GENERAL ASSEMBLY

INTRO TO DATA SCIENCE

LECTURE 7: KNN CLASSIFICATION

LAST TIME:

- POLYNOMIAL REGRESSION
- LOGISTIC REGRESSION
- REGULARIZATION

QUESTIONS?

I. CLASSIFICATION PROBLEMS

II. BUILDING EFFECTIVE CLASSIFIERS

EXERCISES:

III. THE KNN CLASSIFICATION MODEL

I. CLASSIFICATION PROBLEMS

	continuous	*categorical*
supervised	???	???
unsupervised	???	???

	continuous	categorical
supervised	regression	classification
unsupervised	dimension reduction	clustering

Here's (part of) an example dataset (Fisher's Iris Data Set)

Fisher's *Iris* Data

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

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. setosa*
4.9	3.0	1.4	0.2	*I. setosa*
4.7	3.2	1.3	0.2	*I. setosa*
4.6	3.1	1.5	0.2	*I. setosa*
5.0	3.6	1.4	0.2	*I. setosa*
5.4	3.9	1.7	0.4	*I. setosa*
4.6	3.4	1.4	0.3	*I. setosa*
5.0	3.4	1.5	0.2	*I. setosa*

independent variables

Here's (part of) an example dataset (Fisher's Iris Data Set):

Fisher's *Iris* Data

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

class labels (qualitative)

independent variables

Q: What does "supervised" mean?

Q: What does "supervised" mean?

A: We know the labels.

```
>>> from sklearn import datasets
>>> iris = datasets.load_iris()
>>> iris.target_names
array(['setosa', 'versicolor', 'virginica'],
      dtype='|S10')
>>> iris.feature_names
['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal width (cm)']
>>> iris.data
array([[ 5.1,  3.5,  1.4,  0.2],
       [ 4.9,  3. ,  1.4,  0.2],
       [ 4.7,  3.2,  1.3,  0.2],
       [ 4.6,  3.1,  1.5,  0.2],
       [ 5. ,  3.6,  1.4,  0.2],
```

Q: How does a classification problem work?

Q: How does a classification problem work?

A: Data in, predicted labels out.

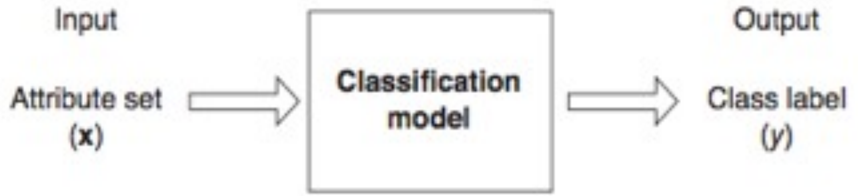
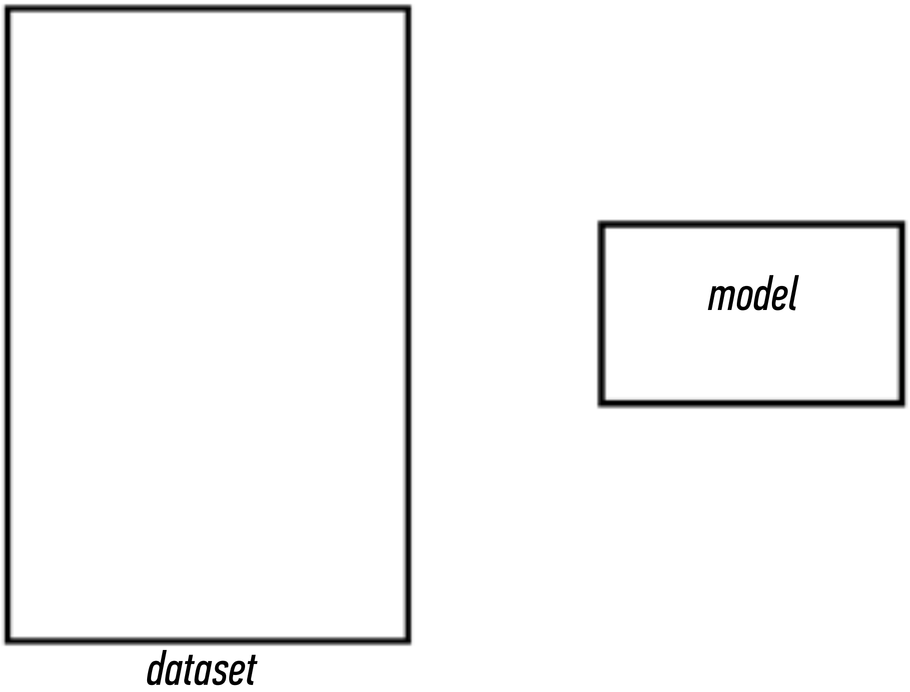


Figure 4.2. Classification as the task of mapping an input attribute set $\mathbf{x}$ into its class label y.

