

95-865 Unstructured Data Analytics

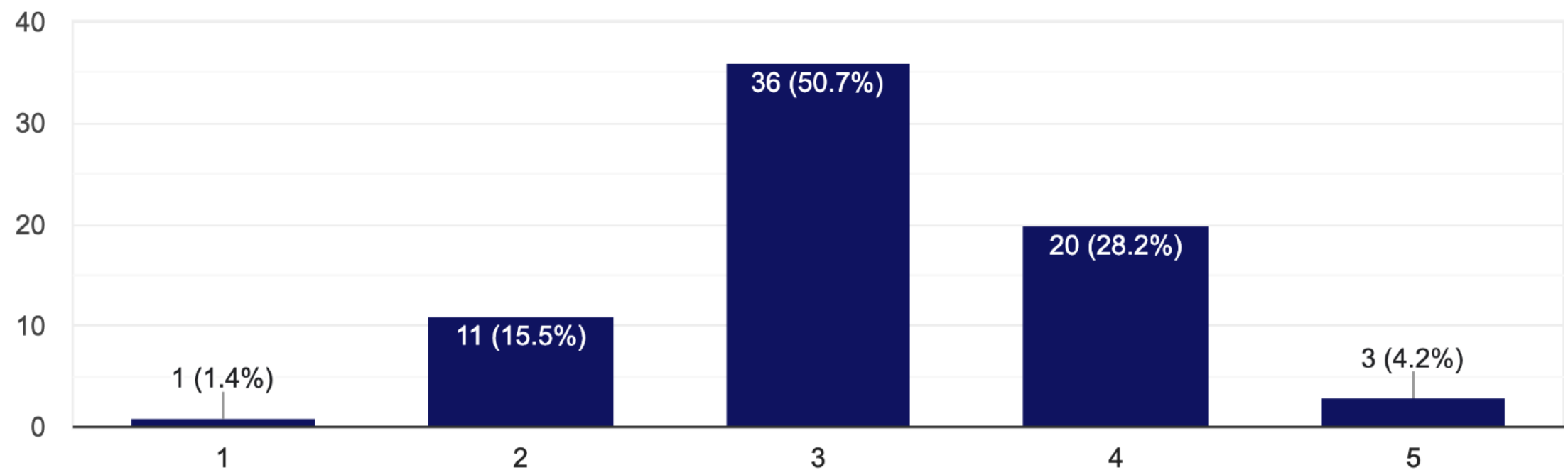
Lecture 10: Wrap up topic modeling,
intro to predictive data analytics

Slides by George H. Chen

HW1 Questionnaire Results (1/n)

How difficult did you find homework 1?

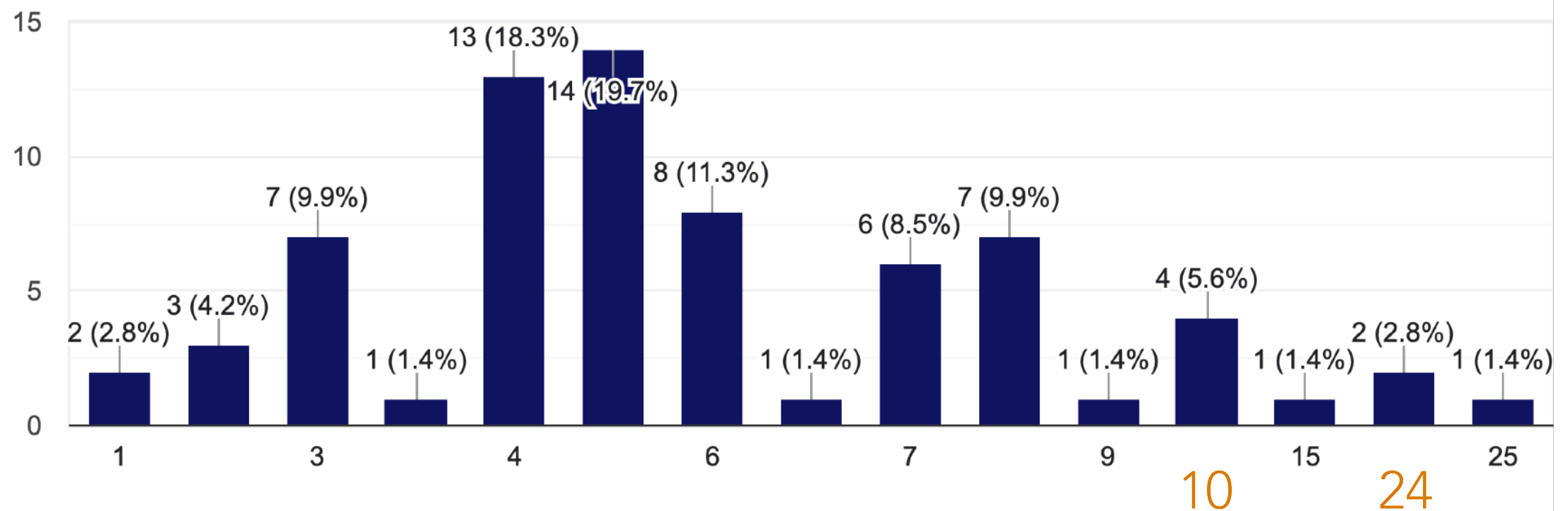
71 responses



HW1 Questionnaire Results (2/n)

How many hours did you take (roughly) to complete homework 1?

71 responses



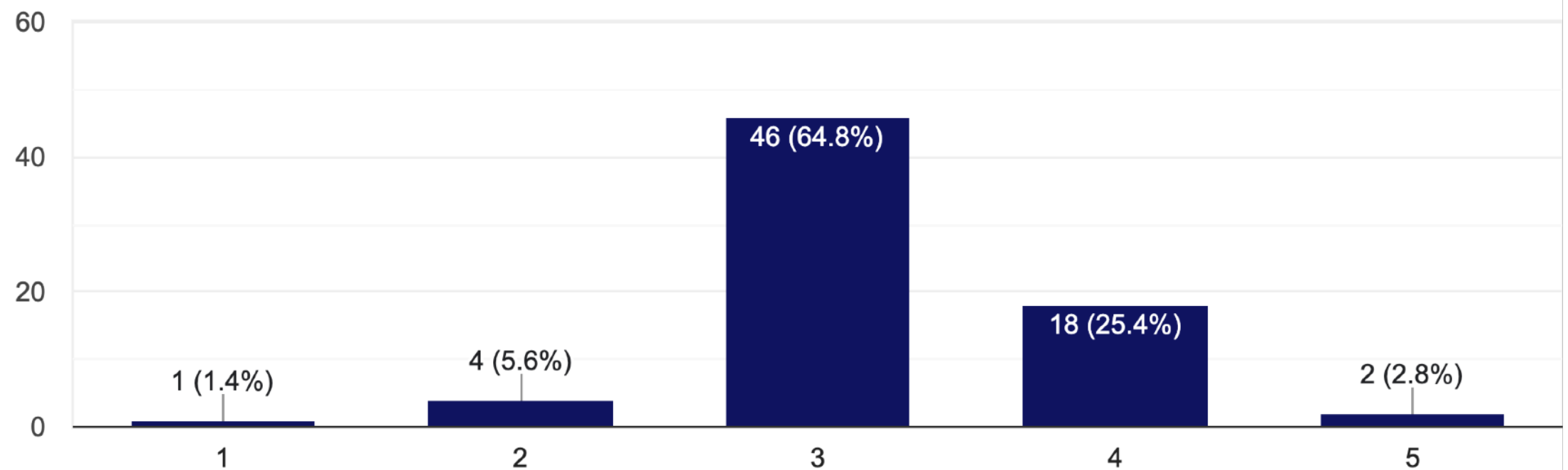
Overall: the vast majority of students are completing homework in a reasonable amount of time!

(We target each HW to take no more than 15 hours; most students should be able to finish HW within 10 hours)

HW1 Questionnaire Results (3/n)

How do you find the lecture pace?

