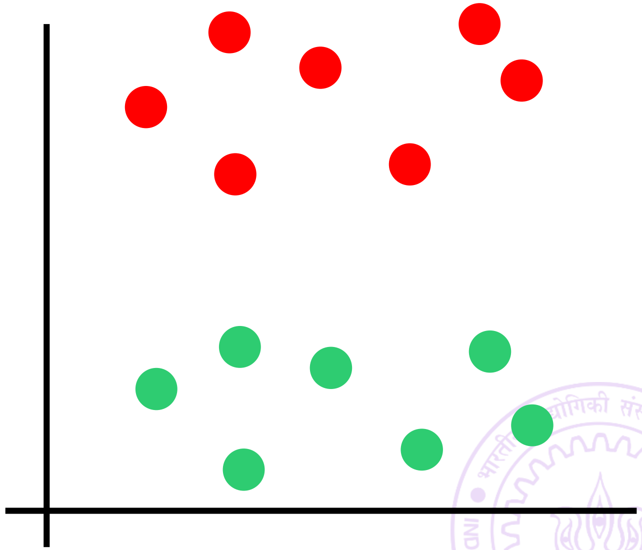
- Be very careful in choosing k (or r)
  - Large k makes partitions too smooth, may cause underfitting
  - Small k makes partitions too jumpy, may cause overfitting
- Be very careful in choosing the metric
  - Try to learn the metric itself in a task specific manner



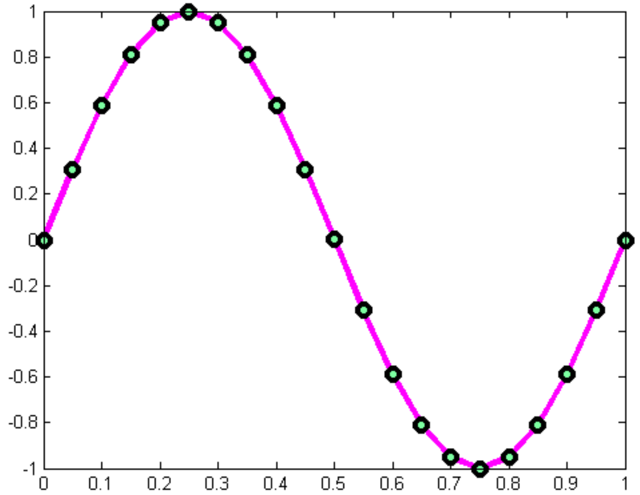
$$d(\mathbf{x}, \mathbf{y}) = (\mathbf{x} - \mathbf{y})^\top M (\mathbf{x} - \mathbf{y})$$