source: http://www-users.cs.umn.edu/~kumar/dmbook/ch4.pdf

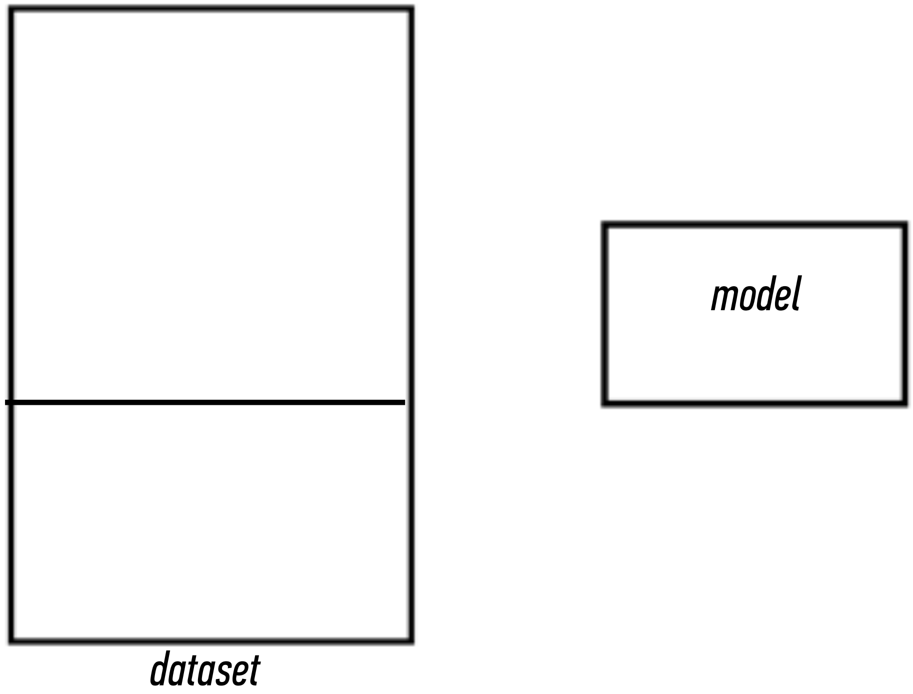
CLASSIFICATION PROBLEMS

Q: What steps does a classification problem require?



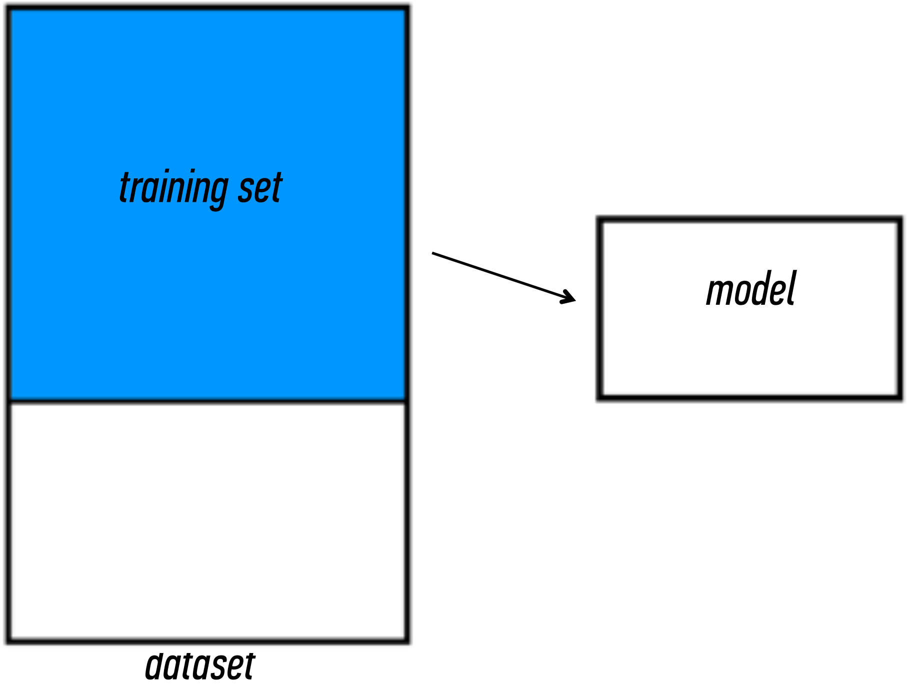
Q: What steps does a classification problem require?

1) *split dataset*



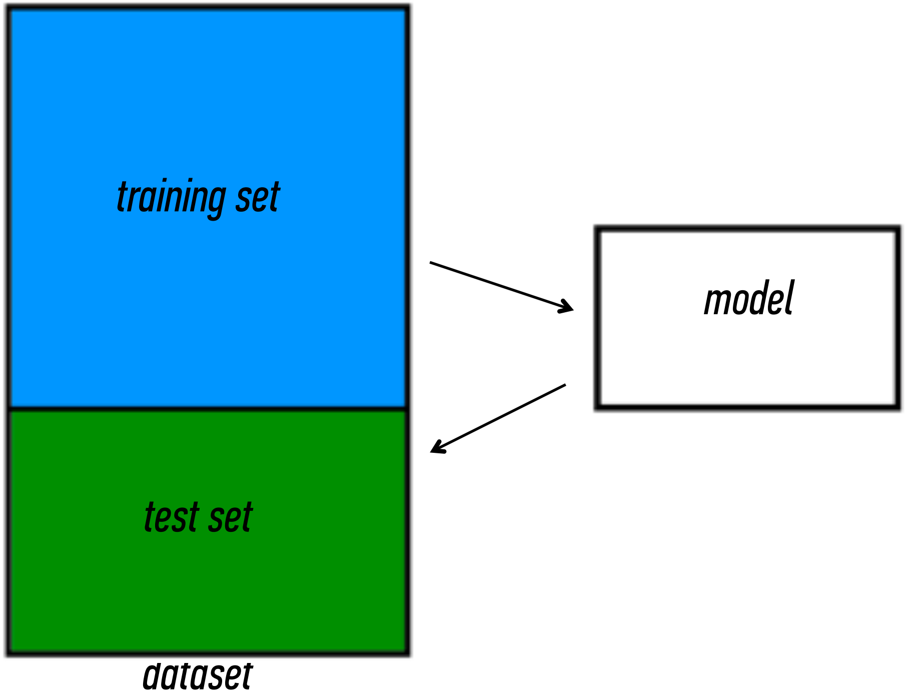
Q: What steps does a classification problem require?