71 responses

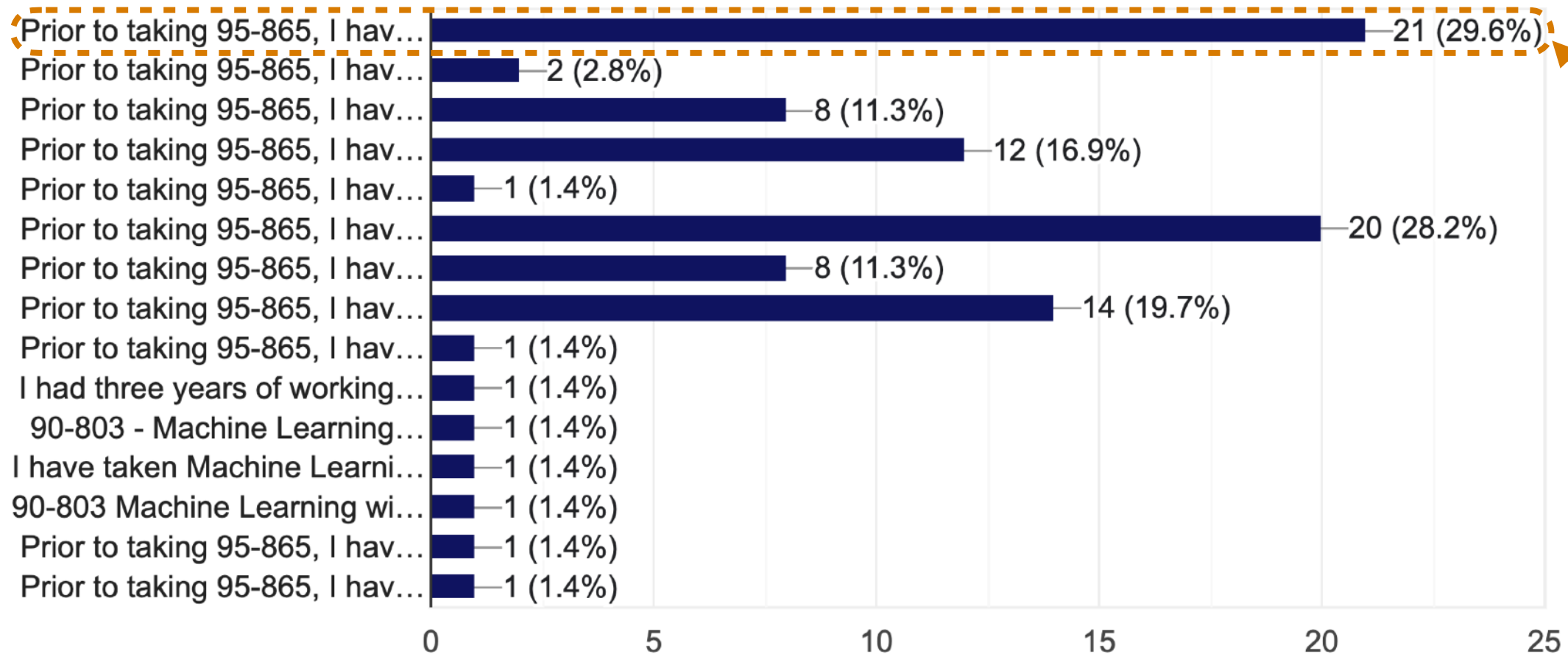


Mostly okay

HW1 Questionnaire Results (4/n)

We have been trying to figure out to what extent students come into the class already having data analysis/machine learning knowledge. Please select... classes you are taking concurrently with 95-865.

71 responses



29.6% of students said they have no prior ML/deep learning experience

HW1 Questionnaire Results (5/n)

- A lot of students want to see more code examples
 - Already we're averaging roughly 1 demo a lecture and I cannot easily add more demos at this point
 - I think it's valuable for you to also find demos by other people! There are now *tons* of code demos by lots of people!
- Some students discuss having trouble following along with code demos live in middle of lecture
 - What would work best for your learning depends on the student
 - One strategy: during lecture, don't actually try to do the code demo the same time I do it; instead try the demo yourself after class
 - Try to take notes of things you don't understand during live code demos?
 - Please do interrupt me during code demos if you have a question!

HW1 Questionnaire Results (6/n)

- Some students would like some way to check whether their solutions are correct
 - I think it's actually healthy for you to understand that this is in general difficult
 - When analyzing real data, we often don't know what the "correct" answer is, and sometimes it's really not obvious if there's a code bug
 - Extremely important skill to develop:
 - Come up with toy (really small) test cases for which you can easily derive (manually by hand) what the correct answer is, and check that the code outputs the correct answers for these test cases
- Debugging machine learning code is often difficult because of randomness!

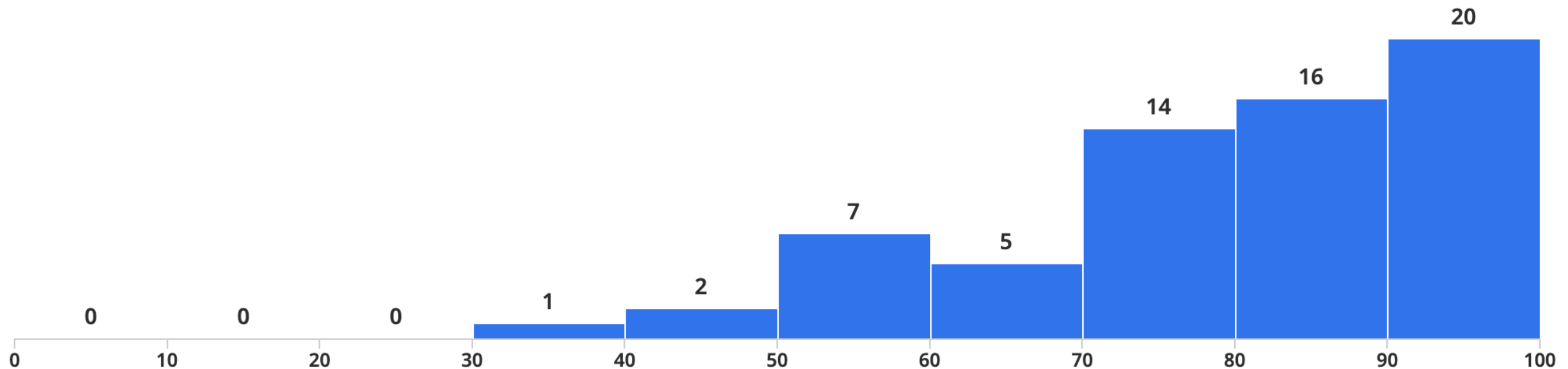
HW1 Questionnaire Results (7/n)

- Some students would like me to spend more time during lecture talking about homework
- I actually think it's more valuable for me to spend time in lecture teaching concepts rather than going over homework details
- Homework intentionally is designed to make you have to think a bit about how to apply what you learned in lecture to real-ish problems
- Please make use of Piazza & our office hours for your homework questions!

HW1 Questionnaire Results (8/n)

- Some students complained that HW datasets are too large
 - The HW datasets are actually *not* too large by unstructured data analysis standards (e.g., LLMs are trained on much larger datasets)
 - We want you to get a sense of computational difficulties that arise in working with real data, and to learn to debug carefully
 - If the code takes a few hours to run, that's actually not that bad!
 - Make sure your code is fully working on a smaller dataset before running your code on the full dataset
 - Many state-of-the-art deep learning systems require enormous compute power and can take weeks to train

Quiz 1



Mean: 78.9

Std dev: 16.25

Maximum achieved: 100 (3 students achieved this)

I'll post regrade instructions later today

Topic Modeling: Last Remarks

- There are other topic model score functions aside from coherence & # unique top words
(e.g., normalized-mutual-information coherence, (log) lift score, ...)
- There are *many* topic models, not just LDA
(e.g., Hierarchical Dirichlet Process, correlated topic models, SAGE, anchor word topic models, Scholar, embedded topic model, ...)
- *Dynamic* topic models can track how topics change *over time*
 - Requires time stamp for every text document we fit the model to
- **Warning:** learning topic models is very sensitive to random initialization
 - Can try fitting data multiple times using different random seeds & seeing which topics consistently show up across random seeds
- There are variants of topic models where users provide supervision
(e.g., user specifies what some topics should be about in terms of top words, or where a topic should predict some outcome)

95-865

Part I: Exploratory data analysis

Identify structure present in “unstructured” data

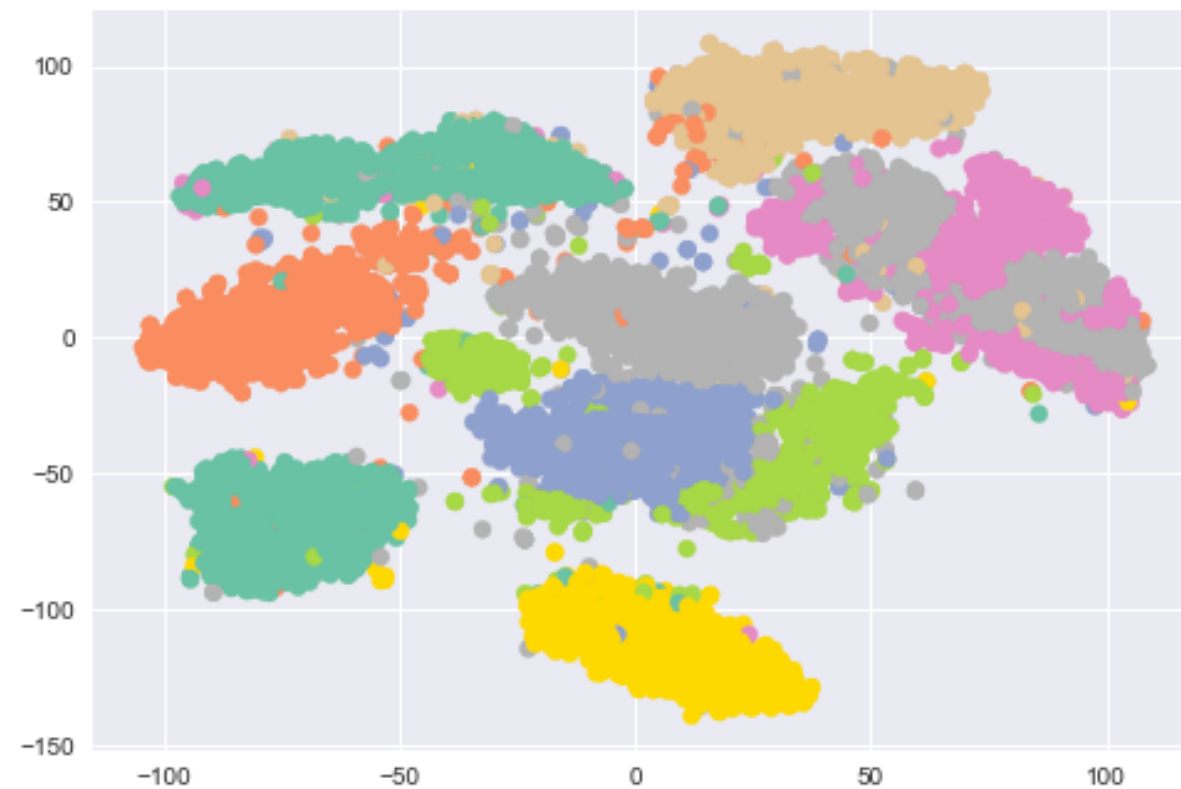
- Frequency and co-occurrence analysis
- Visualizing high-dimensional data/dimensionality reduction
- Clustering
- Topic modeling

