

DATA SCIENCE

LECTURE 4: K-NEAREST NEIGHBORS CLASSIFICATION

YUCHEN ZHAO / DAT-14

RECAP

LAST TIME:

I. DATA RETRIEVAL

II. ETL INTRO

III. VISUALIZATION

RECAP

LAST TIME:

I. DATA RETRIEVAL (API, JSON)

II. ETL INTRO (DATABASE, SQL)

III. VISUALIZATION (D3.JS)

EXERCISES:

IV. PANDAS

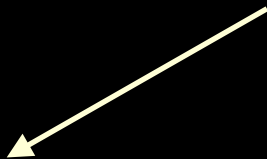
V. MINING TWITTER VIA API

QUESTIONS?

INTRO TO DATA SCIENCE

FINAL PROJECT KICKOFF

why start so early?



FINAL PROJECT KICKOFF

Final Projects in Brief:

- *Select a dataset*
- *Formulate a hypothesis / problem statement*
- *What question do you want to answer? / What do you want to predict?*
- *Why is it useful to do so?*
- *Wrangle the data*
- *Build at least one model*
- *Test that model*
- *Write up your project and what you learned*
- *Present your project to the class (last week of class)*

PROJECT REQUIREMENT & SCHEDULE:
**[HTTPS://GITHUB.COM/GA-STUDENTS/
DAT_SF_14/TREE/MASTER/PROJECT](https://github.com/GA-STUDENTS/DAT_SF_14/tree/master/project)**

I. CLASSIFICATION PROBLEMS

II. BUILDING EFFECTIVE CLASSIFIERS

III. KNN CLASSIFICATION

EXERCISES:

IV. EXPLORING & IMPLEMENTING K-NN CLASSIFICATION

I. CLASSIFICATION PROBLEMS

	<i>continuous</i>	<i>categorical</i>
<i>supervised</i>	???	???
<i>unsupervised</i>	???	???

	<i>continuous</i>	<i>categorical</i>
<i>supervised</i>	<i>regression</i>	<i>classification</i>
<i>unsupervised</i>	<i>dimension reduction</i>	<i>clustering</i>

CLASSIFICATION PROBLEMS

Here's (part of) an example dataset:

Fisher's *Iris* Data

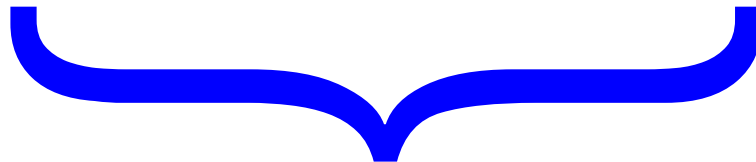
Sepal length ⇅	Sepal width ⇅	Petal length ⇅	Petal width ⇅	Species ⇅
5.1	3.5	1.4	0.2	<i>I. setosa</i>
4.9	3.0	1.4	0.2	<i>I. setosa</i>
4.7	3.2	1.3	0.2	<i>I. setosa</i>
4.6	3.1	1.5	0.2	<i>I. setosa</i>
5.0	3.6	1.4	0.2	<i>I. setosa</i>
5.4	3.9	1.7	0.4	<i>I. setosa</i>
4.6	3.4	1.4	0.3	<i>I. setosa</i>
5.0	3.4	1.5	0.2	<i>I. setosa</i>

Here's (part of) an example dataset:

*independent
variables*

Fisher's Iris Data

Sepal length ⇅	Sepal width ⇅	Petal length ⇅	Petal width ⇅	Species ⇅
5.1	3.5	1.4	0.2	<i>I. setosa</i>
4.9	3.0	1.4	0.2	<i>I. setosa</i>
4.7	3.2	1.3	0.2	<i>I. setosa</i>
4.6	3.1	1.5	0.2	<i>I. setosa</i>
5.0	3.6	1.4	0.2	<i>I. setosa</i>
5.4	3.9	1.7	0.4	<i>I. setosa</i>
4.6	3.4	1.4	0.3	<i>I. setosa</i>
5.0	3.4	1.5	0.2	<i>I. setosa</i>



Here's (part of) an example dataset:

Fisher's Iris Data

Sepal length ⇅	Sepal width ⇅	Petal length ⇅	Petal width ⇅	Species ⇅
5.1	3.5	1.4	0.2	<i>I. setosa</i>
4.9	3.0	1.4	0.2	<i>I. setosa</i>
4.7	3.2	1.3	0.2	<i>I. setosa</i>
4.6	3.1	1.5	0.2	<i>I. setosa</i>
5.0	3.6	1.4	0.2	<i>I. setosa</i>
5.4	3.9	1.7	0.4	<i>I. setosa</i>
4.6	3.4	1.4	0.3	<i>I. setosa</i>
5.0	3.4	1.5	0.2	<i>I. setosa</i>

*independent
variables*

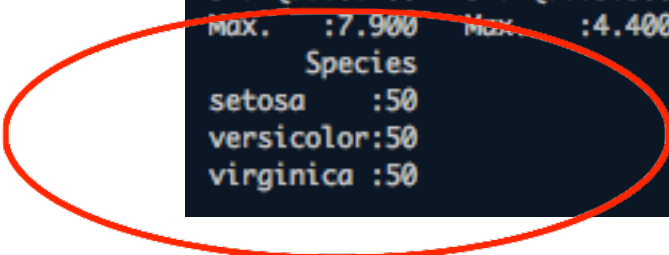
*class
labels
(qualitative)*

Q: What does “supervised” mean?

Q: What does “supervised” mean?

A: We know the labels.

```
Welcome to R! Thu Feb 28 13:07:25 2013
> summary(iris)
  Sepal.Length  Sepal.Width  Petal.Length  Petal.Width
Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100
1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300
Median :5.800   Median :3.000   Median :4.350   Median :1.300
Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
   Species
setosa   :50
versicolor:50
virginica :50
```



Q: How does a classification problem work?

Q: How does a classification problem work?

A: Data in, predicted labels out.

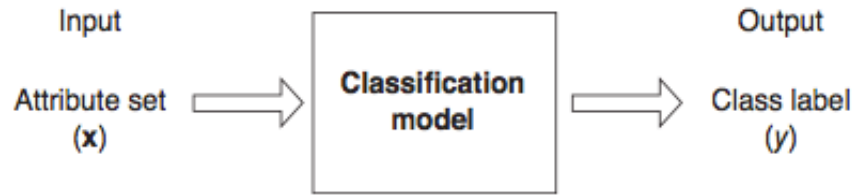
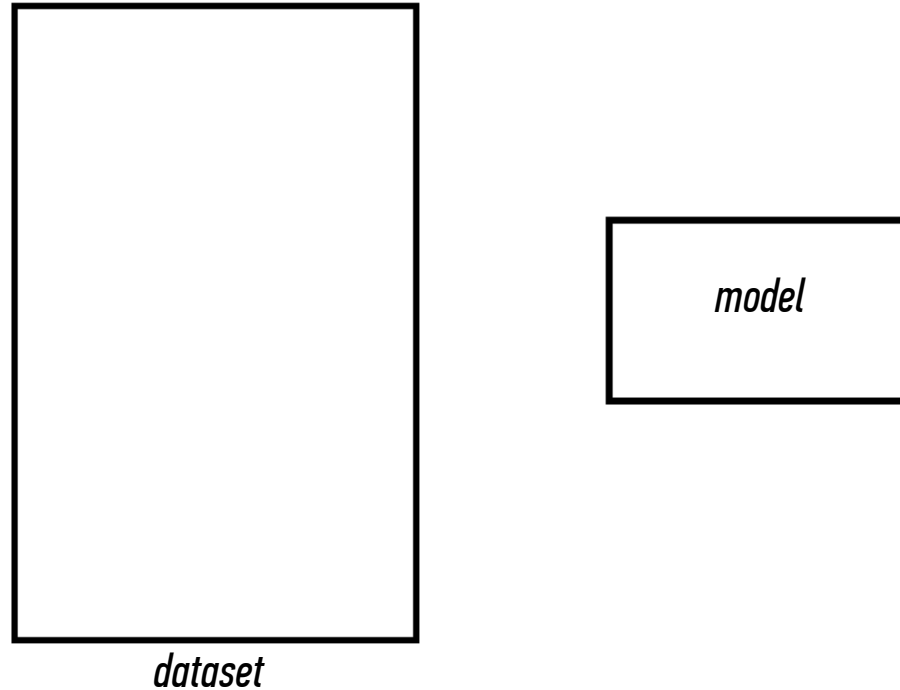