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



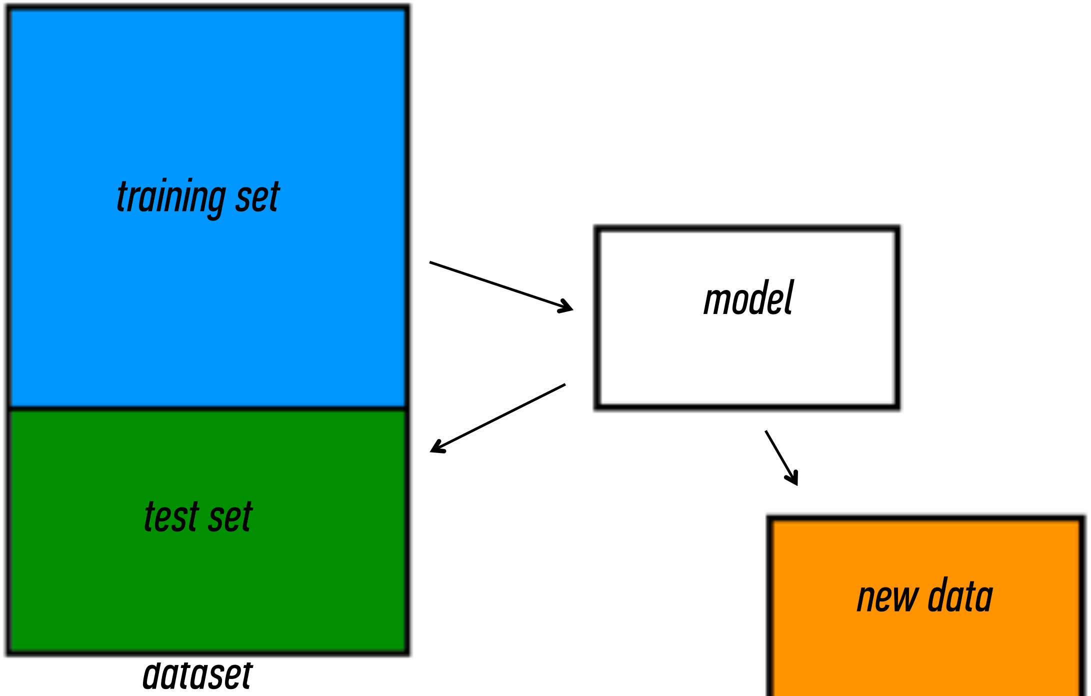
Q: What steps does a classification problem require?

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



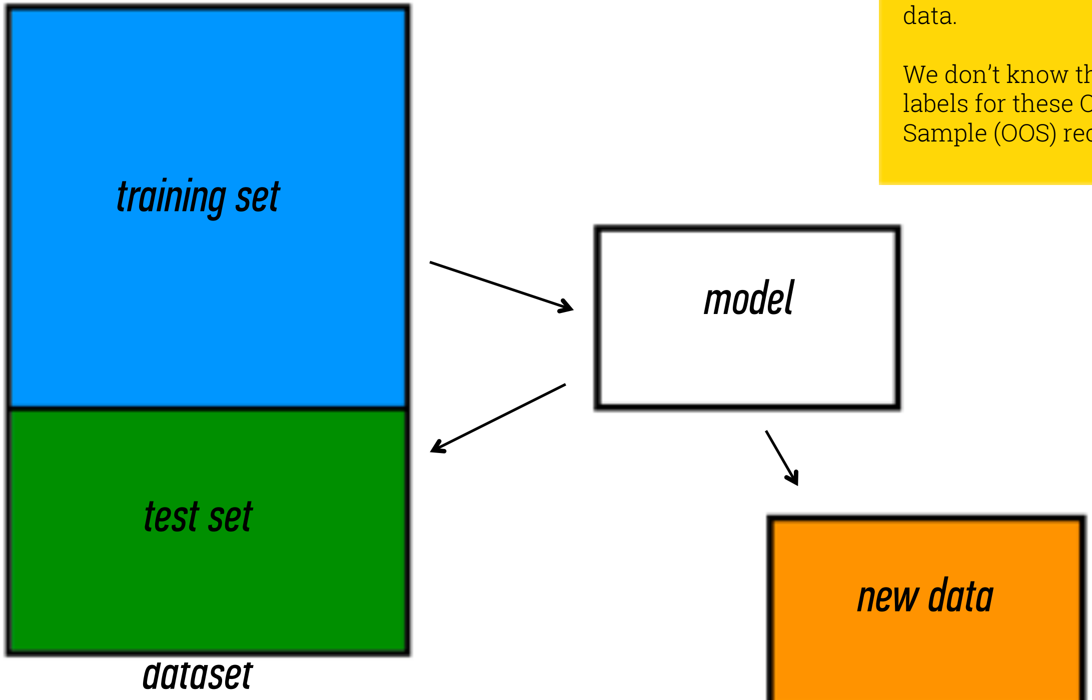
Q: What steps does a classification problem require?

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*



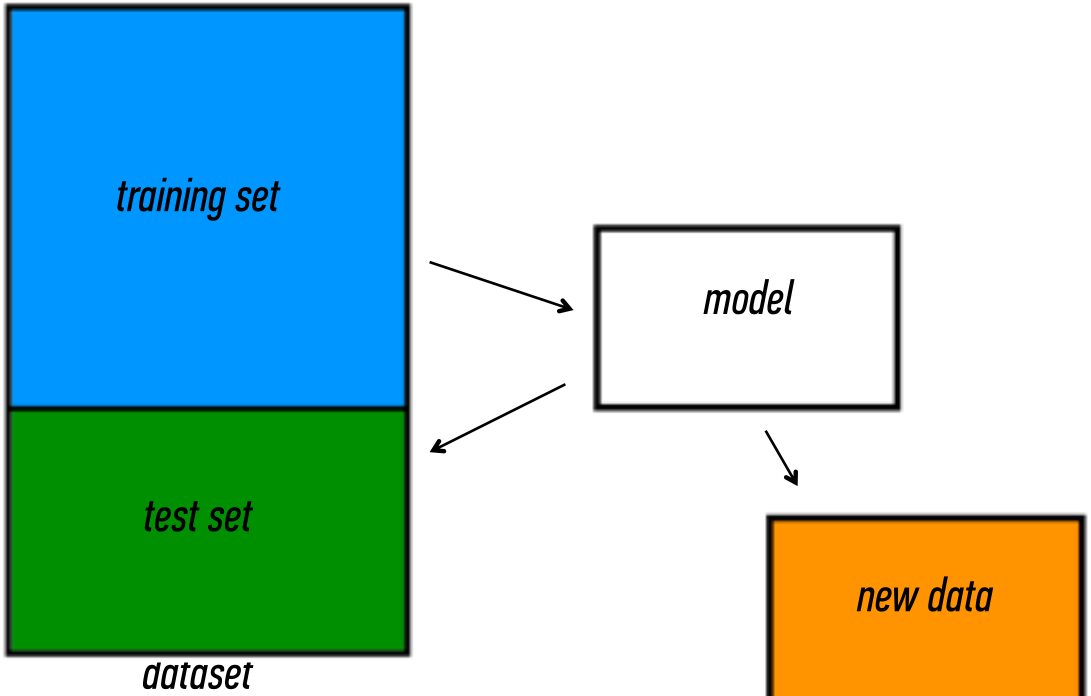
NOTE

This new data is called out of sample data.