Part II: Predictive data analysis

Make predictions using known structure in data

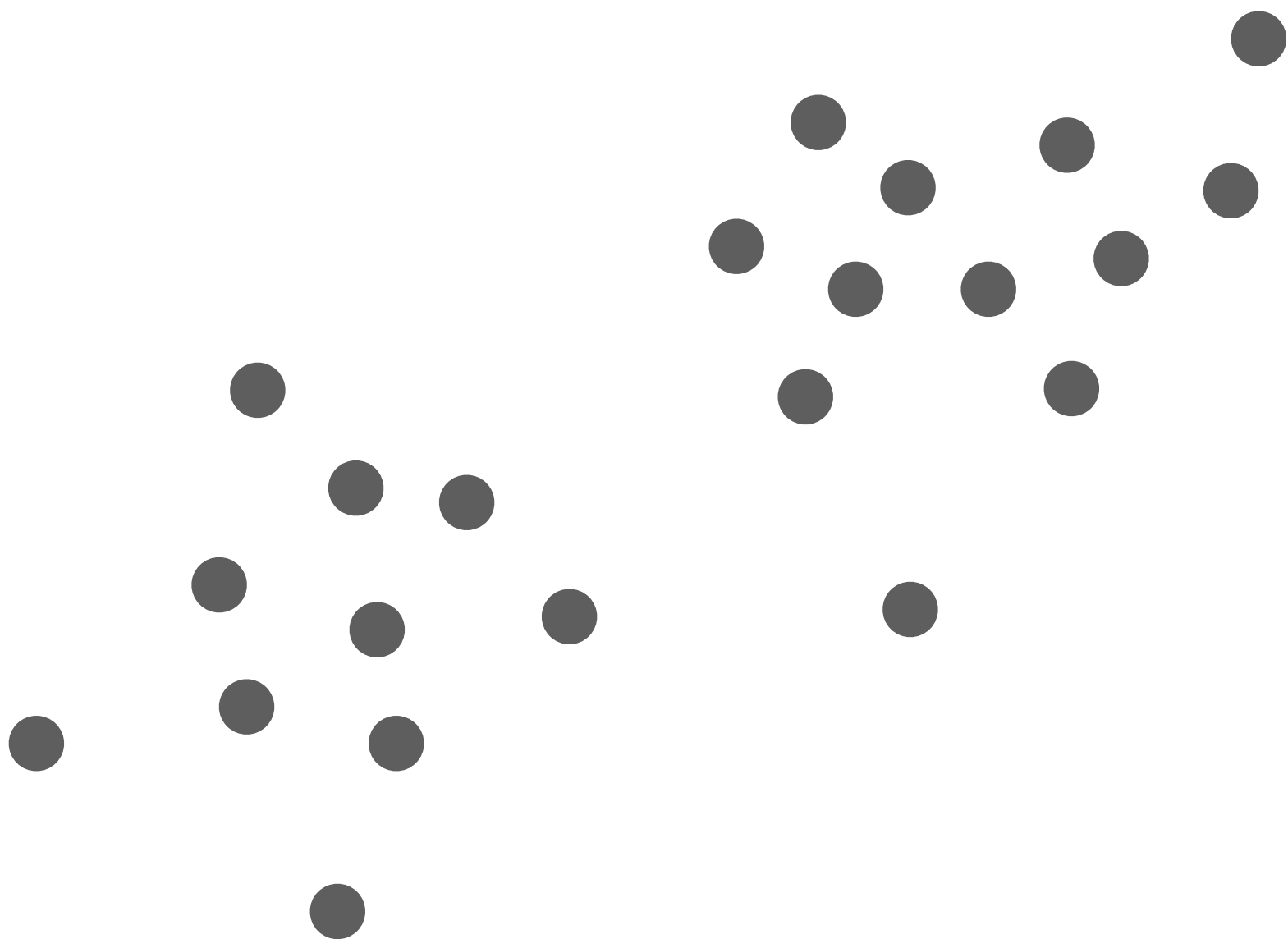
- Basic concepts and how to assess quality of prediction models
- Neural nets and deep learning for analyzing images and text

What if we have labels?



Example: MNIST handwritten digits have known labels

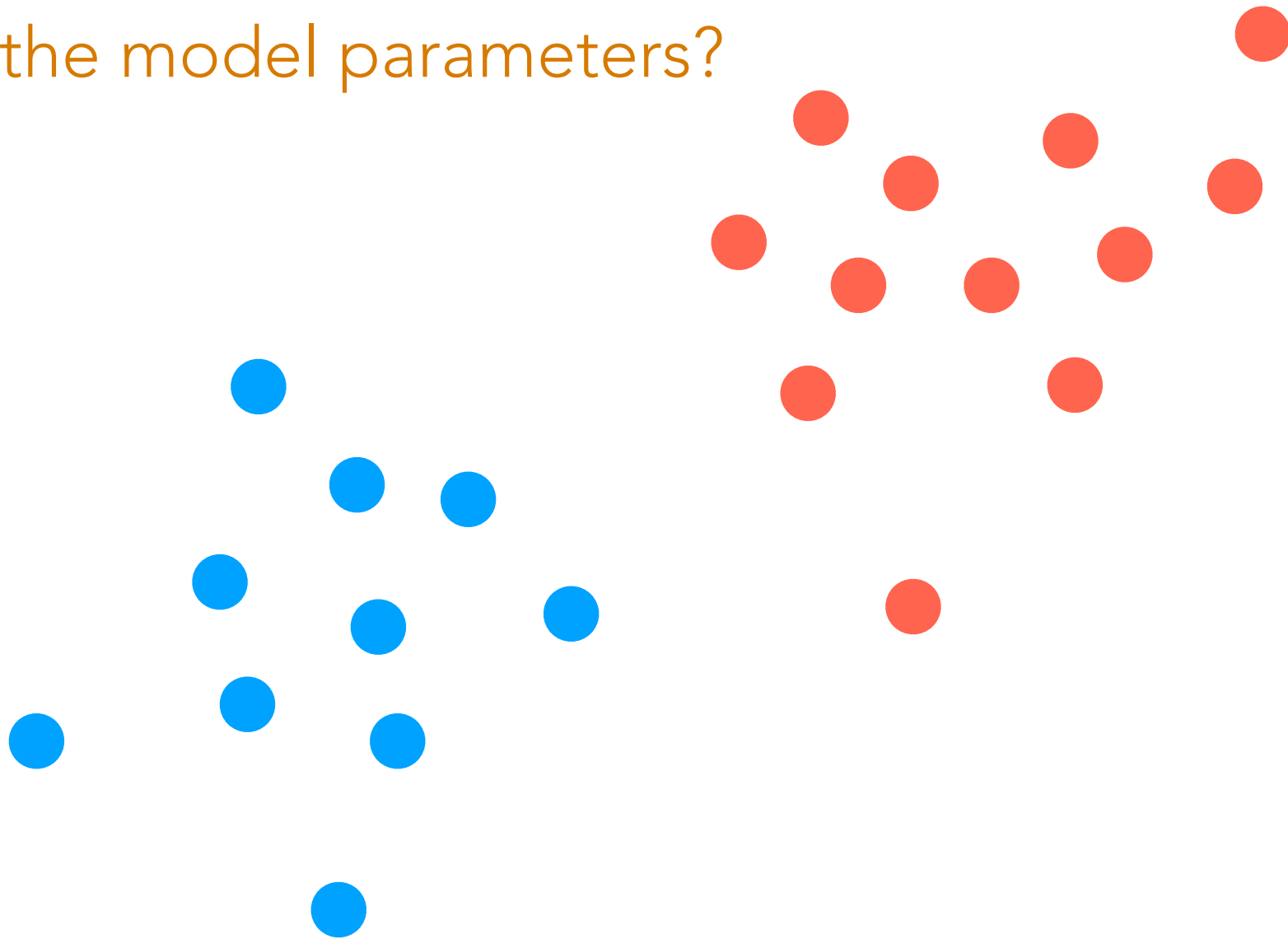
If the labels are known...