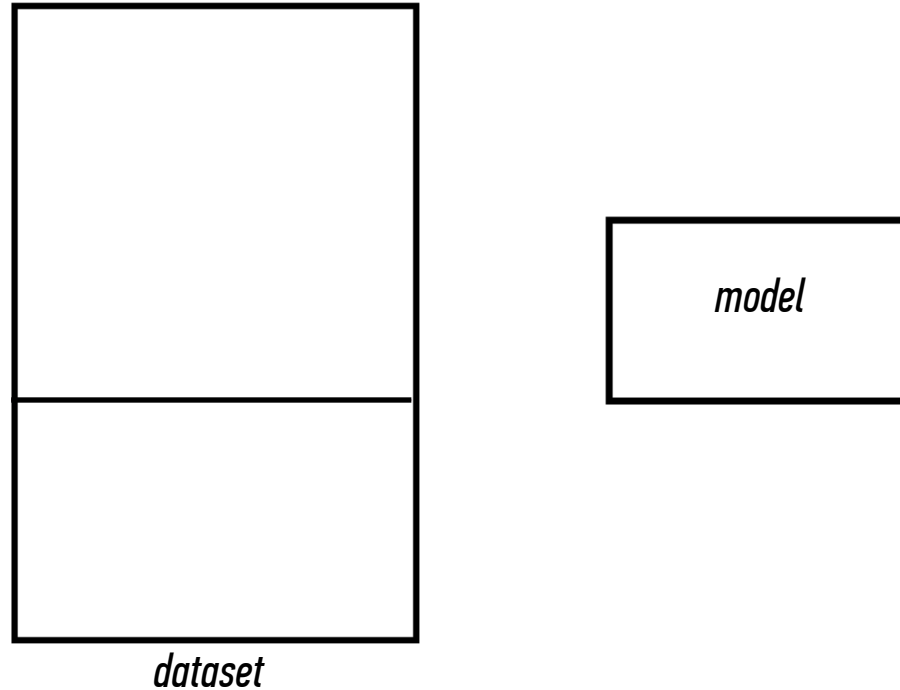
Figure 4.2. Classification as the task of mapping an input attribute set x into its class label y .

Q: What steps does a classification problem require?



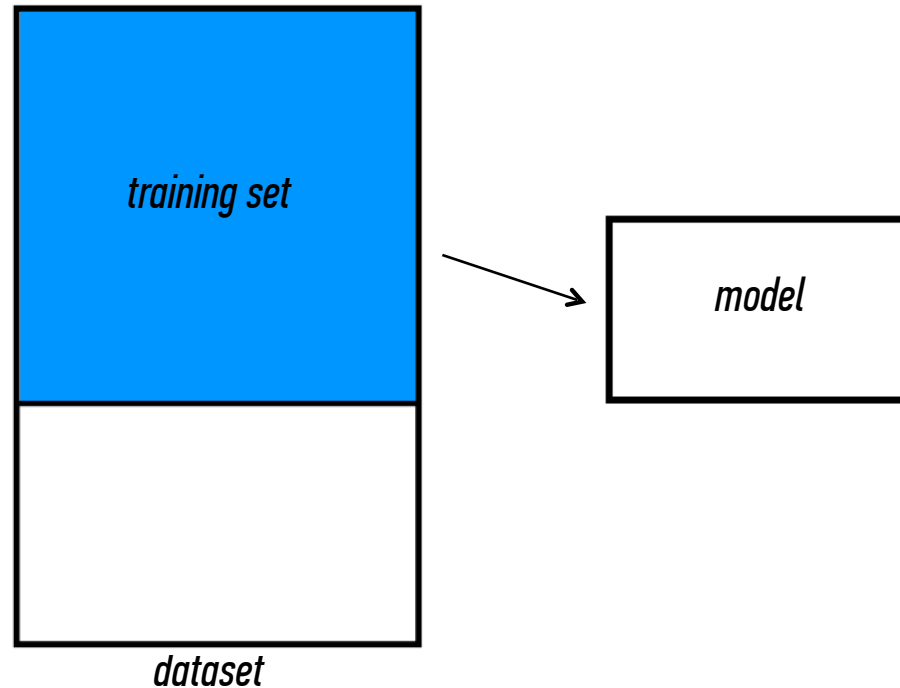
Q: What steps does a classification problem require?

1) split dataset



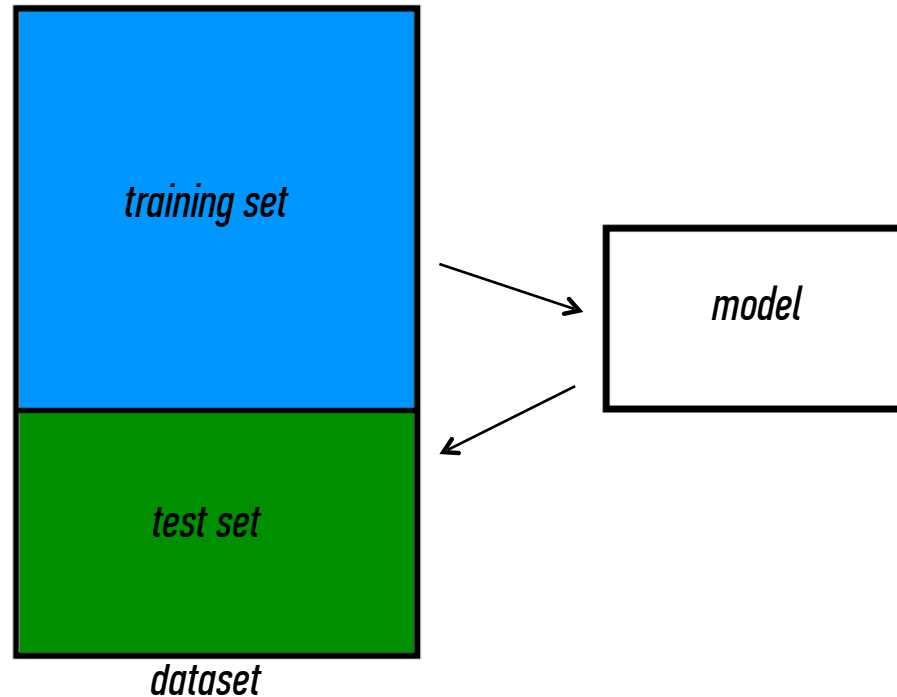
Q: What steps does a classification problem require?

- 1) split dataset*
- 2) train model*



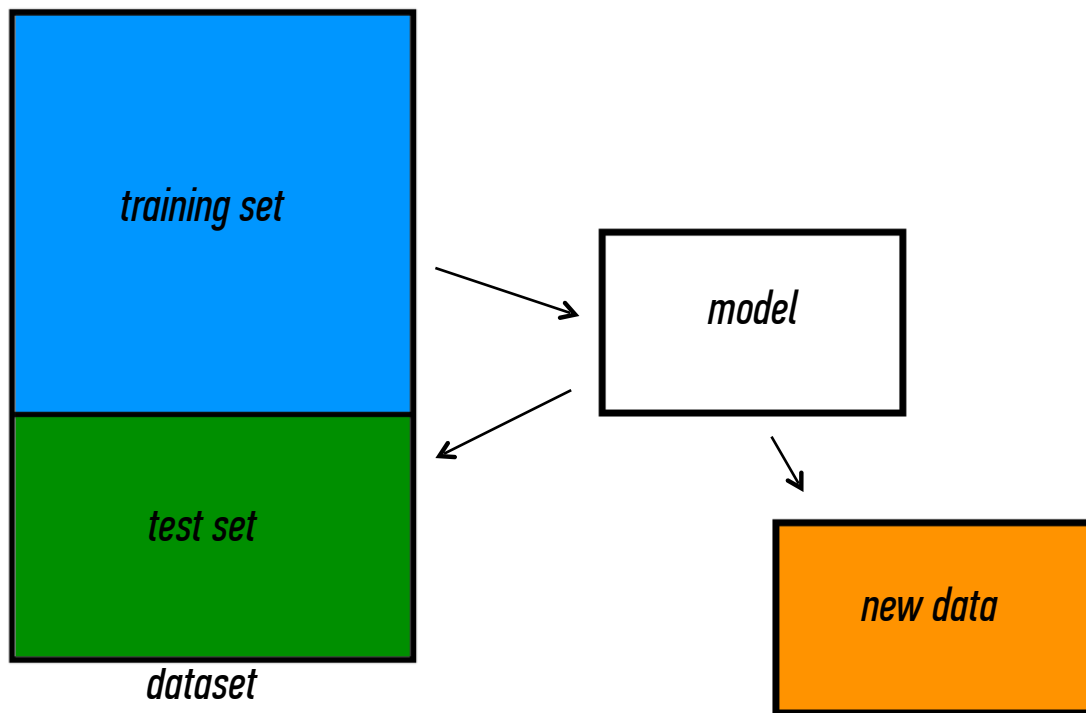
Q: What steps does a classification problem require?