We don't know the labels for these Out of Sample (OOS) records!

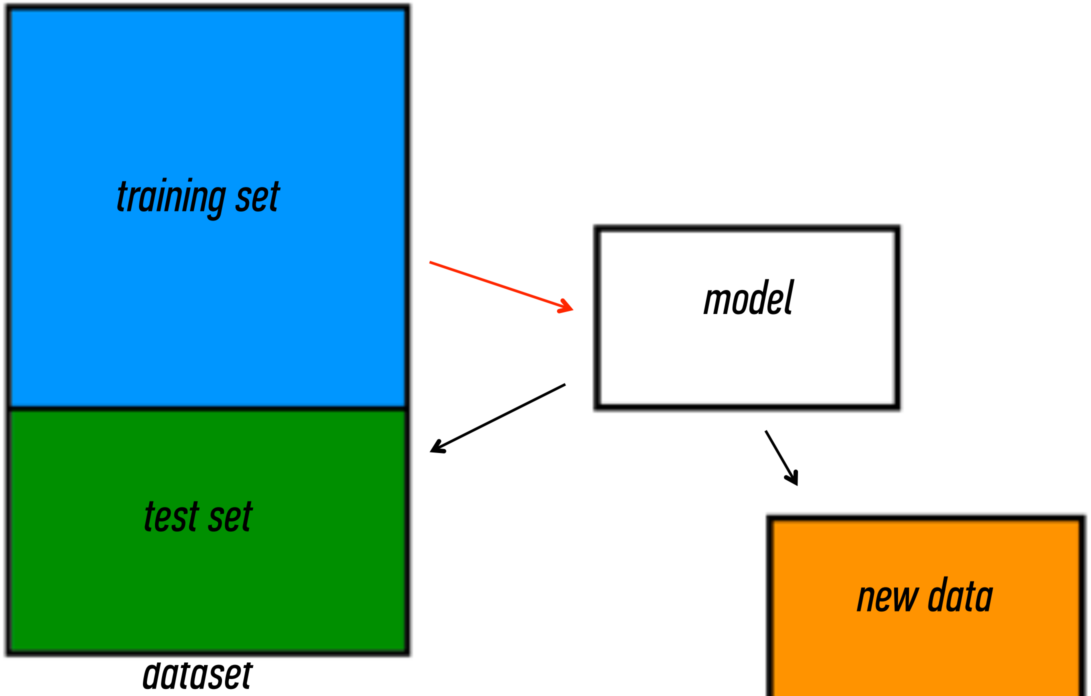
II. BUILDING EFFECTIVE CLASSIFIERS

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



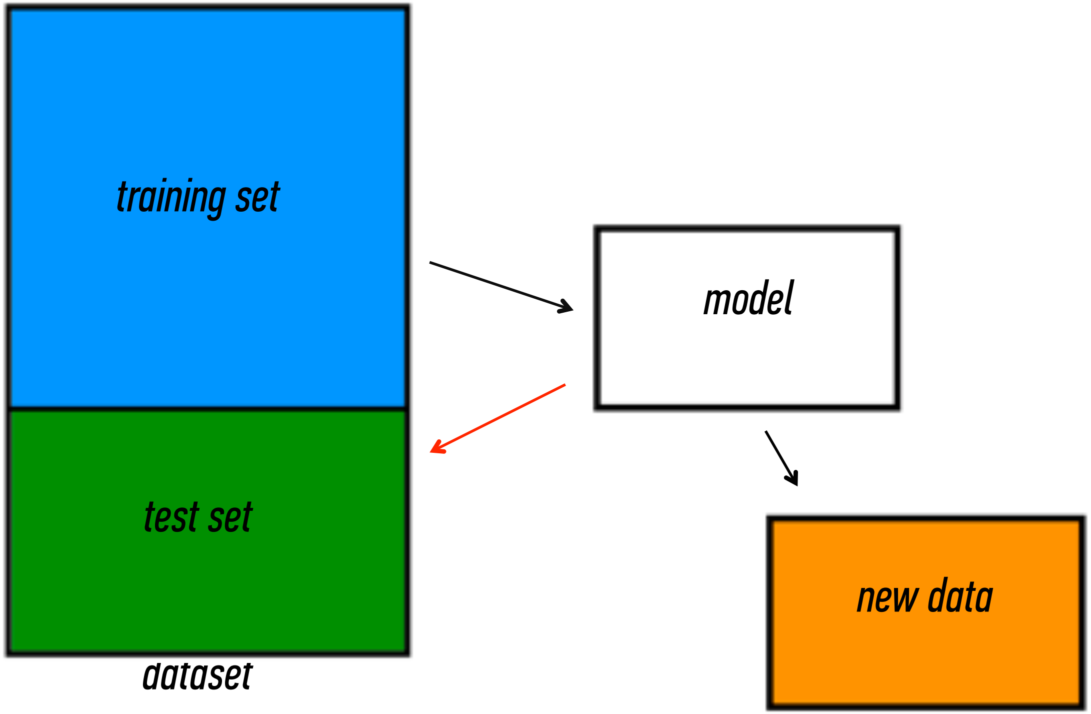
Q: What types of prediction error will we run into?