If the labels are known...

And we assume data generated by GMM...

What are the model parameters?



(Flashback) Learning a GMM

Don't need this top part if we know the labels!

Step 0: Guess k

Step 1: Guess cluster probabilities, means, and covariances
(often done using k -means)

Repeat until convergence:

Step 2: Compute probability of each point being in each of the k clusters

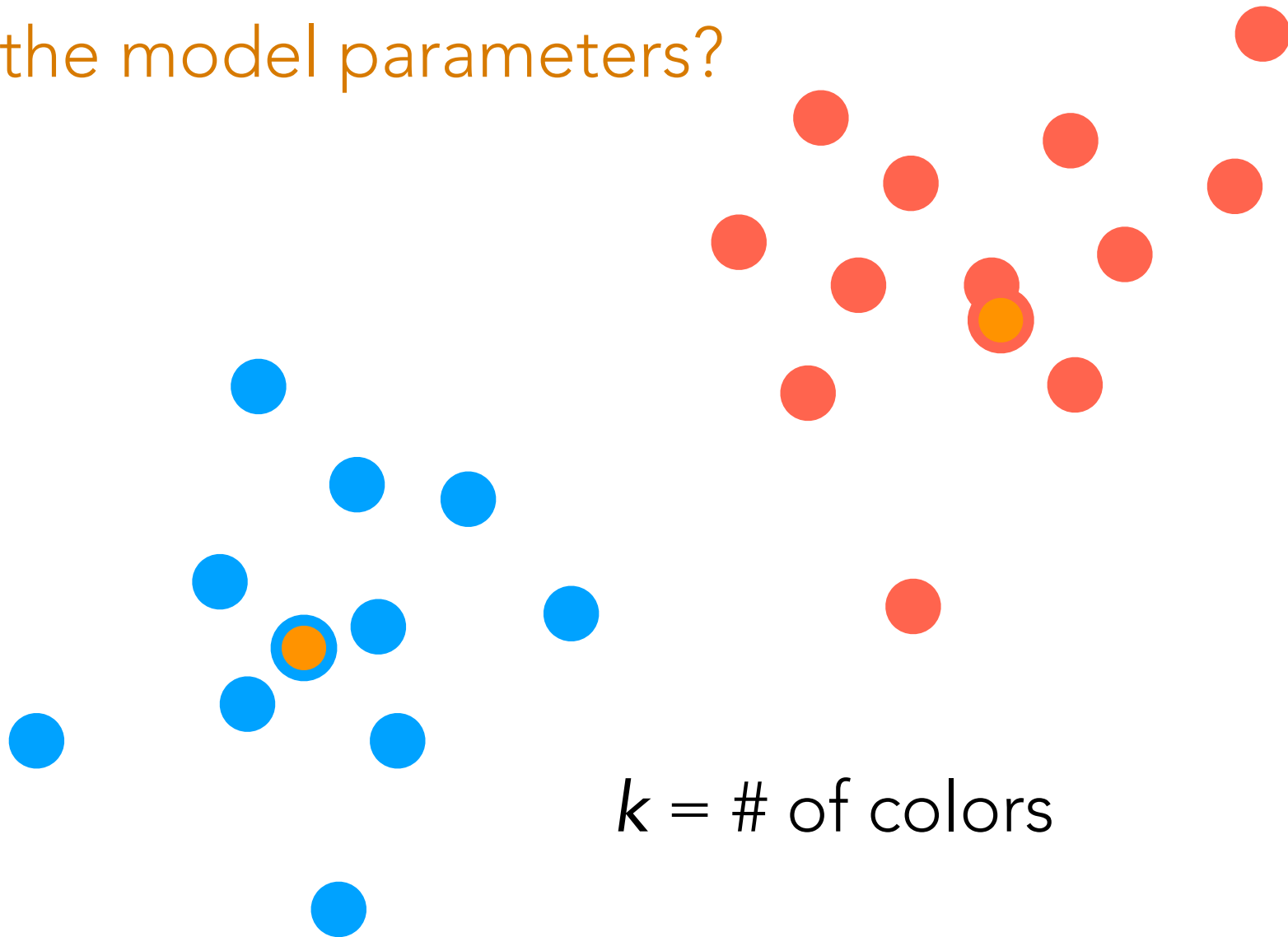
Step 3: Update cluster probabilities, means, and covariances accounting for probabilities of each point belonging to each of the clusters

We don't even need to repeat until convergence

If the labels are known...

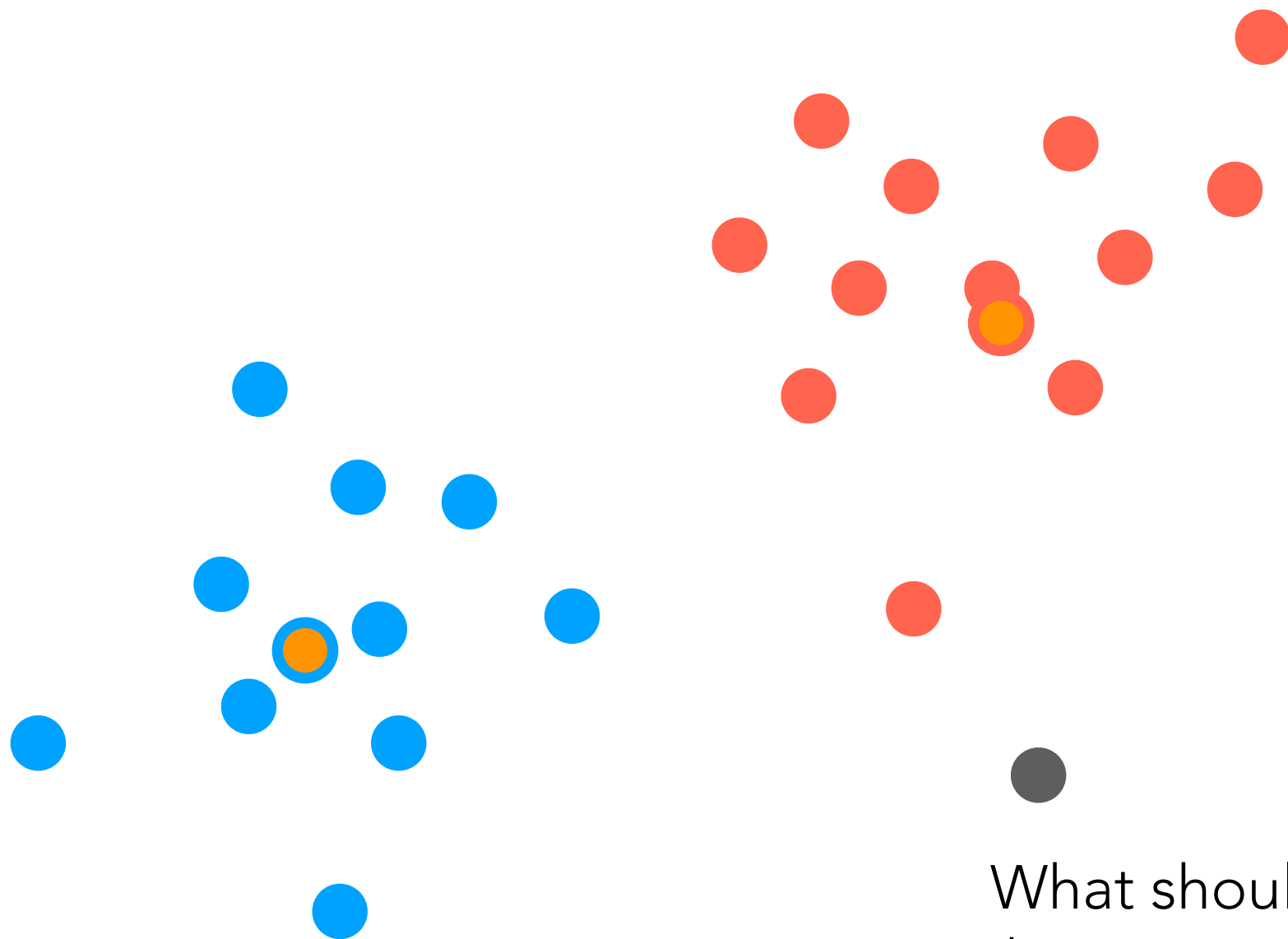
And we assume data generated by GMM...

What are the model parameters?