$$M = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}$$

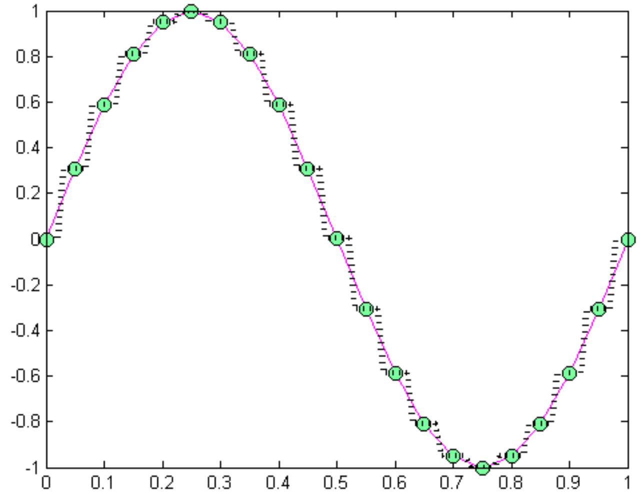
Mahalanobis metric



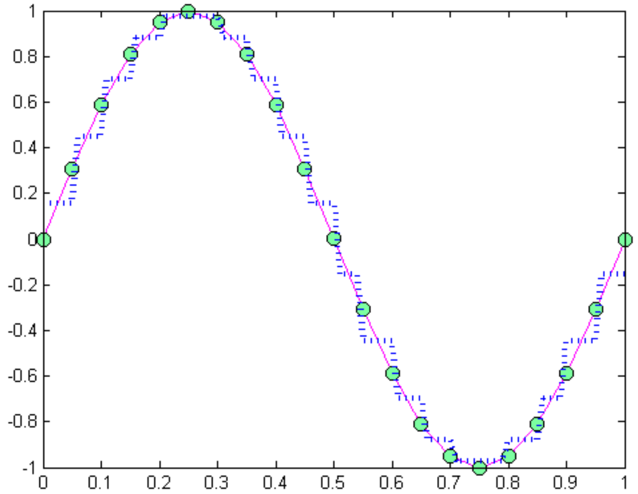
Regression using k-NN