1) training error



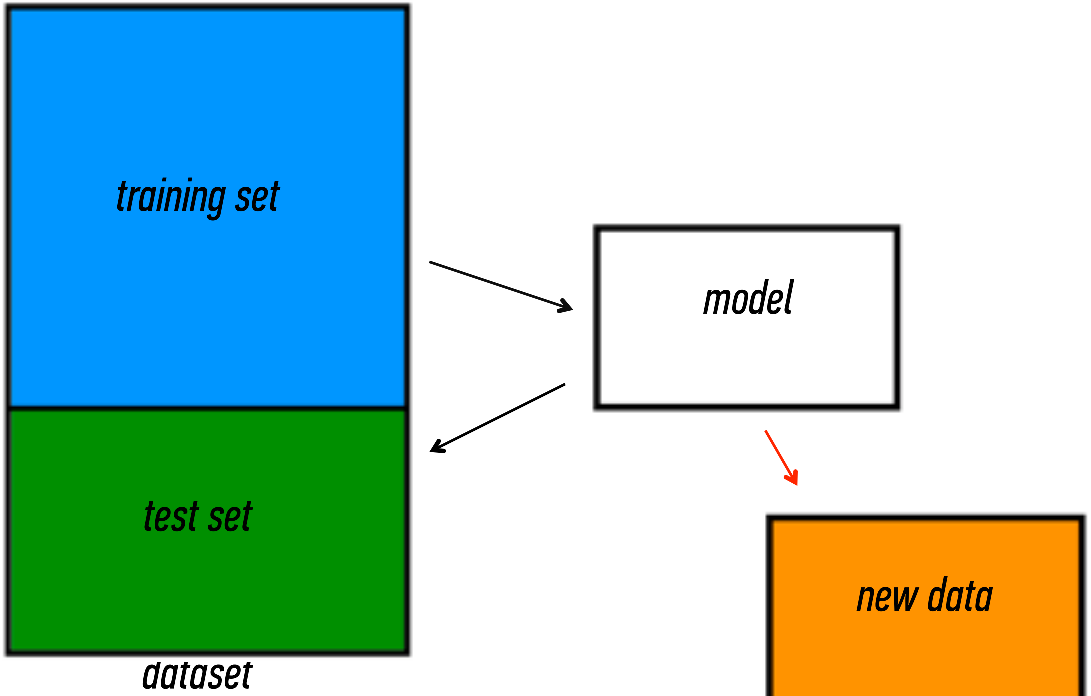
Q: What types of prediction error will we run into?

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



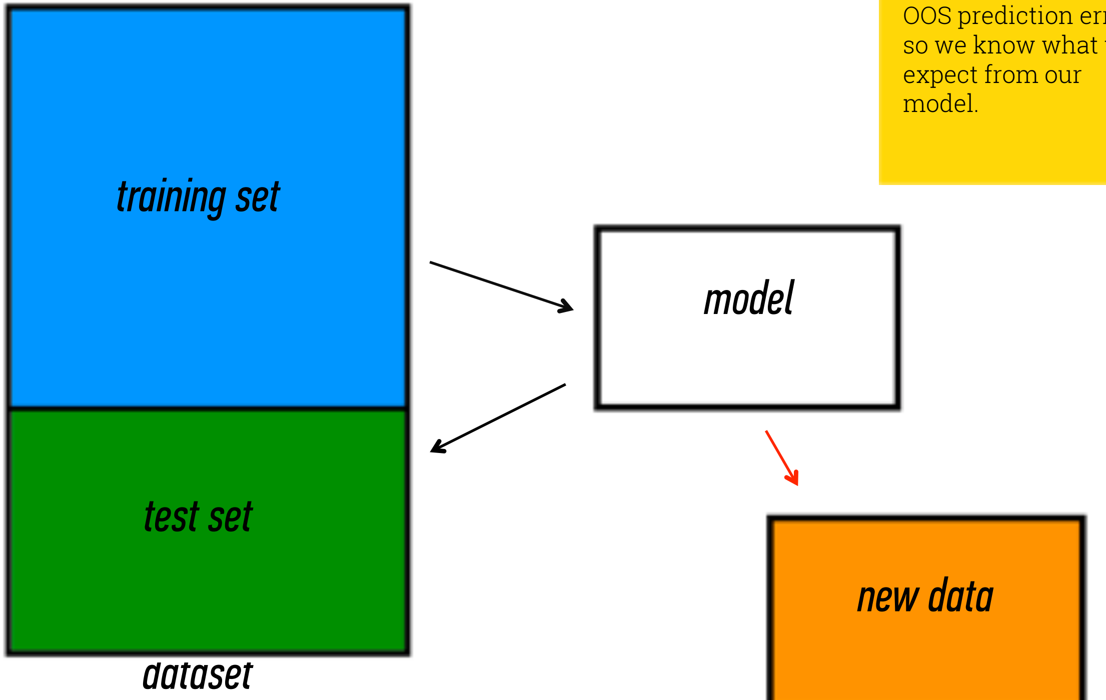
Q: What types of prediction error will we run into?

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.

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?

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.