- 1) split dataset*
- 2) train model*
- 3) test model*



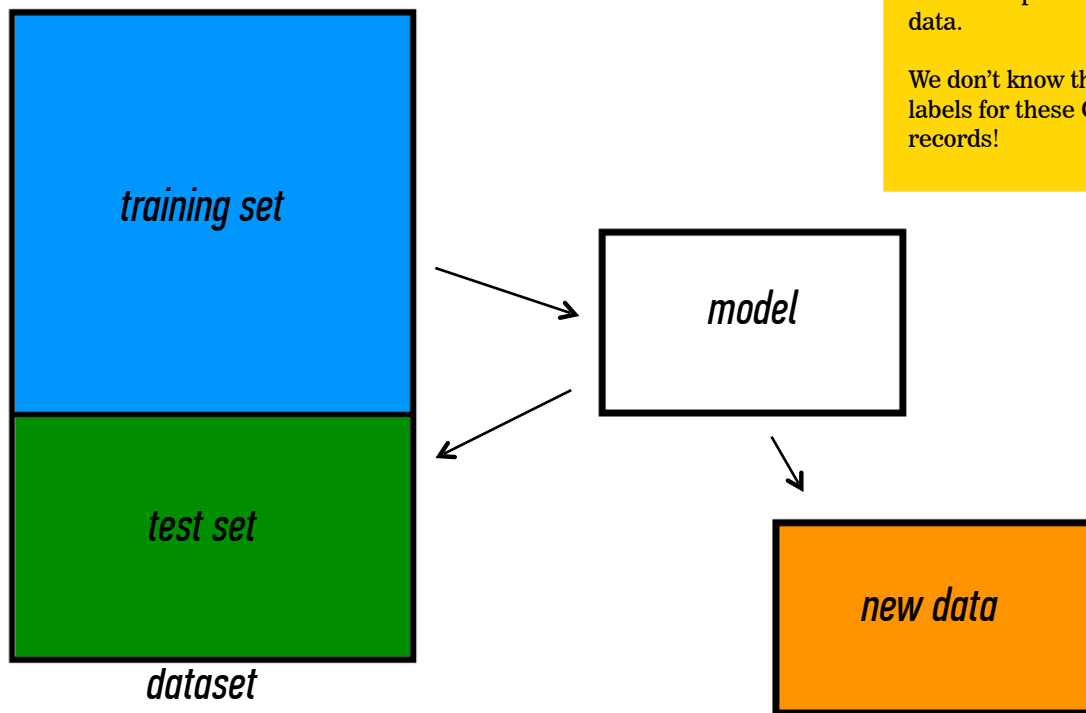
Q: What steps does a classification problem require?

- 1) split dataset*
- 2) train model*
- 3) test model*
- 4) make predictions*



Q: What steps does a classification problem require?

- 1) split dataset*
- 2) train model*
- 3) test model*
- 4) make predictions*



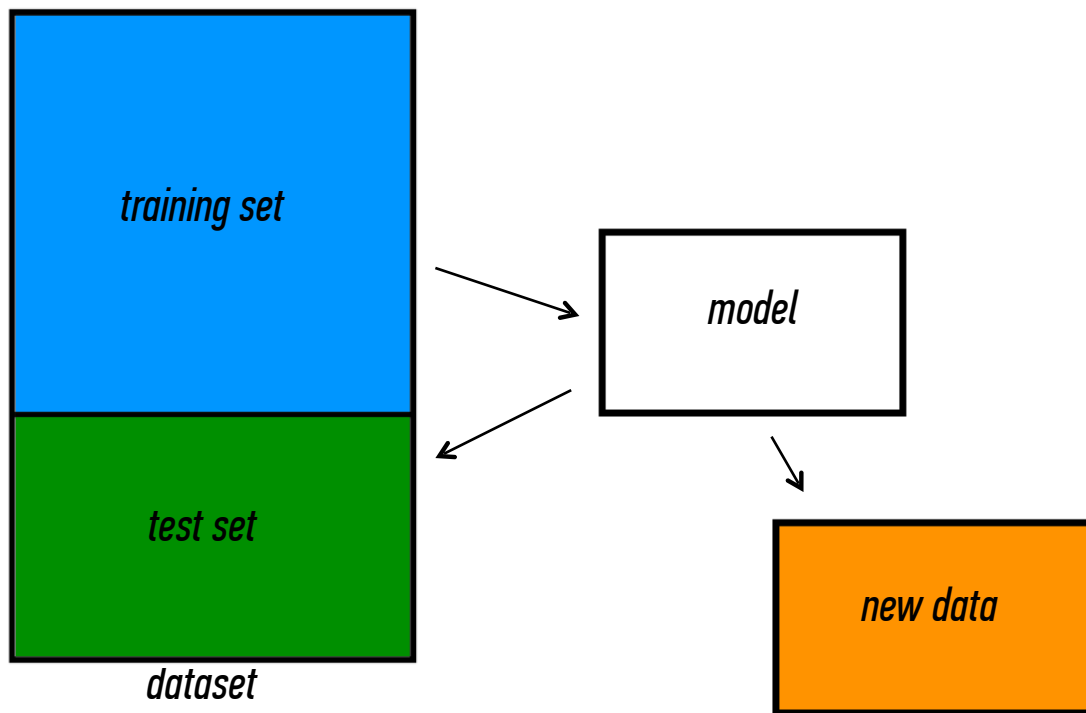
NOTE

This new data is called out of sample data.

We don't know the labels for these OOS records!

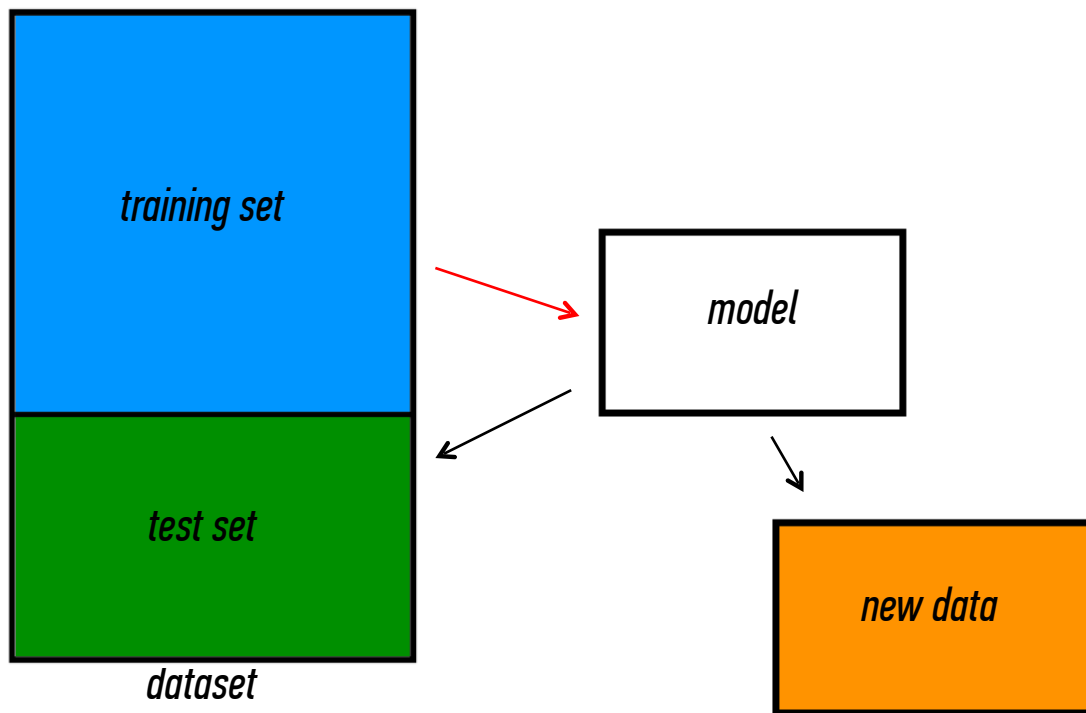
II. BUILDING EFFECTIVE CLASSIFIERS

Q: What types of prediction error will we run into?