$k = \#$ of colors

We can directly estimate cluster means, covariance matrices

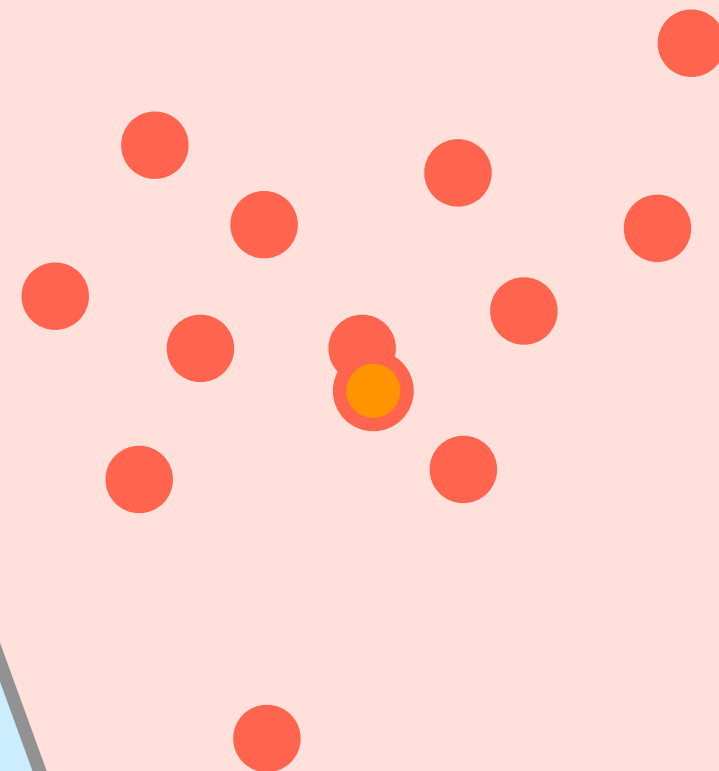
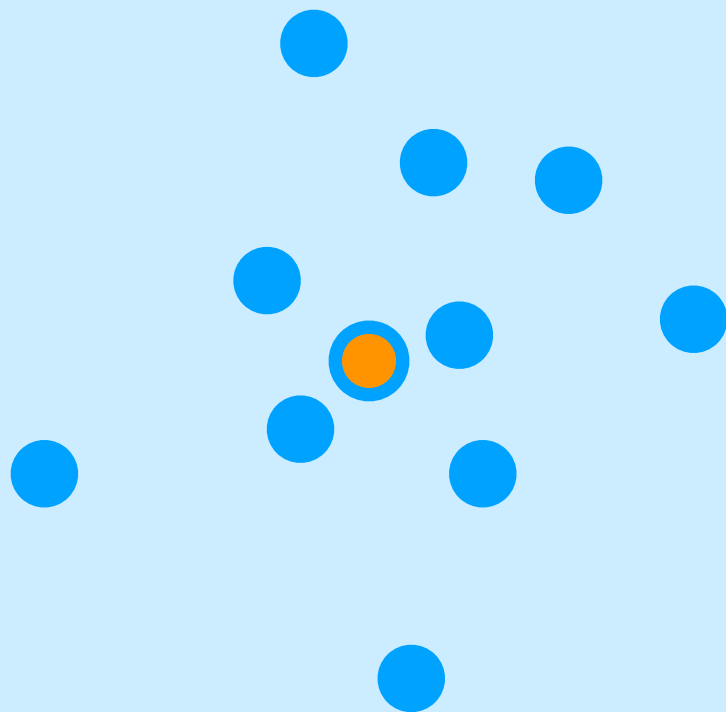


What should the label of
this new "test" point be?

Whichever cluster has
higher probability!

Decision boundary

We just created a **classifier**
(a procedure that given a test data point
tells us what "class" it belongs to)



This classifier we've created assumes a
generative model

What should the label of
this new "test" point be?

Whichever cluster has
higher probability!

Predictive Data Analysis

Training data

$$(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$$

Goal: Given new test feature vector x , predict label y

- y is discrete (such as colors **red** and **blue**)
→ prediction is referred to as **classification**
- y is continuous (such as a real number)
→ prediction is referred to as **regression**

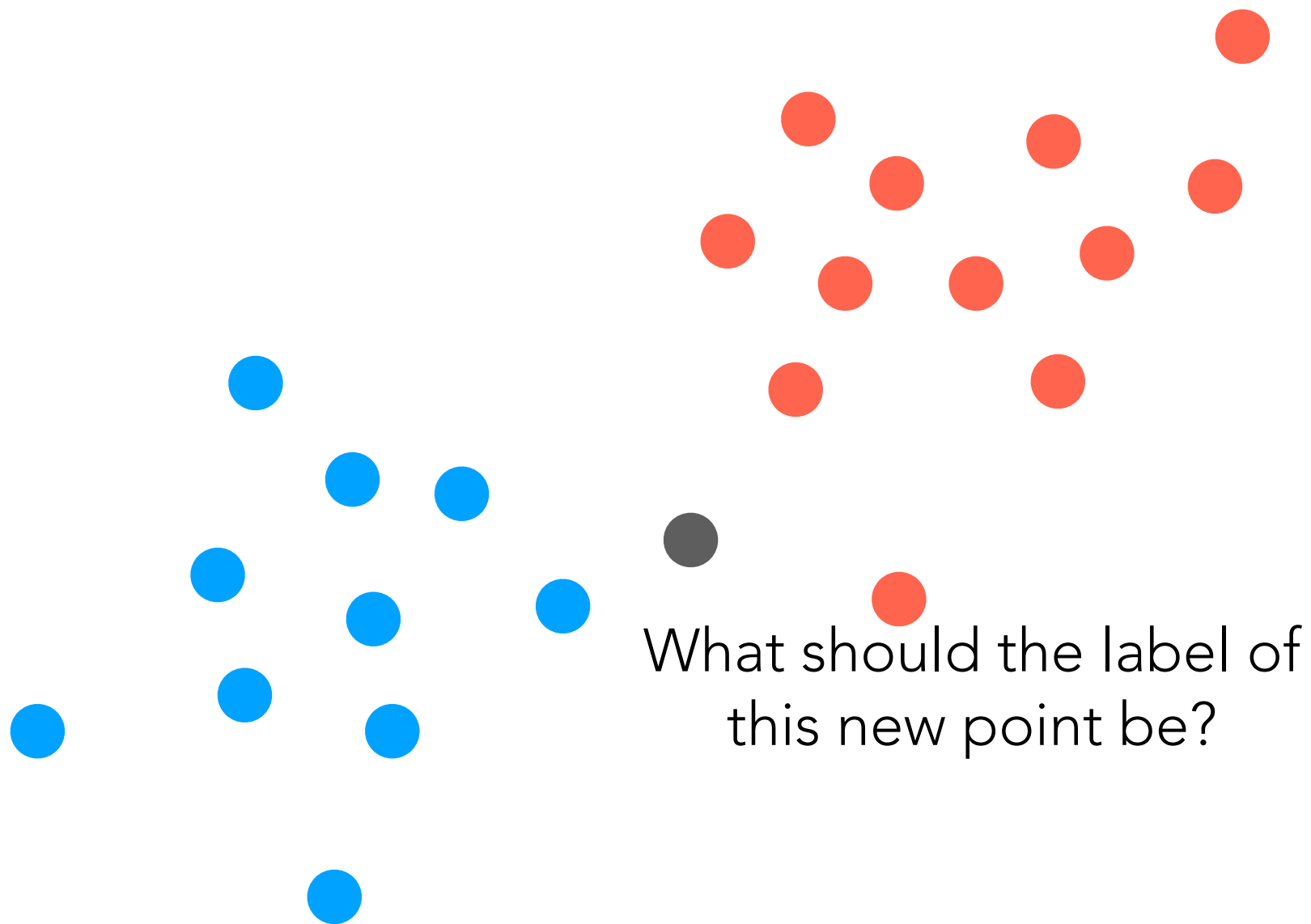
We could have many such test feature vectors, which we collectively refer to as *test data*