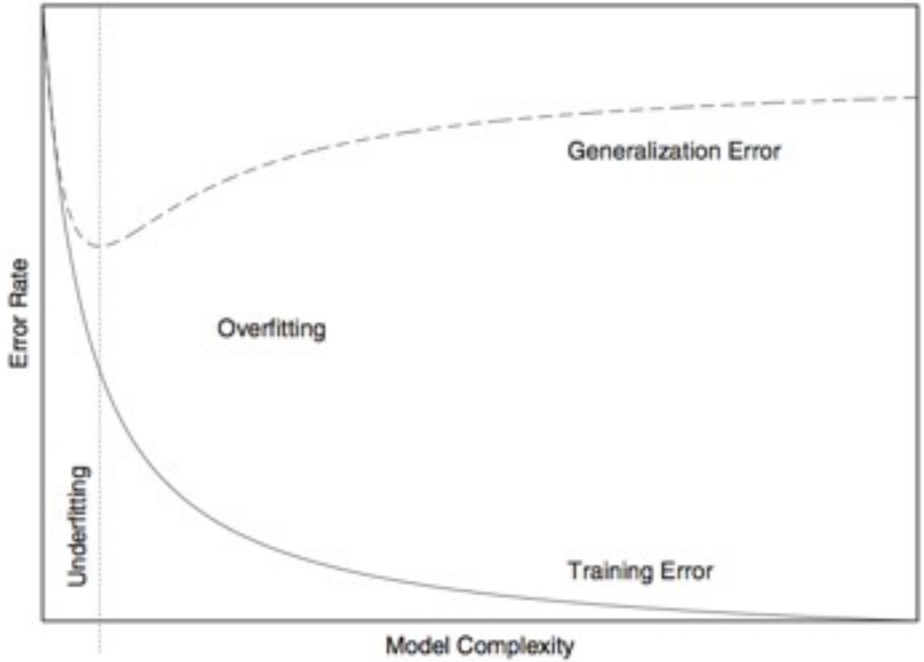
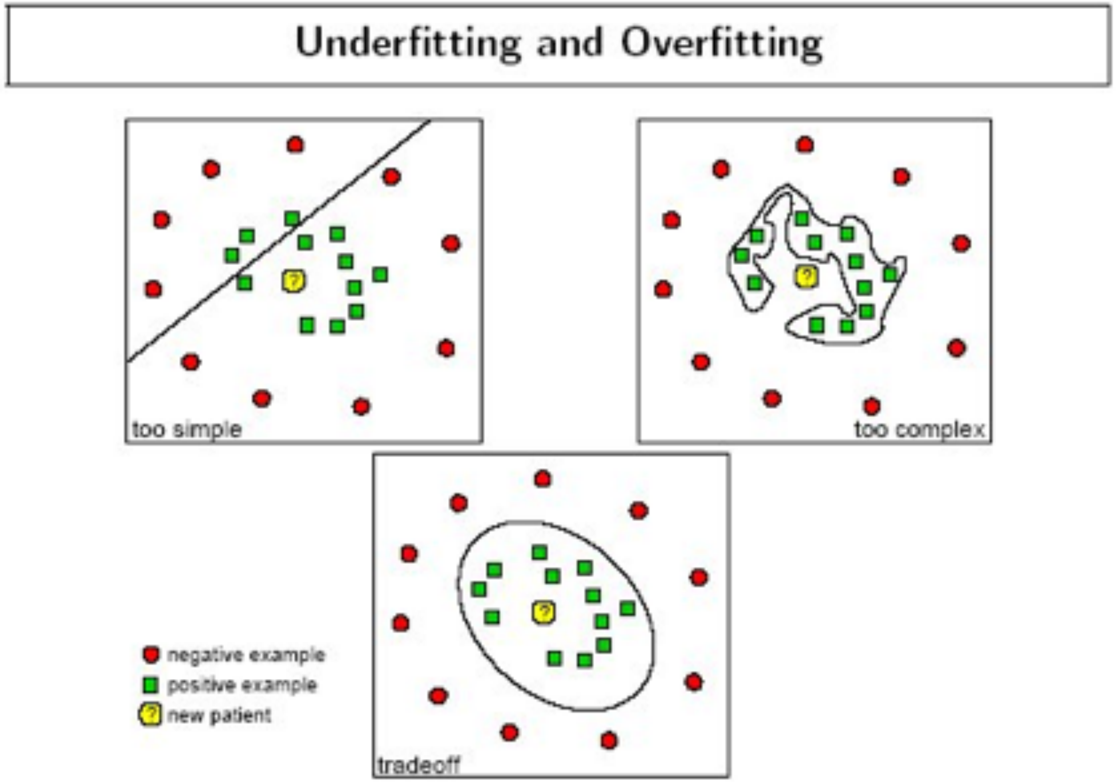


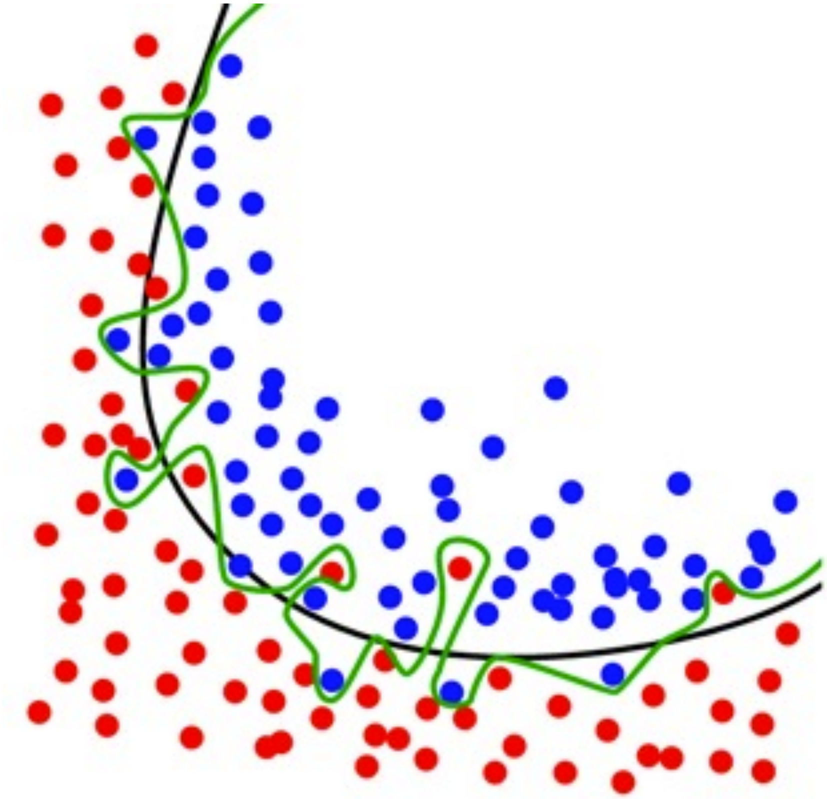
FIGURE 18-1. Overfitting: as a model becomes more complex, it becomes increasingly able to represent the training data. However, such a model is overfitted and will not generalize well to data that was not used during training.

source: Data Analysis with Open Source Tools, by Philipp K. Janert. O'Reilly Media, 2011.

OVERFITTING – EXAMPLE



source: http://www.dtreg.com



source: http://www.dtreg.com

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.

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.

Something is still missing!

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.