Q: What types of prediction error will we run into?

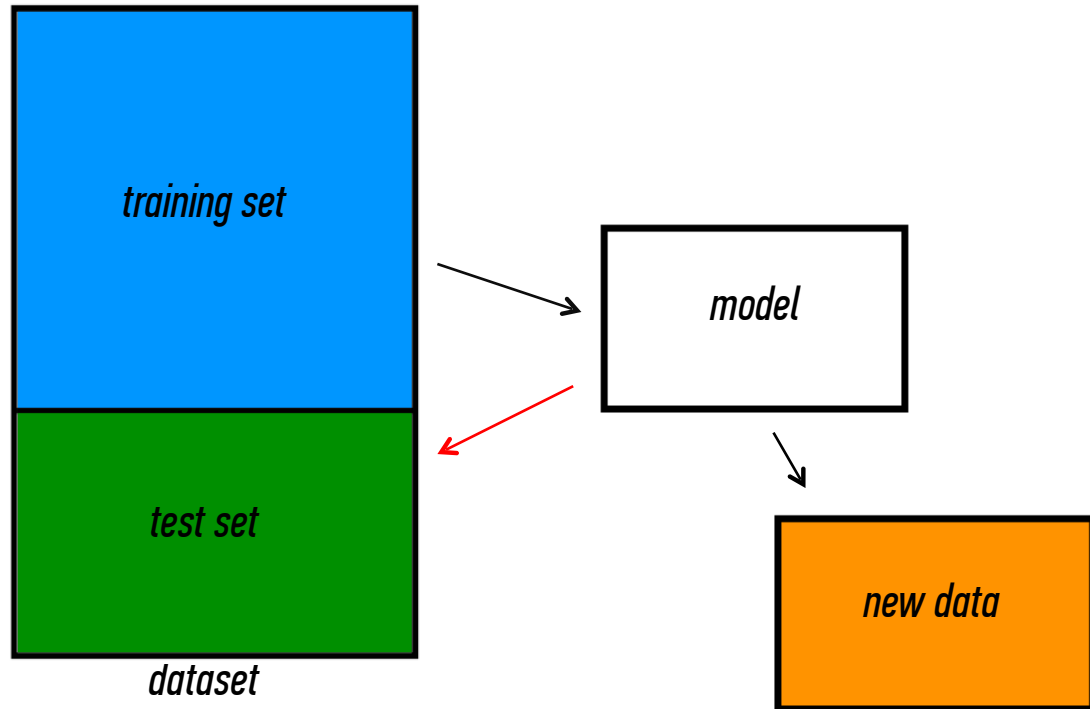
1) training error



Q: What types of prediction error will we run into?

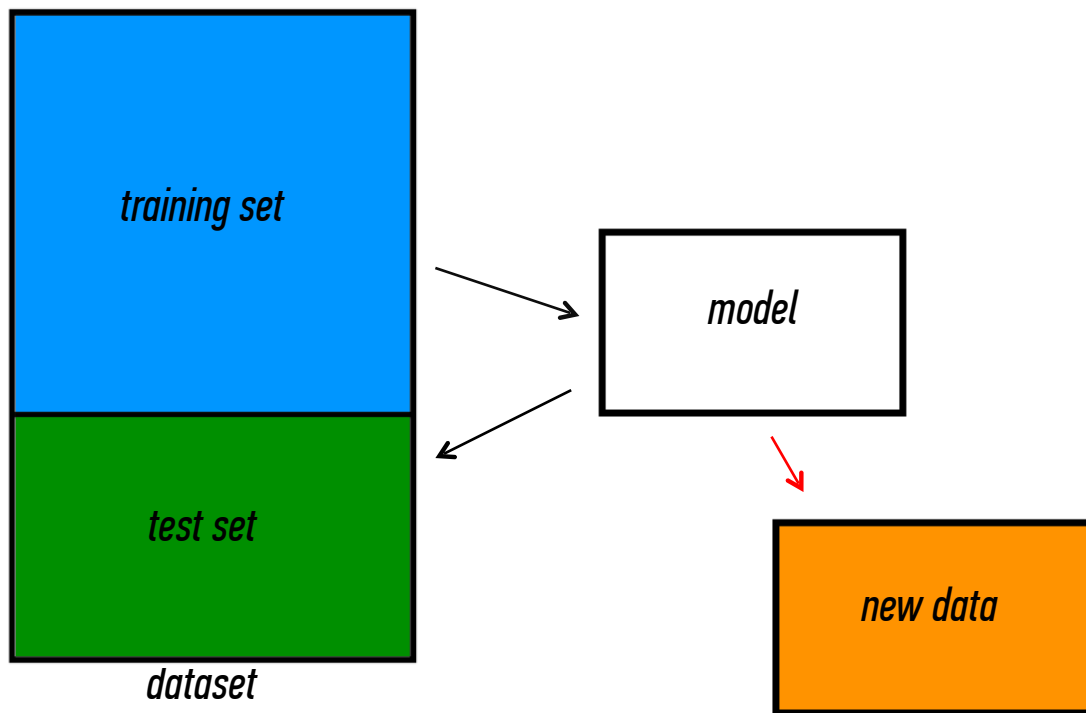
- 1) *training error*
- 2) *generalization error*

NOTE
measures how
well a learning
machine
generalizes to
unseen(test) data



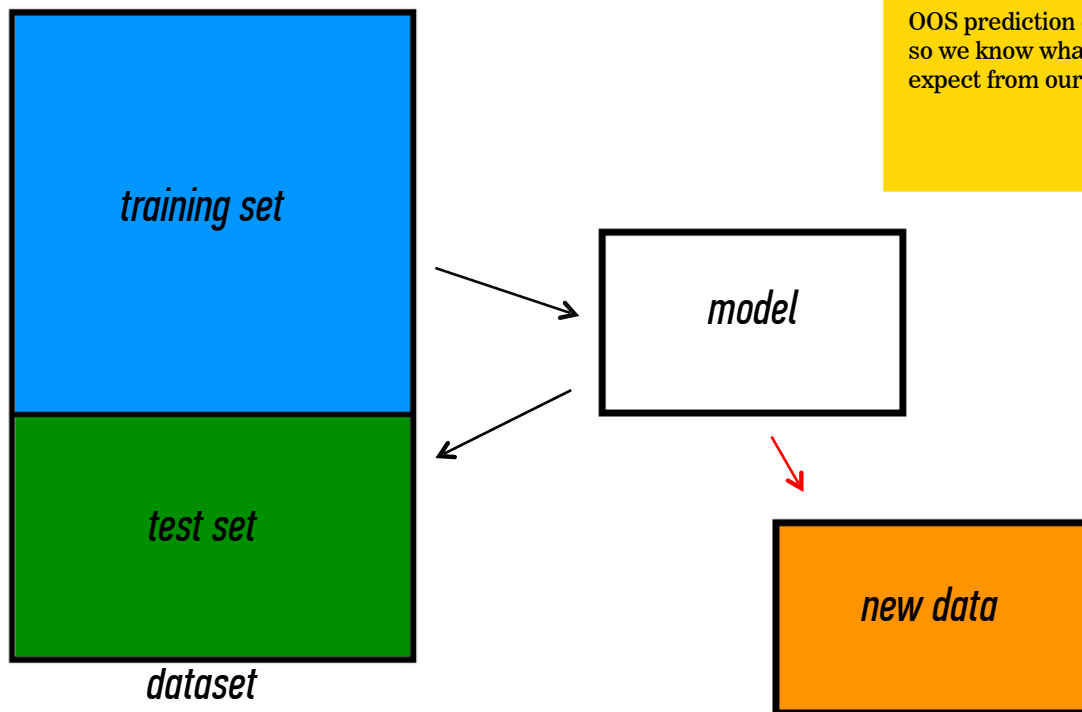
Q: What types of prediction error will we run into?