A giant zoo of methods

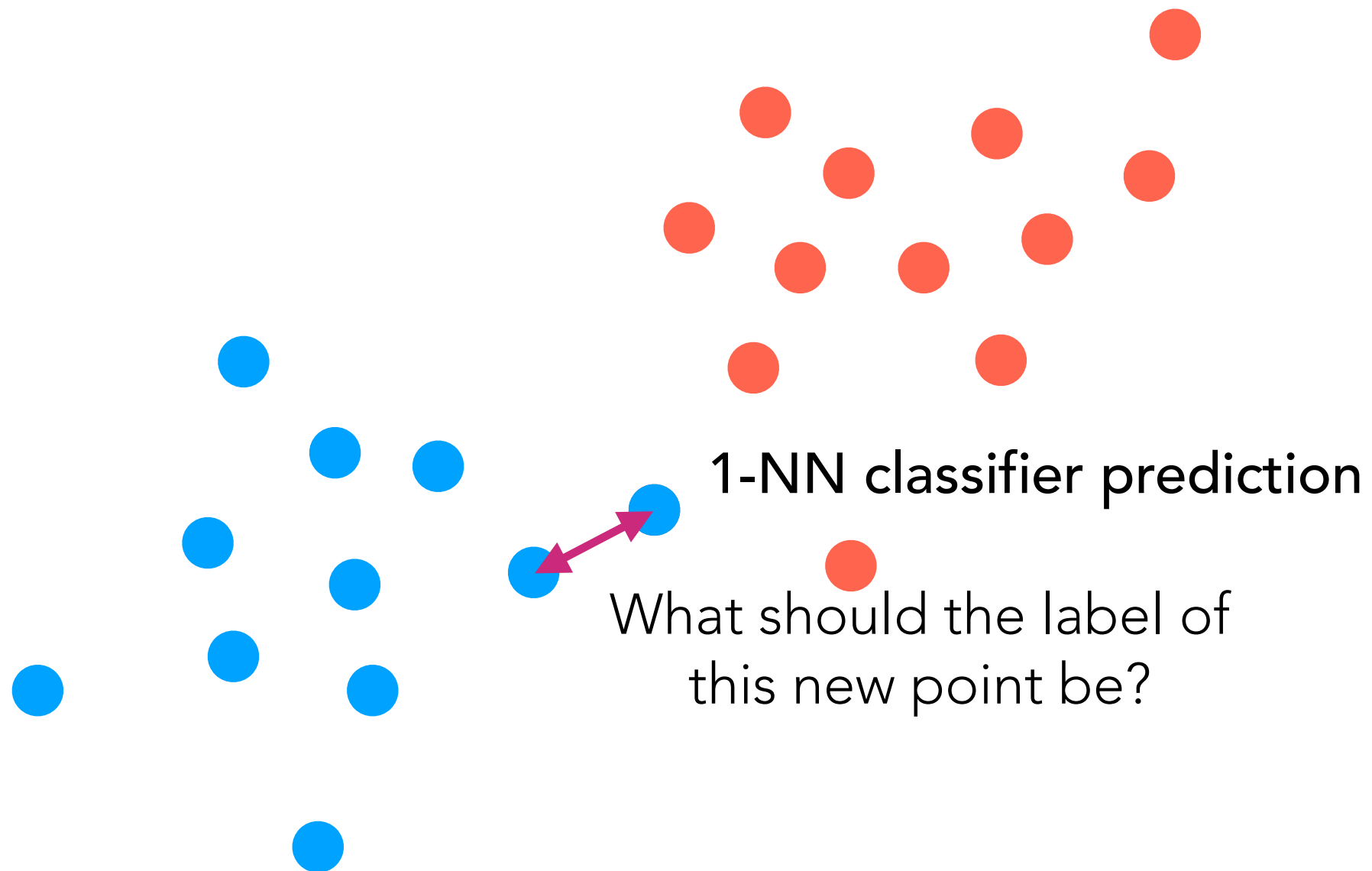
- Generative models (like what we just described)
- Discriminative models (just care about learning prediction rule; after training model, we don't have a way to generate data)

Example of a Discriminative
Method: k -NN Classification

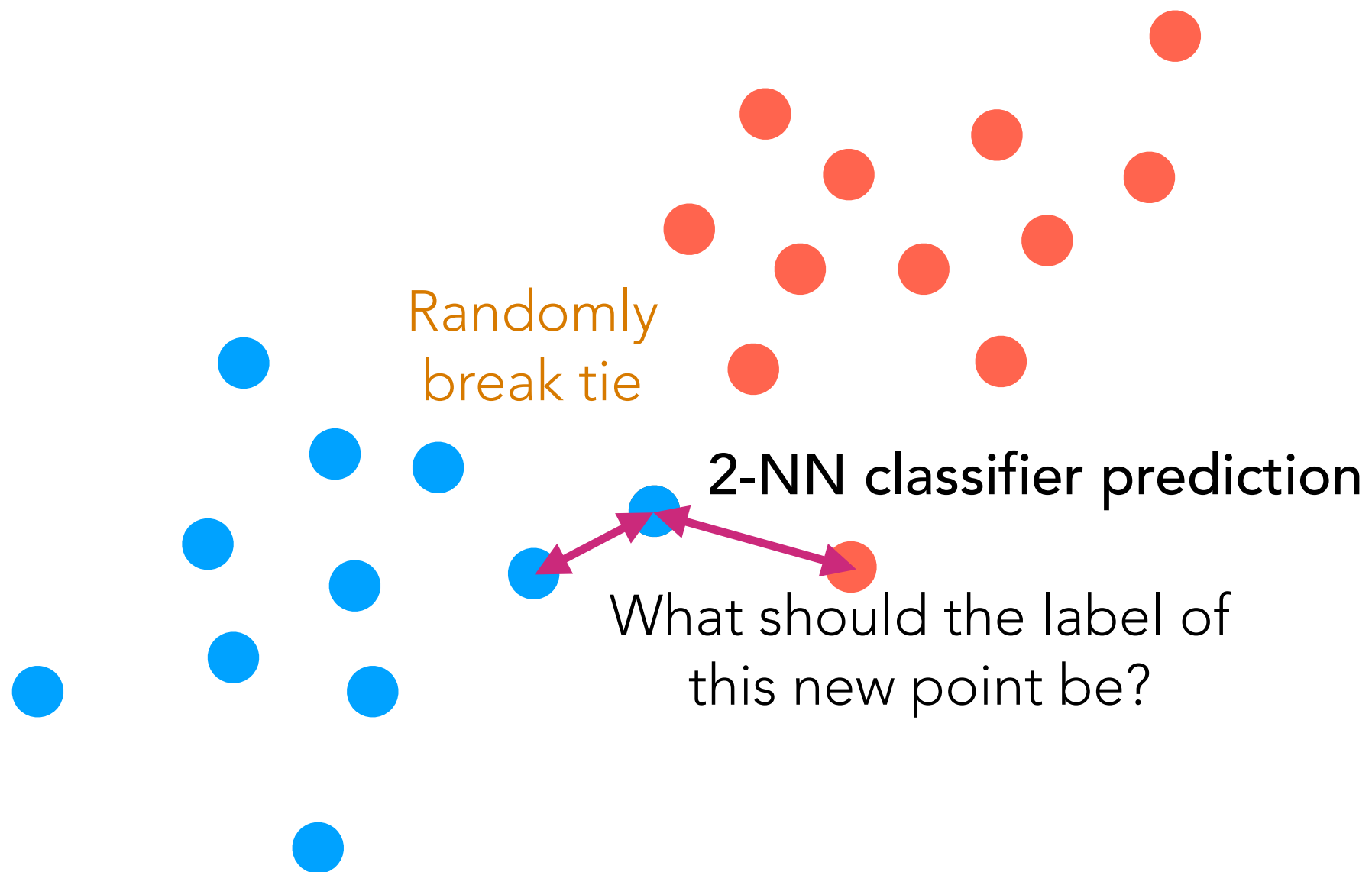
Example: k -NN Classification



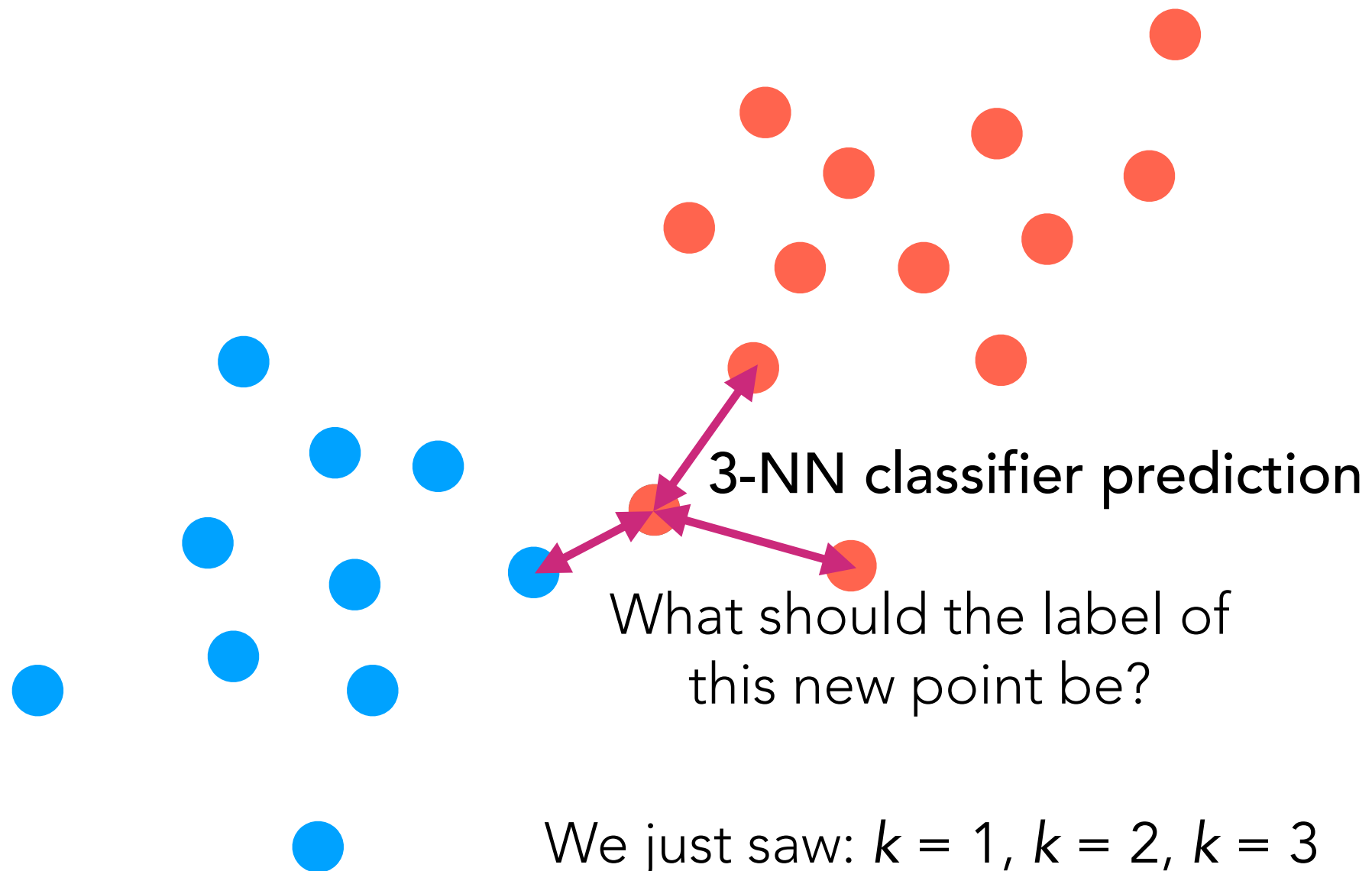
Example: k -NN Classification



Example: k -NN Classification



Example: k -NN Classification



What happens if $k = n$?