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.*

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:

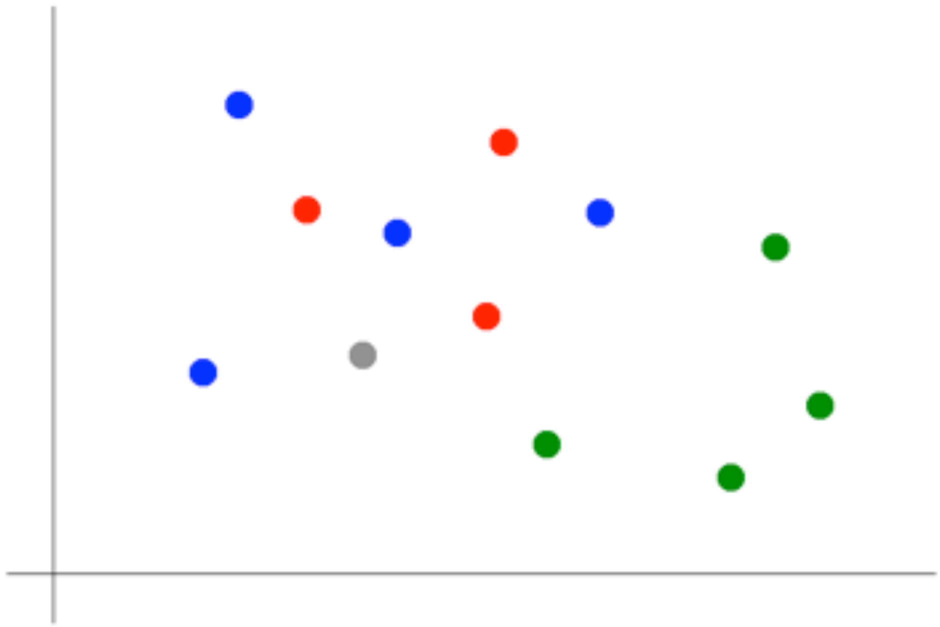
1) *More accurate estimate of OOS prediction error.*
2) *More efficient use of data than single train/test split.*
    - *Each record in our dataset is used for both training and testing.*
3) *Presents tradeoff between efficiency and computational expense.*
    - *10-fold CV is 10x more expensive than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

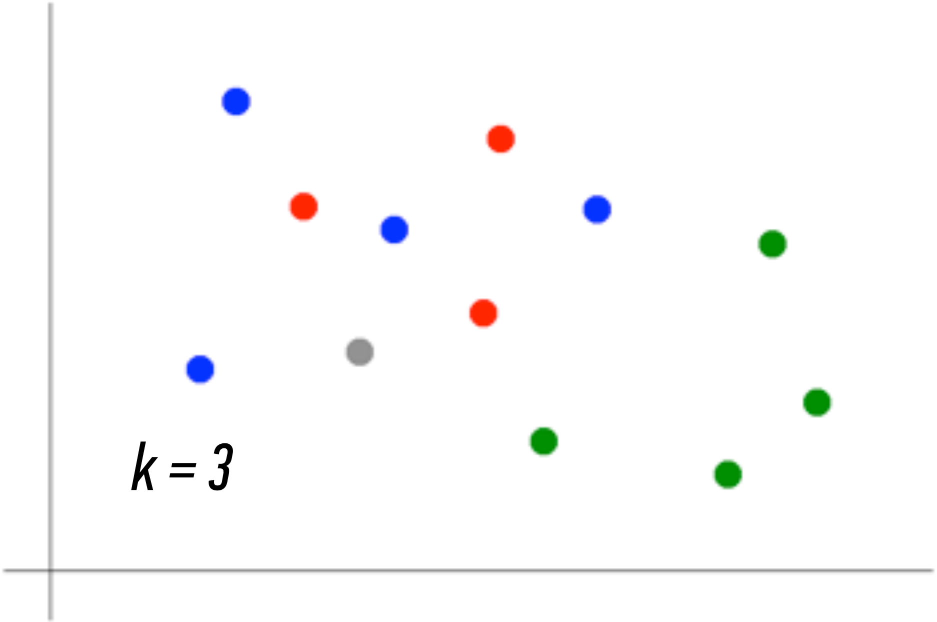
III. KNN CLASSIFICATION

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



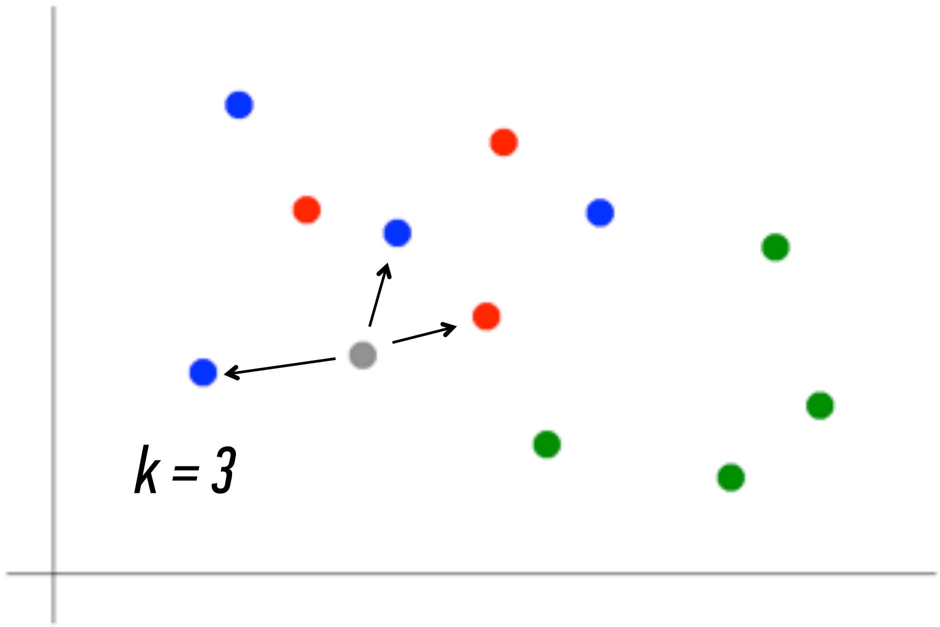
Suppose we want to predict the color of the grey dot.