- 1) training error*
- 2) generalization error*
- 3) OOS error*



Q: What types of prediction error will we run into?

- 1) training error*
- 2) generalization error*
- 3) OOS error*



NOTE

We want to estimate OOS prediction error so we know what to expect from our model.

Q: Why should we use training & test sets?

Q: Why should we use training & test sets?

Thought experiment:

Suppose instead, we train our model using the entire dataset.

Q: Why should we use training & test sets?

Thought experiment:

Suppose instead, we train our model using the entire dataset.

Q: How low can we push the training error?

Q: Why should we use training & test sets?

Thought experiment:

Suppose instead, we train our model using the entire dataset.

Q: How low can we push the training error?

- We can make the model arbitrarily complex (effectively “memorizing” the entire training set).*

Q: Why should we use training & test sets?

Thought experiment:

Suppose instead, we train our model using the entire dataset.

Q: How low can we push the training error?

- We can make the model arbitrarily complex (effectively “memorizing” the entire training set).*

A: Down to zero!

Q: Why should we use training & test sets?

Thought experiment:

Suppose instead, we train our model using the entire dataset.

Q: How low can we push the training error?

- We can make the model arbitrarily complex (effectively “memorizing” the entire training set).*

A: Down to zero!

NOTE

This phenomenon is called overfitting.

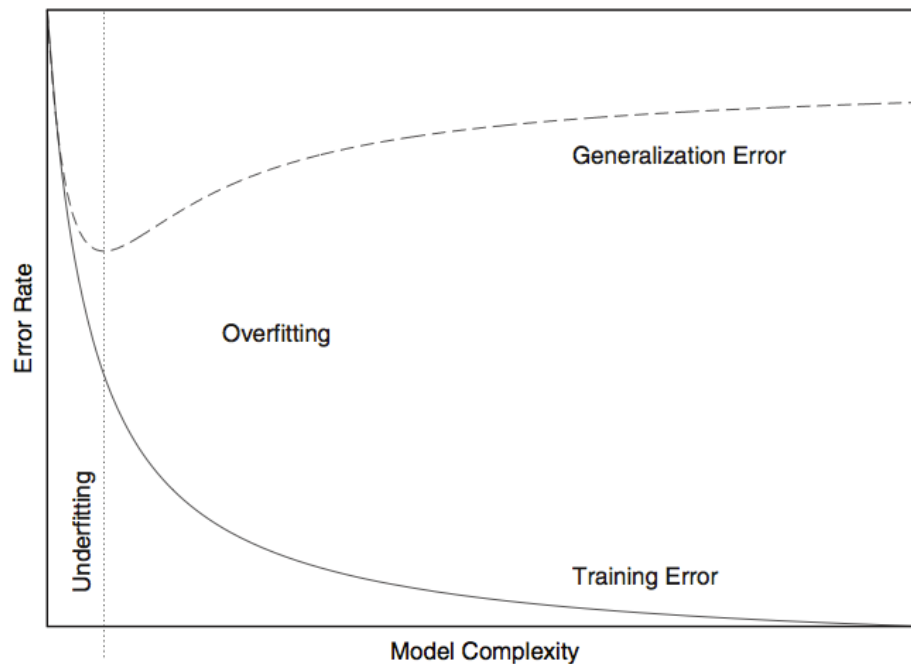
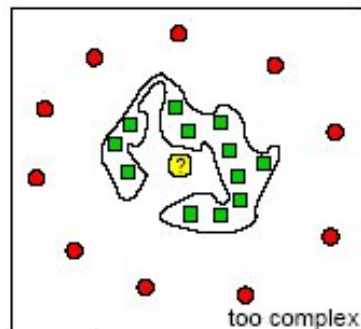
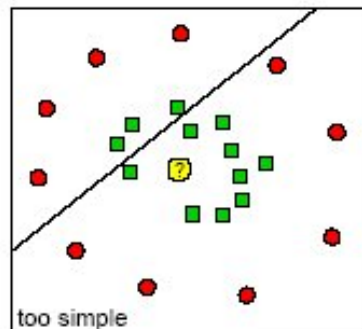


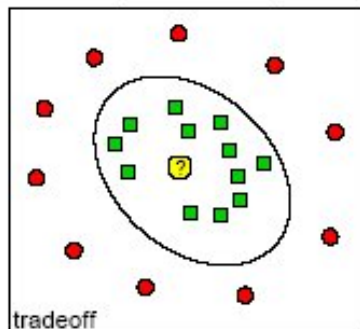
FIGURE 18-1. Overfitting: as a model becomes more complex, it becomes increasingly able to represent the training data. However, such a model is overfitted and will not generalize well to data that was not used during training.

OVERFITTING - EXAMPLE

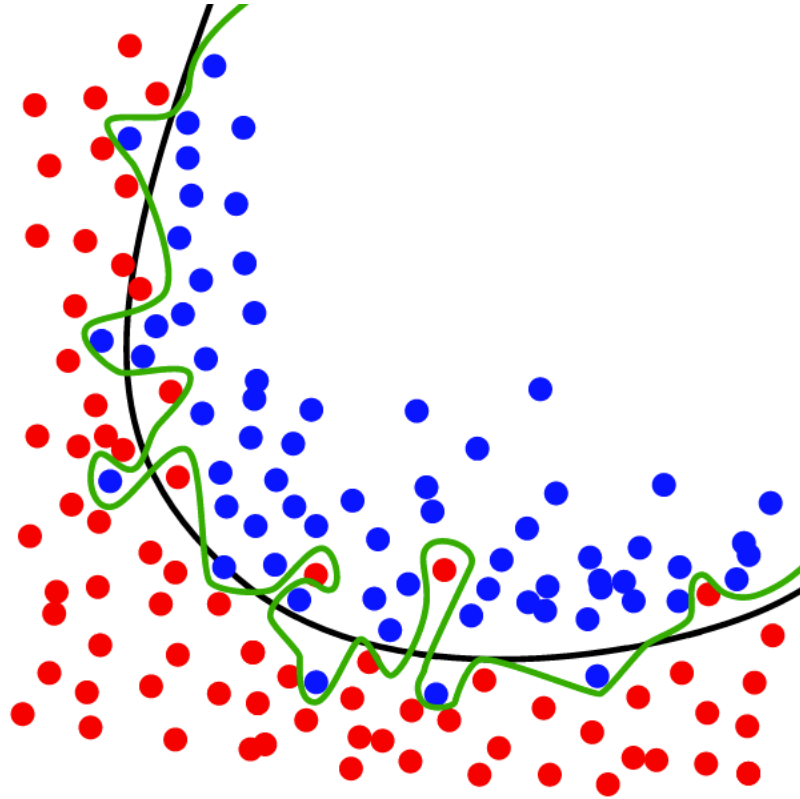
Underfitting and Overfitting



- negative example
- positive example
- ⓪ new patient



OVERFITTING - EXAMPLE



Q: Why should we use training & test sets?

Thought experiment:

Suppose instead, we train our model using the entire dataset.

Q: How low can we push the training error?

- We can make the model arbitrarily complex (effectively “memorizing” the entire training set).*

A: Down to zero!

A: Training error is not a good estimate of OOS accuracy.

NOTE

This phenomenon is called overfitting.

GENERALIZATION ERROR

Suppose we do the train/test split.

Suppose we do the train/test split.

Q: How well does generalization error predict OOS accuracy?

Suppose we do the train/test split.

Q: How well does generalization error predict OOS accuracy?

Thought experiment:

Suppose we had done a different train/test split.

Suppose we do the train/test split.

Q: How well does generalization error predict OOS accuracy?

Thought experiment:

Suppose we had done a different train/test split.

Q: Would the generalization error remain the same?

Suppose we do the train/test split.

Q: How well does generalization error predict OOS accuracy?

Thought experiment:

Suppose we had done a different train/test split.

Q: Would the generalization error remain the same?

A: Of course not!

Suppose we do the train/test split.

Q: How well does generalization error predict OOS accuracy?

Thought experiment:

Suppose we had done a different train/test split.

Q: Would the generalization error remain the same?

A: Of course not!

A: On its own, not very well.

Suppose we do the train/test split.