How do we choose k ?

What I'll describe next can be used to select hyperparameter(s) for any prediction method


Fundamental question:
How do we assess how good a prediction method is?

Hyperparameters vs. Parameters

- We fit a model's parameters to training data (terminology: we "learn" the parameters)
- We pick values of hyperparameters and they do *not* get fit to training data
- Example: Gaussian mixture model
 - Hyperparameter: number of clusters k
 - Parameters: cluster probabilities, means, covariance matrices
- Example: k -NN classification
 - Hyperparameter: number of nearest neighbors k
 - Parameters: N/A

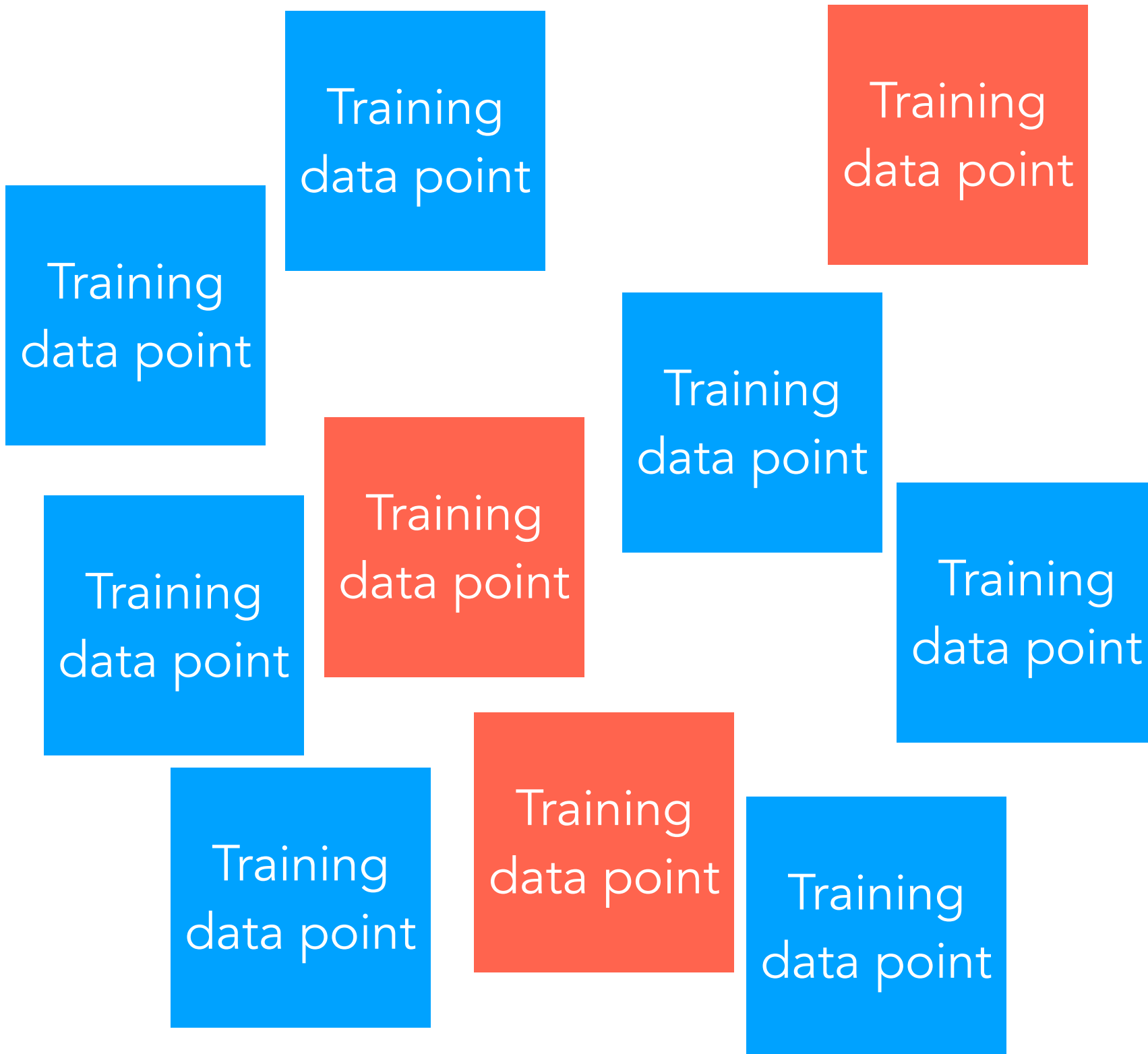
Actually, there's another hyperparameter: distance function to use
(for simplicity, we assume Euclidean distance for now)



 Major assumption:
training and test data “look alike”
(technically: training and test data are i.i.d.
sampled from the same underlying distribution)

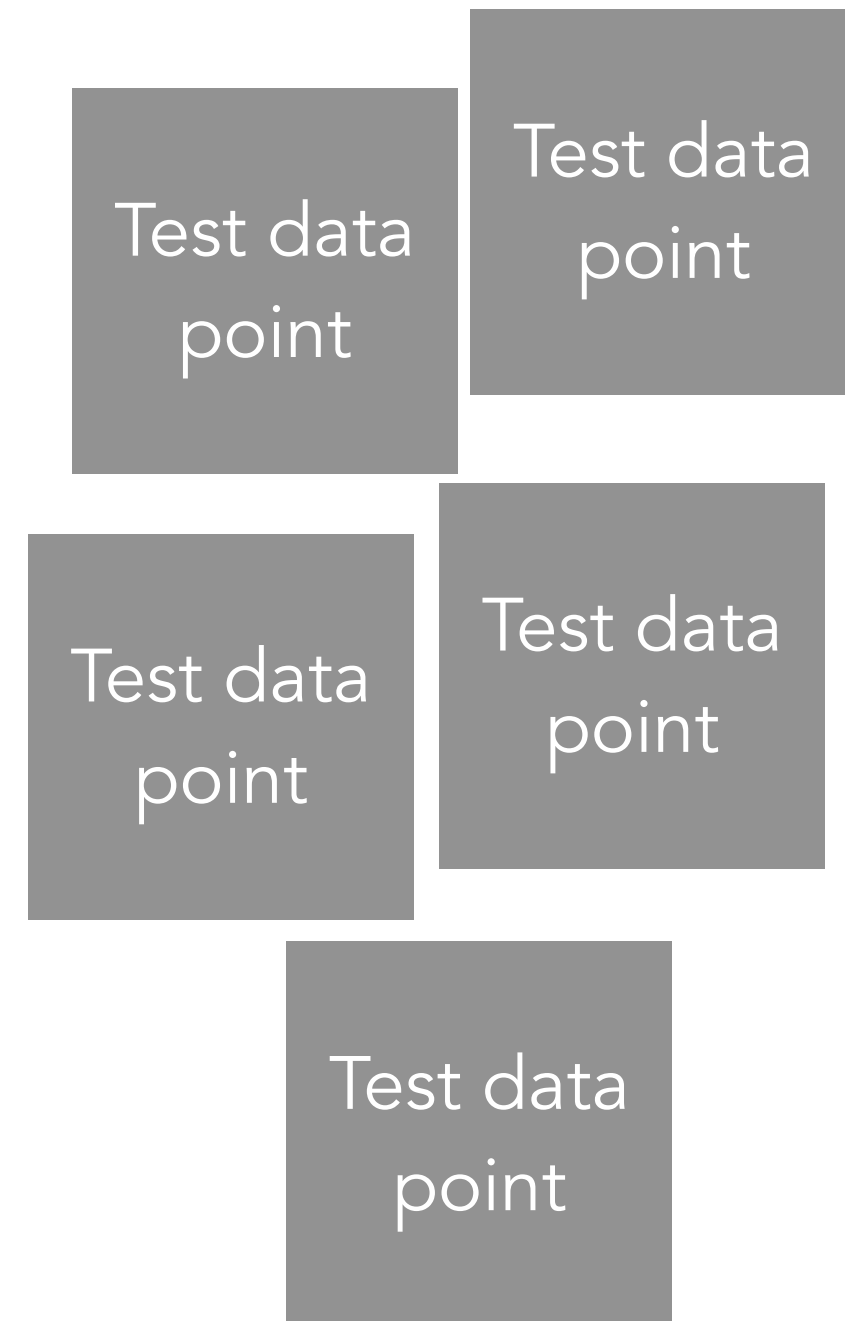
Prediction is harder when training and test data appear quite different!