1) *Pick a value for k.*



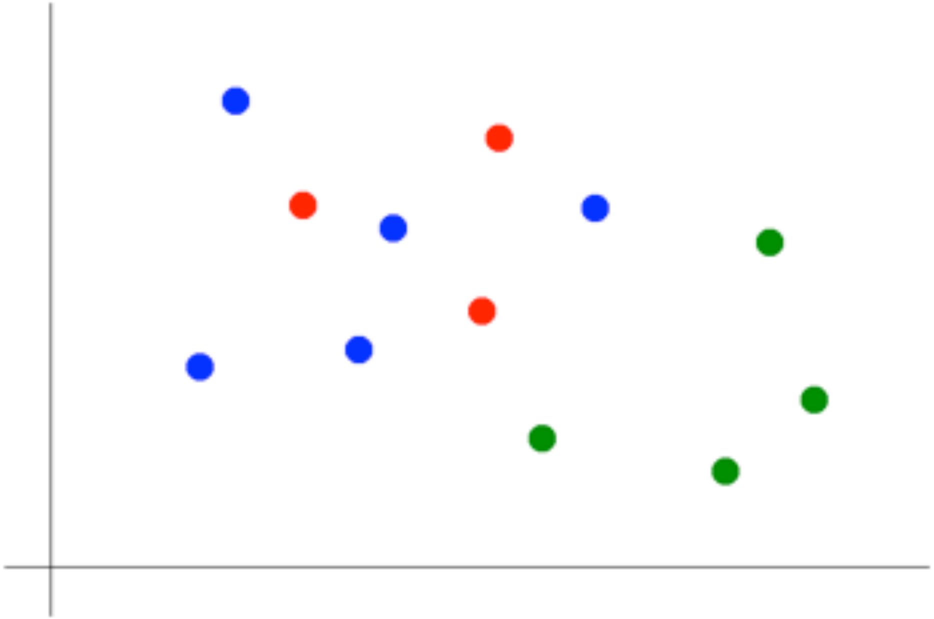
Suppose we want to predict the color of the grey dot.

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



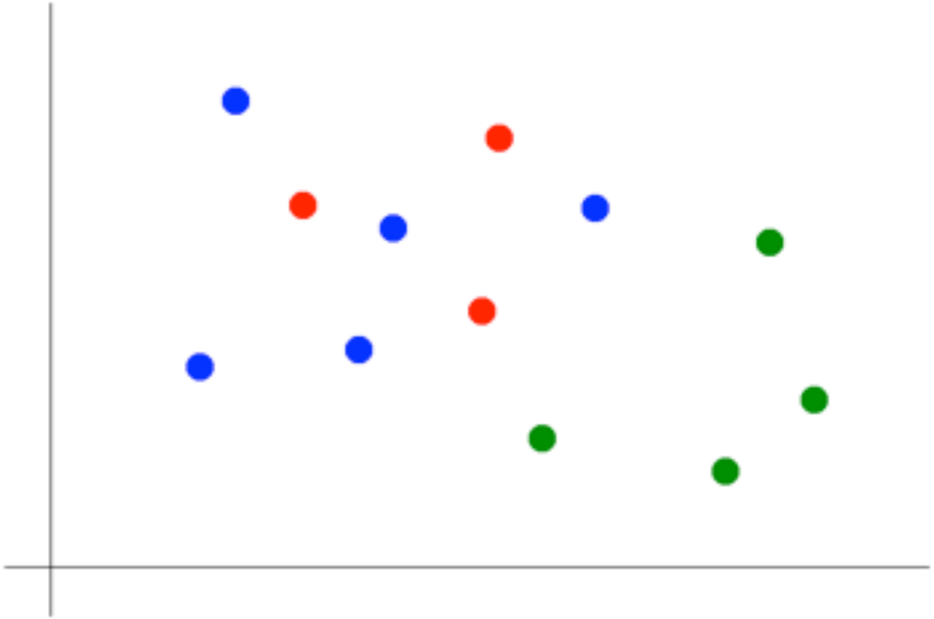
Suppose we want to predict the color of the grey dot.

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.

We measure distance using the ***Euclidean distance function.***

We measure distance using the ***Euclidean distance function.***

$$d(p, q) = \sqrt{}(p_n - q_n)^2$$

We measure distance using the ***Euclidean distance function.***

$$d(p, q) = \sqrt{}(p_n - q_n)^2$$

That is, for each feature of data, we'd measure the distance between two observations.

Consider the iris data set's four features,

Sepal length/width and petal length/width:

Consider the iris data set's four features,
Sepal length/width and petal length/width:

$$d(p, q) = \sqrt{}(p_{s.length} - q_{s.length})^2 + (p_{s.width} - q_{s.width})^2 + (p_{p.length} - q_{p.length})^2 + (p_{p.width} - q_{p.width})^2$$

Consider the iris data set's four features,
Sepal length/width and petal length/width:

$d(p, q) = \sqrt{}(p_{s.length} - q_{s.length})^{2} + (p_{s.width} - q_{s.width})^{2} + (p_{p.length} - q_{p.length})^{2} + (p_{p.width} - q_{p.width})^{2}$

$d(p, q) = \sqrt{}((5.1 - 4.9)^{2} + (3.5 - 3.0)^{2} + (1.4 - 1.4)^{2} + (.2 - .2)^{2})$

$d(p, q) = \sqrt{}(.04 + .25 + 0 + 0) = .53$

There are various ways to measure distance between points , so if distance measurement is interesting to you, learn more about ***taxicab geometry*** *(the* ***L1 norm*** *from regressions last week!) and the* ***Minkowski distance****.*

*There are various ways to measure distance between points , so if distance measurement is interesting to you, learn more about **taxicab geometry** (the **L1 norm** from regressions last week!) and the **Minkowski distance**.*

*For a completely different metric, also learn about how computer scientists use the **Levenshtein distance**!*

LAB: KNN CLASSIFICATION

DISCUSSION:

1. WHAT ARE SOME POTENTIAL SETBACKS OR PITFALLS TO THE KNN ALGORITHM?

2. WHAT ARE SOME POTENTIAL IMPLEMENTATION CHANGES TO THE ALGORITHM THAT COULD BE MADE TO GET AROUND THESE PITFALLS?

NEXT CLASS: PROBABILITY AND NAIVE BAYES CLASSIFICATION