Q: How well does generalization error predict OOS accuracy?

Thought experiment:

Suppose we had done a different train/test split.

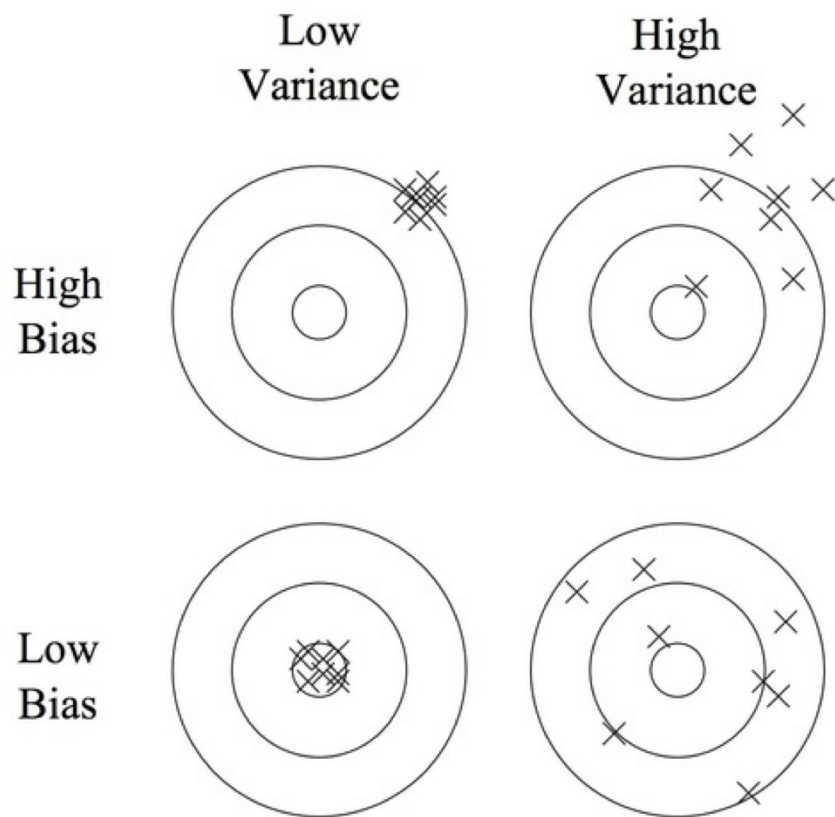
Q: Would the generalization error remain the same?

A: Of course not!

A: On its own, not very well.

NOTE

The generalization error gives a high-variance estimate of OOS accuracy.



GENERALIZATION ERROR

Something is still missing!

GENERALIZATION ERROR

Something is still missing!

Q: How can we do better?

Something is still missing!

Q: How can we do better?

Thought experiment:

Different train/test splits will give us different generalization errors.

Something is still missing!

Q: How can we do better?

Thought experiment:

Different train/test splits will give us different generalization errors.

Q: What if we did a bunch of these and took the average?

Something is still missing!

Q: How can we do better?

Thought experiment:

Different train/test splits will give us different generalization errors.

Q: What if we did a bunch of these and took the average?

A: Now you're talking!

Something is still missing!

Q: How can we do better?

Thought experiment:

Different train/test splits will give us different generalization errors.

Q: What if we did a bunch of these and took the average?

A: Now you're talking!

A: Cross-validation.

CROSS-VALIDATION

Steps for n -fold cross-validation:

Steps for n -fold cross-validation:

- 1) Randomly split the dataset into n equal partitions.*

Steps for n -fold cross-validation:

- 1) Randomly split the dataset into n equal partitions.*
- 2) Use partition 1 as test set & union of other partitions as training set.*

Steps for n -fold cross-validation:

- 1) Randomly split the dataset into n equal partitions.*
- 2) Use partition 1 as test set & union of other partitions as training set.*
- 3) Find generalization error.*

Steps for n -fold cross-validation:

- 1) Randomly split the dataset into n equal partitions.*
- 2) Use partition 1 as test set & union of other partitions as training set.*
- 3) Find generalization error.*
- 4) Repeat steps 2-3 using a different partition as the test set at each iteration.*

Steps for n -fold cross-validation:

- 1) Randomly split the dataset into n equal partitions.*
- 2) Use partition 1 as test set & union of other partitions as training set.*
- 3) Find generalization error.*
- 4) Repeat steps 2-3 using a different partition as the test set at each iteration.*
- 5) Take the average generalization error as the estimate of OOS accuracy.*

CROSS-VALIDATION: 5-FOLD EXAMPLE

Dataset	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	<u>Accuracy</u>
1	Test	Train	Train	Train	Train	$k_1 \%$
2	Train	Test	Train	Train	Train	$k_2 \%$
3	Train	Train	Test	Train	Train	$k_3 \%$
4	Train	Train	Train	Test	Train	$k_4 \%$
5	Train	Train	Train	Train	Test	$k_5 \%$

overall accuracy: $(k_1 + k_2 + k_3 + k_4 + k_5) / 5$

Features of n -fold cross-validation:

Features of n -fold cross-validation:

- 1) More accurate estimate of OOS prediction error.*

Features of n -fold cross-validation:

- 1) More accurate estimate of OOS prediction error.*
- 2) More efficient use of data than single train/test split.*
 - Each record in our dataset is used for both training and testing.*

Features of n -fold cross-validation:

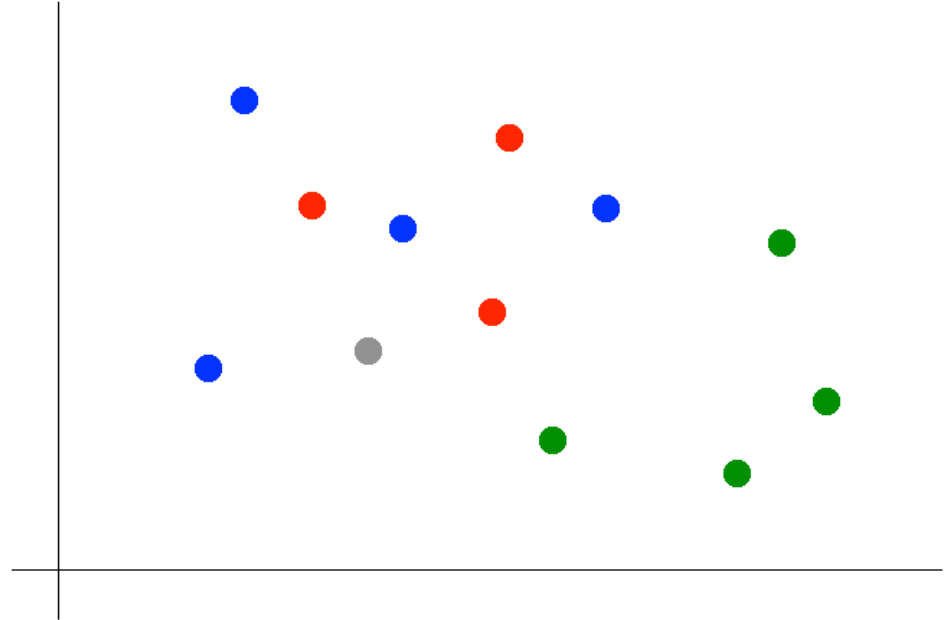
- 1) More accurate estimate of OOS prediction error.*
- 2) More efficient use of data than single train/test split.*
 - Each record in our dataset is used for both training and testing.*
- 3) Presents tradeoff between efficiency and computational expense.*
 - 10-fold CV is 10x more expensive than a single train/test split*

Features of n-fold cross-validation:

- 1) More accurate estimate of OOS prediction error.*
- 2) More efficient use of data than single train/test split.*
 - Each record in our dataset is used for both training and testing.*
- 3) Presents tradeoff between efficiency and computational expense.*
 - 10-fold CV is 10x more expensive than a single train/test split*
- 4) Can be used for model selection.*