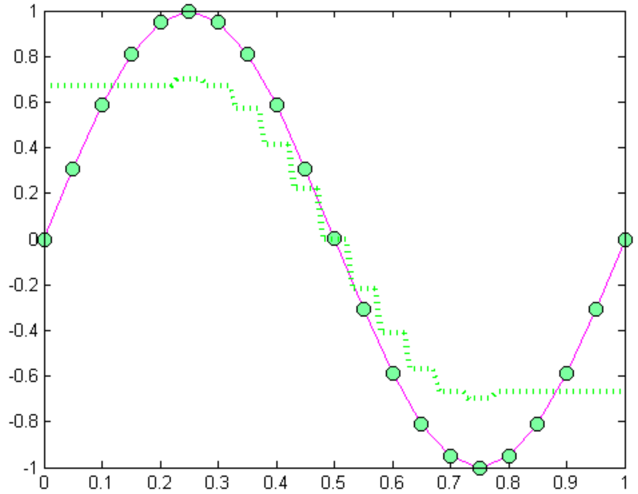
Training data



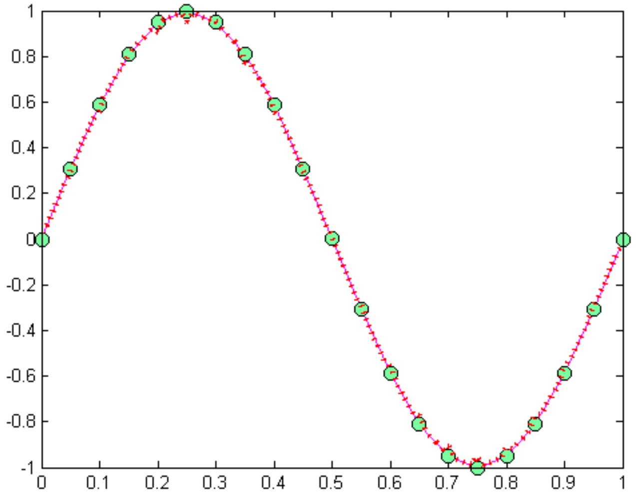
k=1



k=2

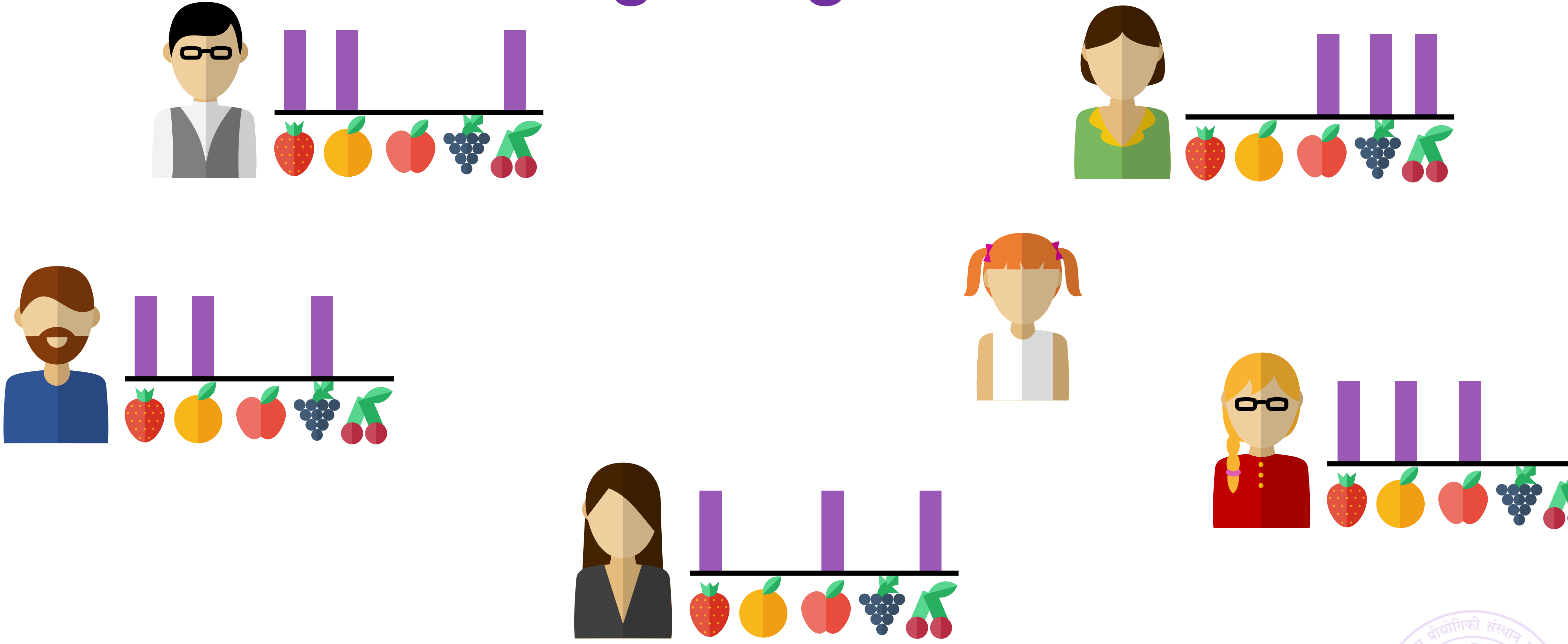


k=9

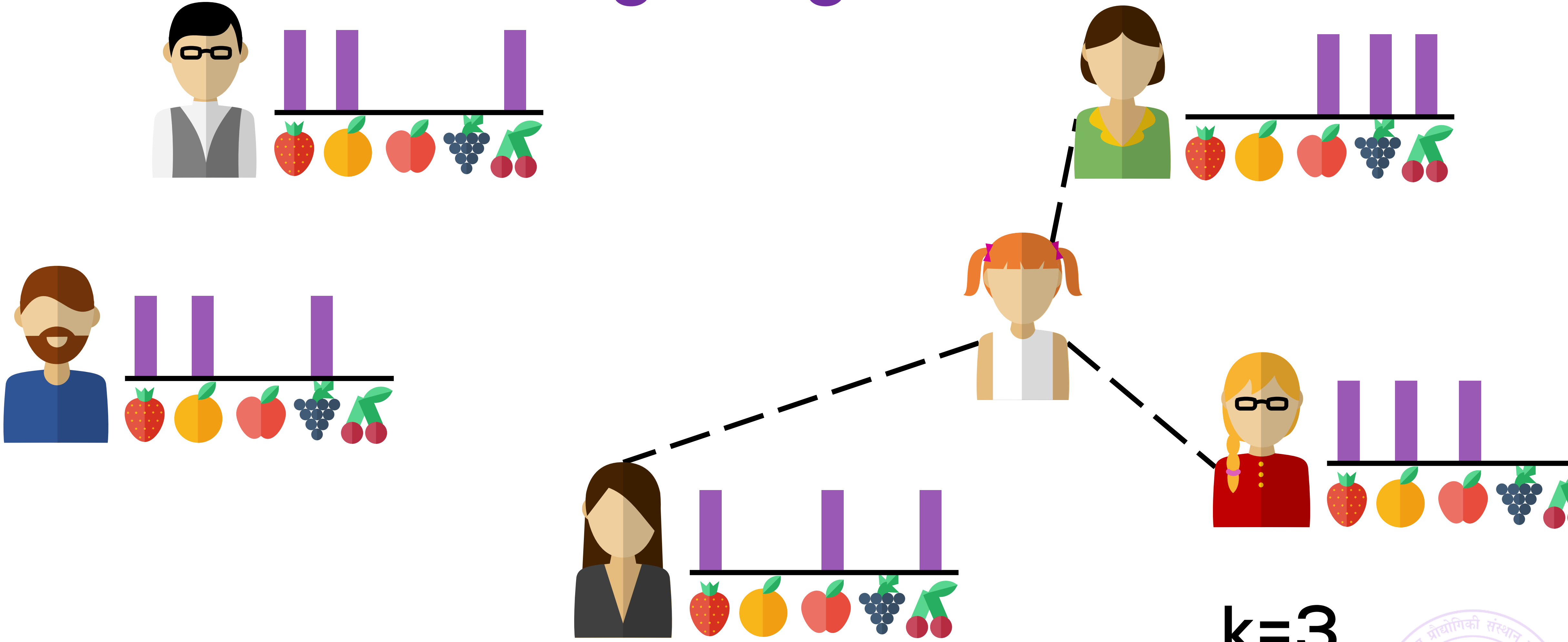