Training data



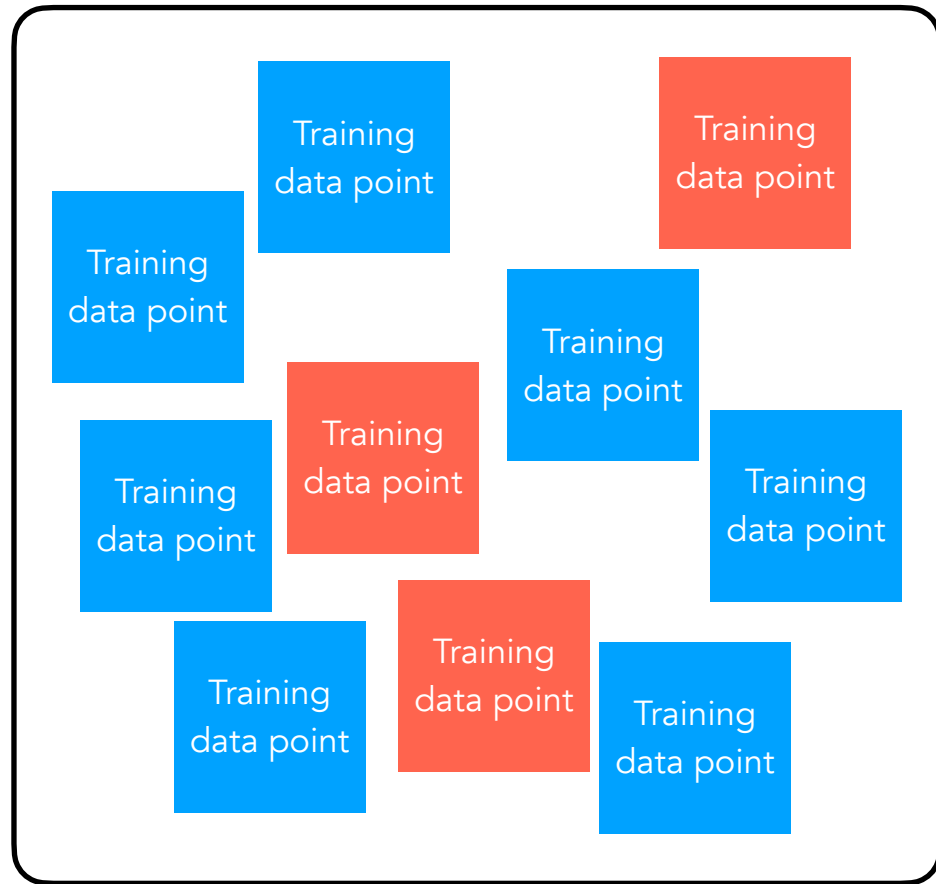
Example: Each data point is an email and we know whether it is spam/ham

Want to classify these points correctly



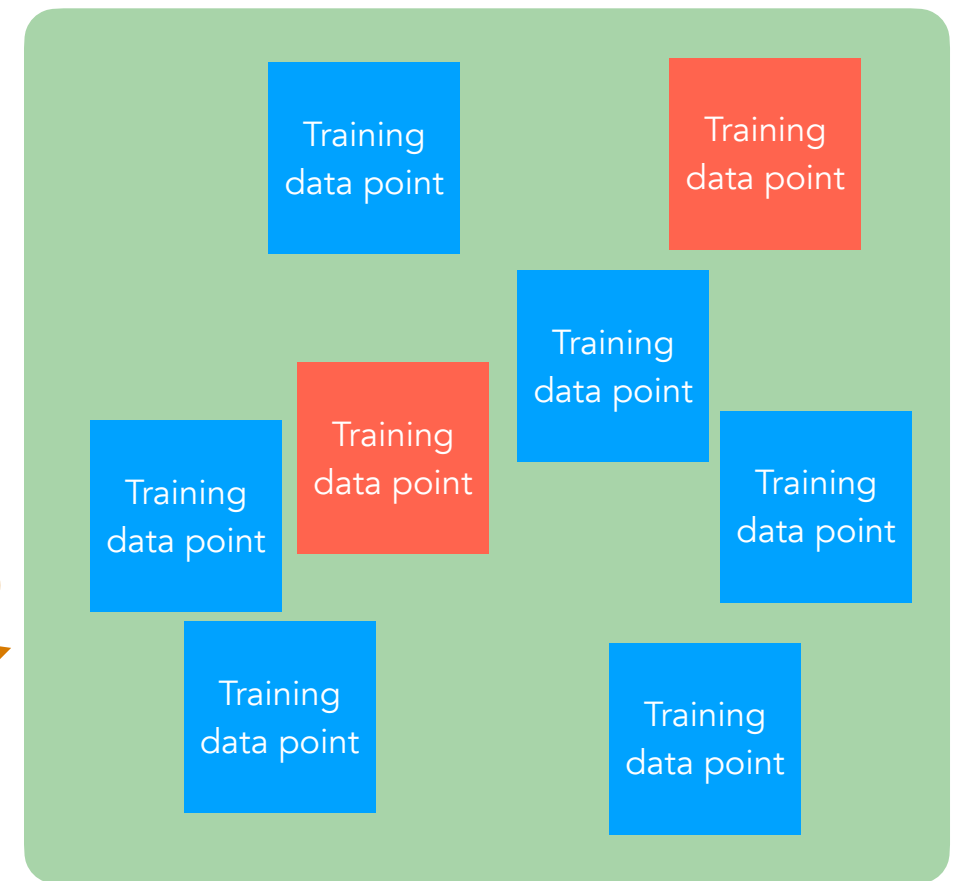
Example: future emails to classify as spam/ham

Training data

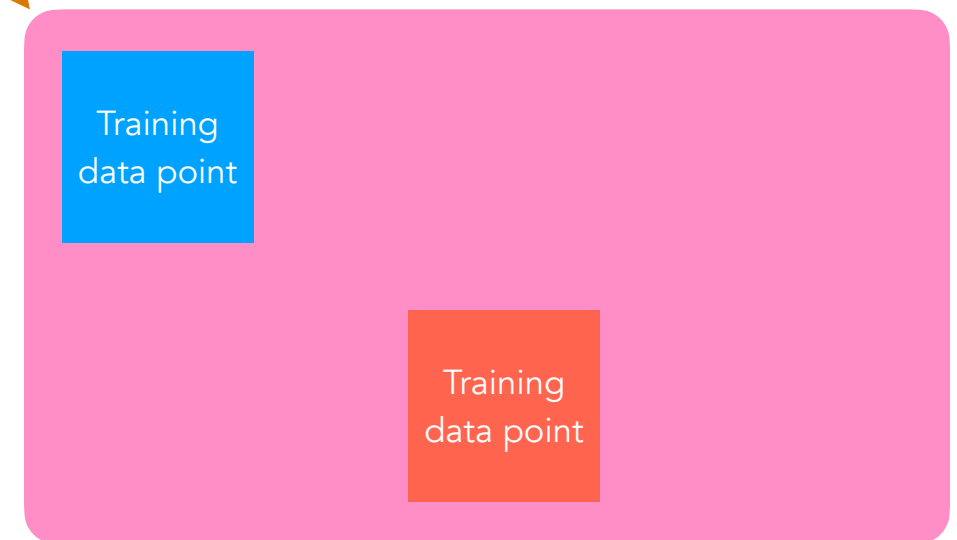


Randomly split into
two portions
(example: 80% / 20%)

"Proper training data"



"Validation data"



For $k = 1, 2, \dots$, some user-specified max:

1. Train k -NN classifier on proper training data
2. Use a score function to evaluate how well the trained model predicts on validation data

Use whichever value of k achieves the best score

There are many score functions possible

Example: fraction of validation points correctly predicted (raw accuracy)


Terminology Remarks

- What we're using is commonly called a **train/validation split**
 - If you also consider that there's a test set that's not part of the train/validation data, then the overall three-piece division is called the **train/validation/test split**
- Confusingly, in the machine learning community, what I'm calling the "proper training data"/"proper training set" is commonly also called the "training data"/"training set" even though it is typically a *subset* of the full training data (that we split into proper training/validation sets)
 - Put another way, what precisely the "training data" refers to can be ambiguous as it could mean the full training data or the full training data minus the validation data
 - In 95-865, to avoid confusion, we use the somewhat non-standard terminology "proper training set" or "proper training data" to refer to the part of the full training data that we actually train a model on (i.e., the full training data minus the validation data)

Hyperparameter Tuning for k -NN Classifier

Demo

Hyperparameter Tuning in General (Not Just for k -NN Classifier)

Suppose that we have a classifier with hyperparameter setting θ 
could consist of multiple hyperparameters
(think of θ as a tuple)

For each hyperparameter setting θ (in a list of hyperparameter settings we are willing to try):

1. Train classifier on proper training data using hyperparameter setting θ
2. Use a score function to evaluate how well the trained model predicts on validation data

Use classifier corresponding to whichever value of θ achieves the best score

- ⚠ How we randomly split the training data into proper training/validation sets affects the scores we get
- ⚠ If the classifier's training procedure is random, then using different random seeds could also change the scores we get