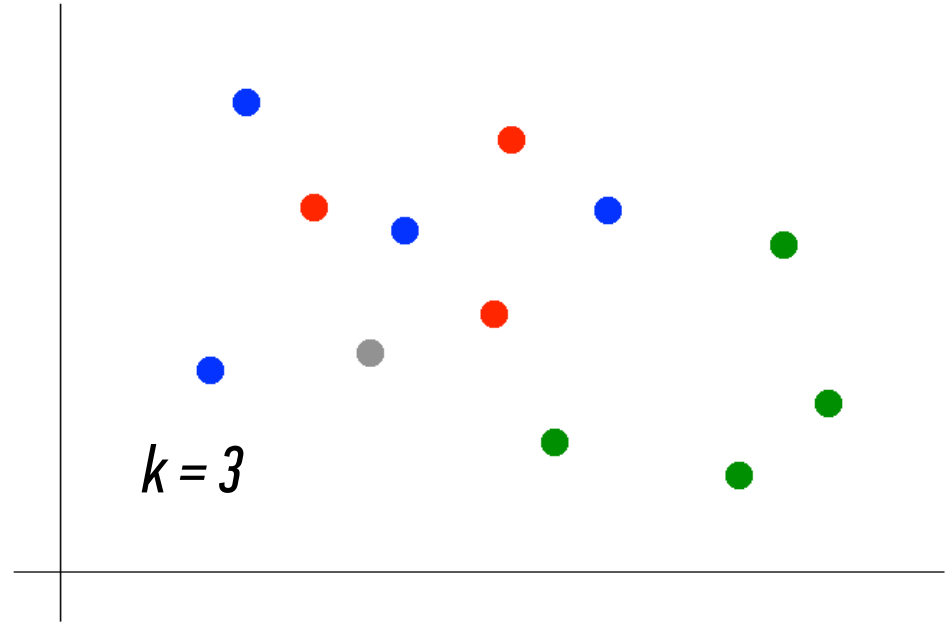
III. KNN CLASSIFICATION

Suppose we want to predict the color of the grey dot.



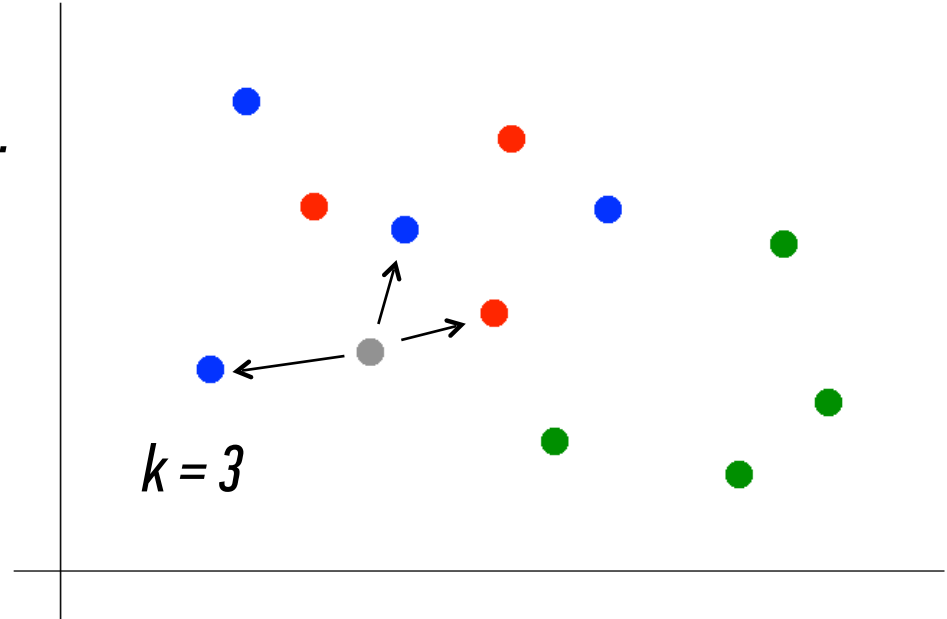
Suppose we want to predict the color of the grey dot.

1) Pick a value for k .



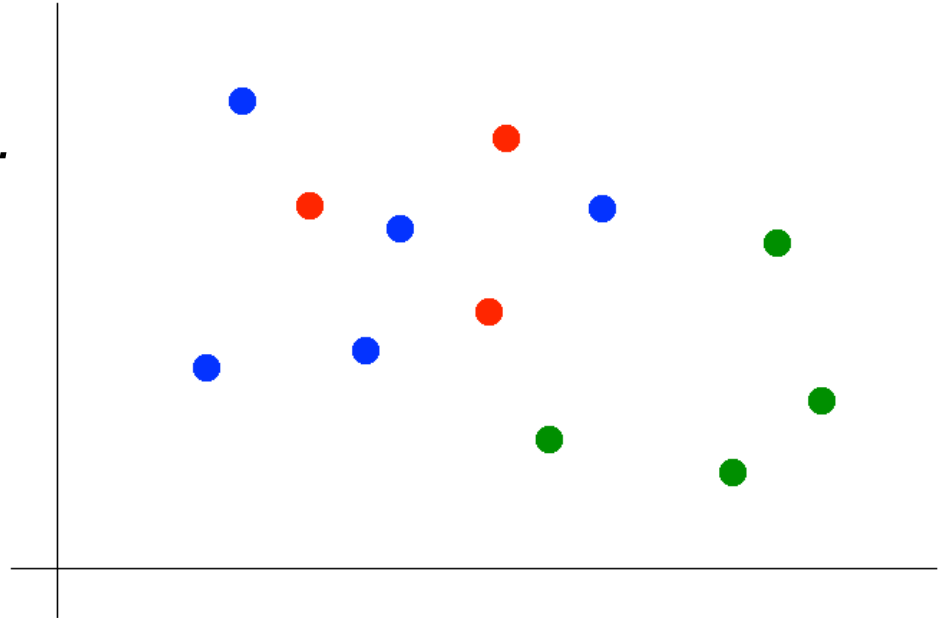
Suppose we want to predict the color of the grey dot.

- 1) Pick a value for k .*
- 2) Find colors of k nearest neighbors.*



Suppose we want to predict the color of the grey dot.

- 1) Pick a value for k .*
- 2) Find colors of k nearest neighbors.*
- 3) Assign the most common color to the grey dot.*

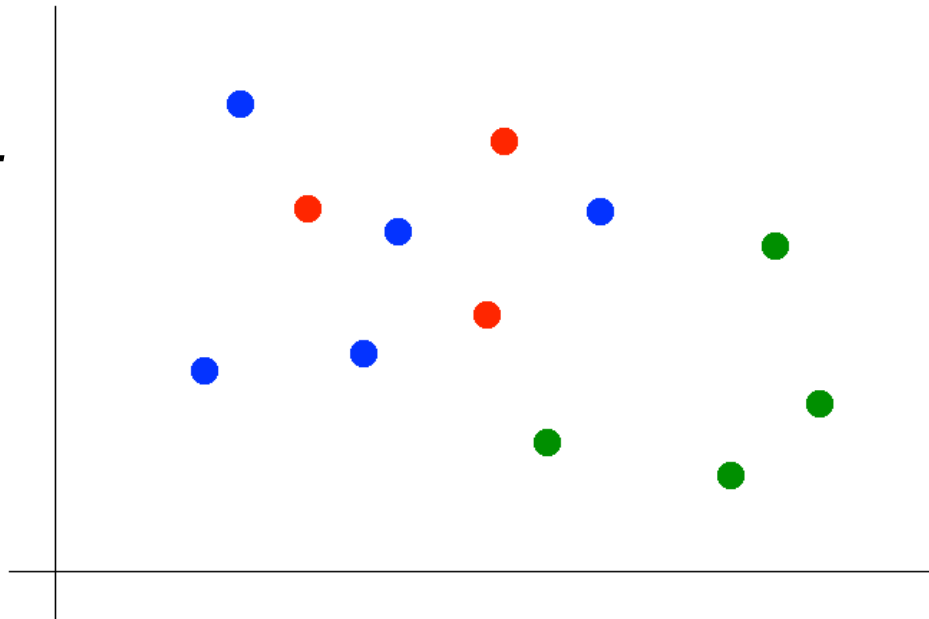


Suppose we want to predict the color of the grey dot.