k=2 + weighted

Multi-label Learning using k-NN



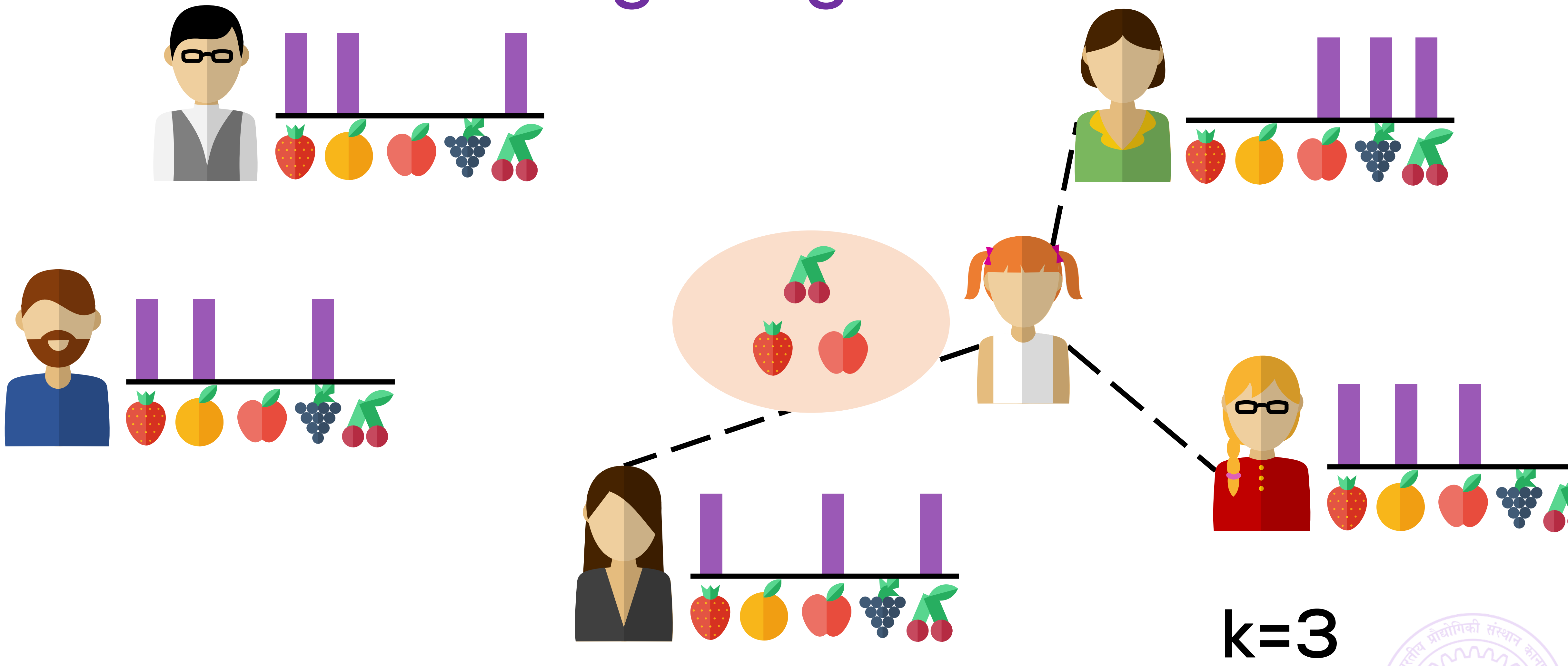
Multi-label Learning using k-NN



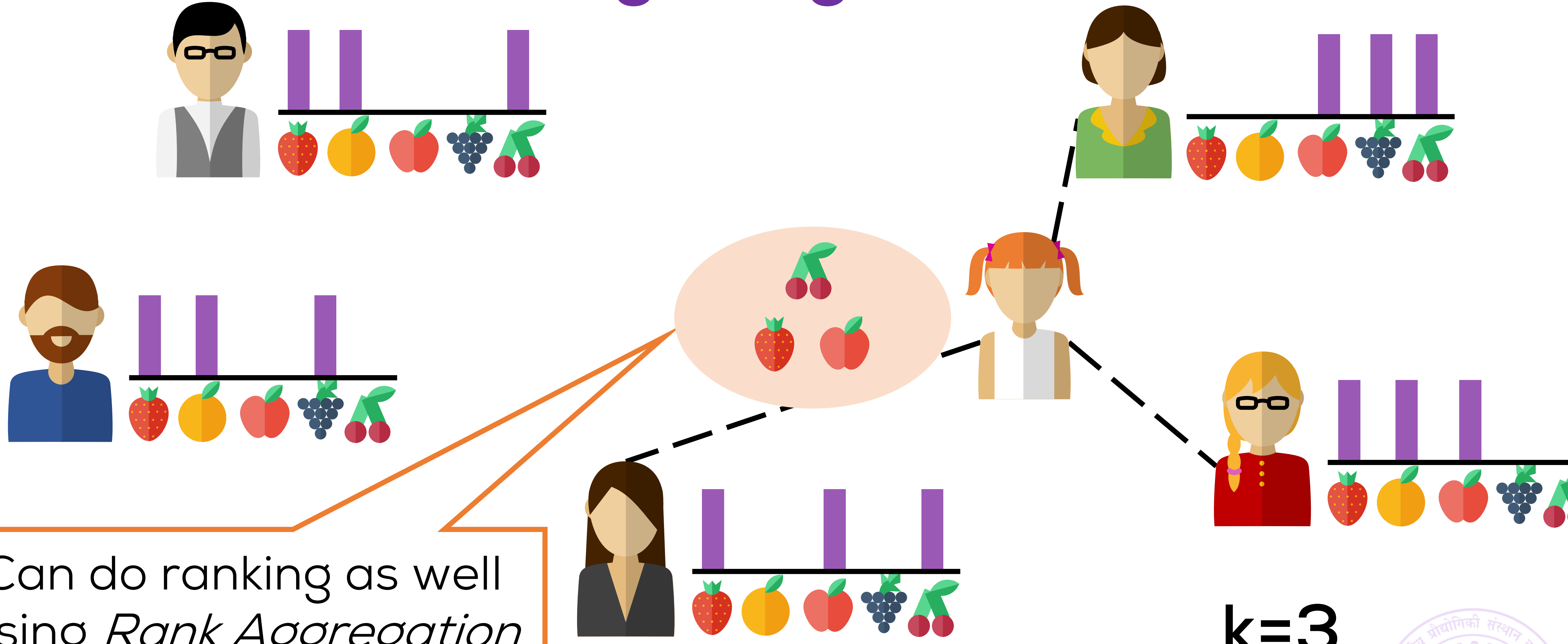
k=3

Multi-label Learning using k-NN

Multi-label Learning using k-NN



Multi-label Learning using k-NN



Can do ranking as well using *Rank Aggregation*

k=3

Please give your Feedback

http://tinyurl.com/ml17-18afb