- 1) Pick a value for k .*
- 2) Find colors of k nearest neighbors.*
- 3) Assign the most common color to the grey dot.*

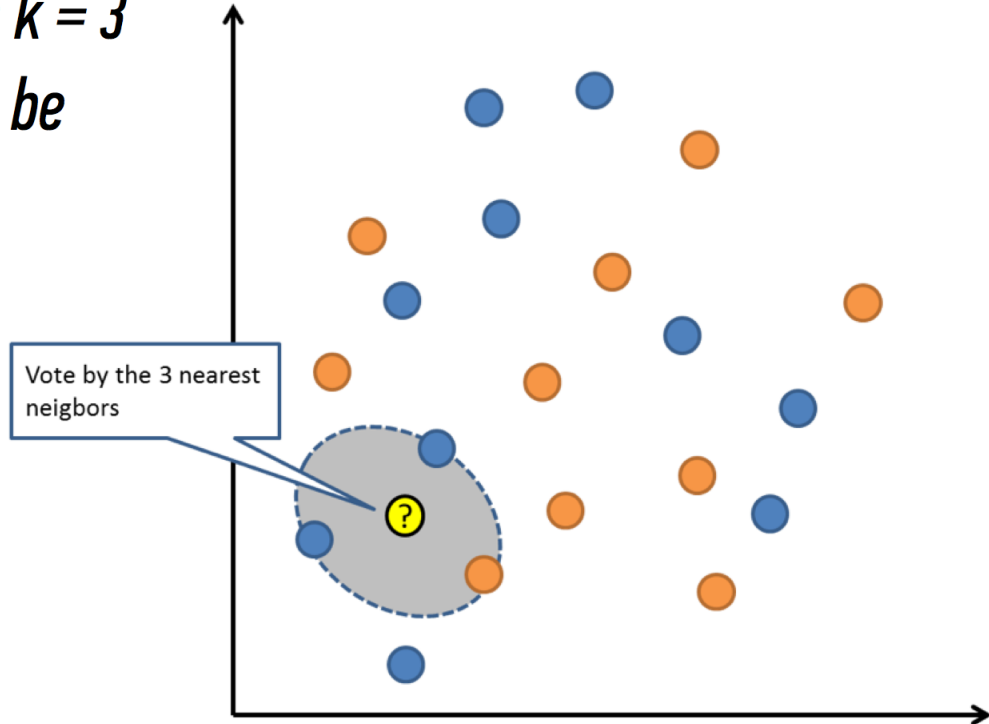
OPTIONAL NOTE

Our definition of “nearest” implicitly uses the Euclidean distance function.

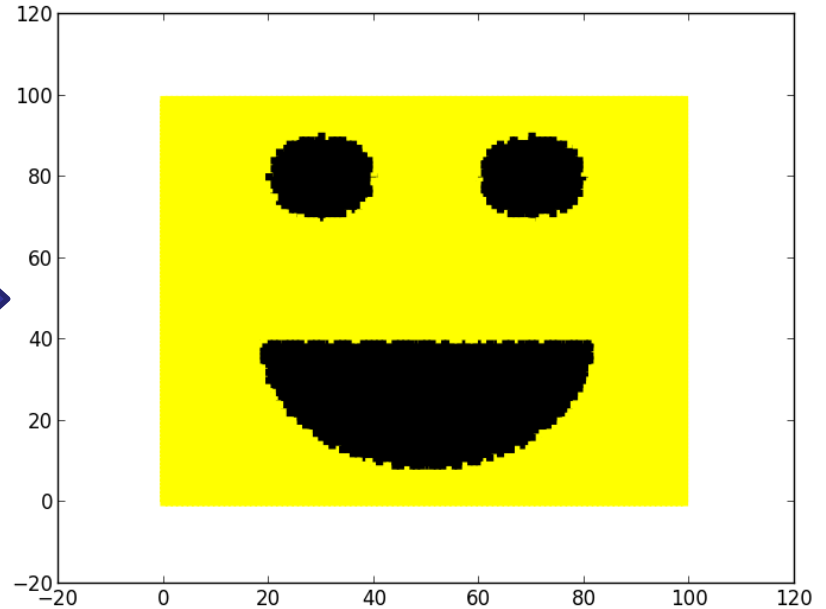
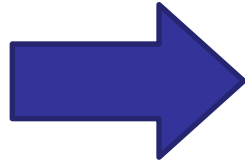
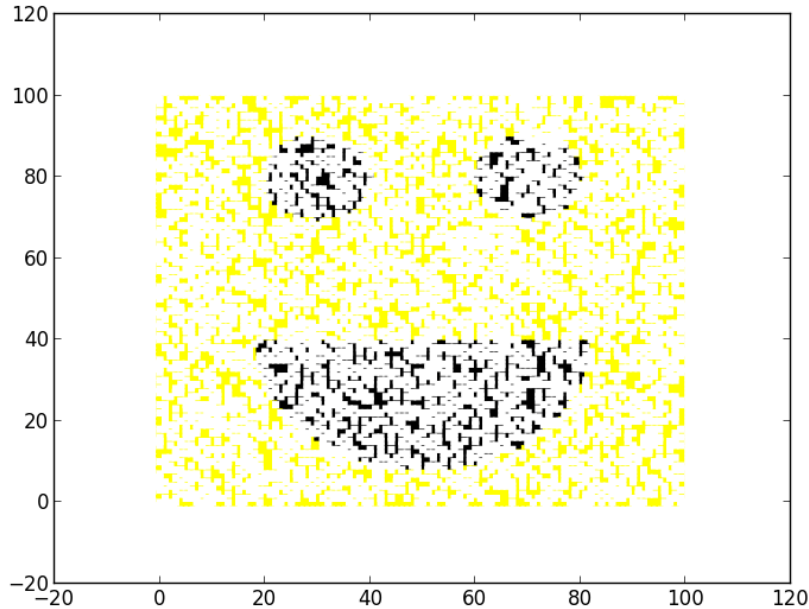


K NEAREST NEIGHBORS CLASSIFICATION

*Another example with $k = 3$
Will our new example be
blue or orange?*



fixing a broken image



*In theory, if **infinite** number of samples available, the larger is k , the better is classification*

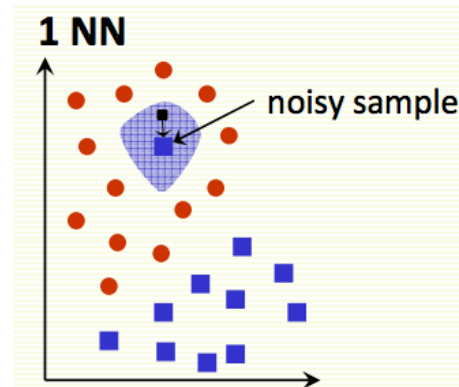
*In theory, if **infinite** number of samples available, the larger is k , the better is classification*

The caveat is that all k neighbors have to be close

- Possible when infinite # samples available*
- Impossible in practice since # samples is finite*

Rule of thumb is $k < \sqrt{n}$, n is number of examples

- interesting theoretical properties*
- In practice, $k = 1$ is often used for efficiency, but can be sensitive to “noise”*



KNN CLASSIFICATION - HOW TO CHOOSE K?

*larger k may improve performance, but too large k destroys locality,
i.e. end up looking at samples that are not neighbors*

cross-validation may be used to choose k



Advantages

- *Can be applied to the data from any distribution*



Advantages

- *Can be applied to the data from any distribution*
- *Very simple and intuitive*



Advantages

- *Can be applied to the data from any distribution*
- *Very simple and intuitive*
- *Good classification if the number of samples is large enough*



Advantages

- *Can be applied to the data from any distribution*
- *Very simple and intuitive*
- *Good classification if the number of samples is large enough*



Disadvantages

- *Choosing k may be tricky*



Advantages

- *Can be applied to the data from any distribution*
- *Very simple and intuitive*
- *Good classification if the number of samples is large enough*



Disadvantages

- *Choosing k may be tricky*
- *Test stage is computationally expensive (no training stage)*



Advantages

- *Can be applied to the data from any distribution*
- *Very simple and intuitive*
- *Good classification if the number of samples is large enough*



Disadvantages

- *Choosing k may be tricky*
- *Test stage is computationally expensive (no training stage)*
- *Need large number of samples for accuracy*

RECAP: KNN – WHAT ARE THE PROS AND CONS?



Advantages

- *Can be applied to the data from any distribution*
- *Very simple and intuitive*
- *Good classification if the number of samples is large enough*



Disadvantages

- *Choosing k may be tricky*
- *Test stage is computationally expensive (no training stage)*
- *Need large number of samples for accuracy*

INTRO TO DATA SCIENCE

LAB