Panel 1

## Evaluating a model

Suppose we have a large dataset.
We've normalized or standardized each attribute, and are ready to use 1-NN.

But before we can deploy the system to make predictions in the real world,
we need to know:
Do we expect the system's predictions to be excellent, or poor, or ?

Panel 2

## Training and Testing Sets

Remember: we want our model to _____ger_____ well to previously unseen problems with unknown solutions.

So we'll divide the dataset into a _____train_____ set (that we'll let the model see) and a _____testing_____ set (to serve as "previously unseen" problems).

We'll develop the model using the training set only.
We'll evaluate the model using the testing set.

Since the testing set came from the full dataset, we know the correct answers,
so we can compare the system's prediction with our correct answers.
That is, for testing set example i, we can compare _____ with _____.

We should choose the training and testing sets carefully, so that both reflect the proportions of different attributes and classifications of the entire dataset.

That is, both the training and testing sets should be _____ of the entire dataset.
(And of course the training and testing sets combined equal the full dataset.)

Panel 3

## Multiple Rounds of Testing

How can we be certain we have a good training and testing set?

It'd be safer to try this multiple times. For example:

randomTesting(`alg`, `dataset`):
  for `i` = 0 to whatever:
    `testing` = some randomly-chosen elements from `dataset`
    `training` = all remaining elements
    `model` = apply `alg` to `training`
    `results` = test `model` on `testing`
  Analyze the results (e.g., take an average, look at max and min, etc.)

Usually, people don't do random testing like this, though.

Let's look at some systematic ways.

Panel 4

## Leave-One-Out Testing

This is the most intensive (effective and expensive) means of testing with multiple training/testing sets.

leaveOneOutTest(`alg`, `dataset`):
  for `i` = 0 to `m-1`:
    `testing` = $\{x^{(i)}\}$
    `training` = `dataset` - $\{x^{(i)}\}$
    `model` = apply `alg` to `training`
    `results` = test `model` on `testing`
  Analyze the results (e.g., take an average, look at max and min, etc.)

For example, suppose I'd like to know how effective 1-NN is on a given dataset (transformed using normalization/standardization in some way).

I could call leaveOneOutTest(1-NN, dataset).

Panel 5

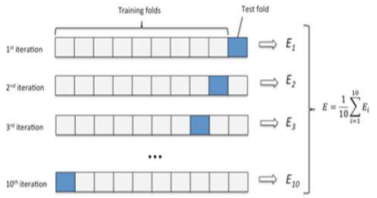
## Alternative: k-fold cross validation

If it's expensive to build a model, then leave-one-out testing is very expensive!
(The main testing loop runs m times, for m examples in the dataset.)

Popular alternative: k-fold crc
(Sometimes also called v-fold cross validation.)

We choose a positive integer k > 1.
We say we have k folds.
That is, each fold is a subset of the entire dataset.
Each is about the same size.

At each iteration of our test,
choose k-1 folds to be in the training set,
remaining fold to be in the testing set.




Panel 6

## k-fold cross validation

```
kFoldCV(alg, dataset, k):
  for each element in dataset:
    randomly assign it to one of the k folds

  for i = 0 to k-1:
    testing = the ith fold
    training = all other folds
    model = apply alg to training
    results = test model on testing
  Analyze the results (e.g., take an average, look at max and min, etc.)
```

So leave-one-out testing is just k-fold cross validation where k = _____.

You're now ready for hw05, hw06, and hw07!

Panel 7

## Data science fundamentals: Big picture so far

What we've discussed and what we haven't

- An algorithm can develop a predictive model on a dataset. We hope the model generalizes well.
  - We've focused on 1-NN so far.
  - There are many other algorithms.
- Before giving a dataset to a model, the dataset should be "cleaned" or preprocessed.
  - One tool is standardization or normalization.
  - There are often other useful data cleaning actions to take too. We practiced some such operations in hw03, section D.
- This (the algorithm, the dataset, and the data cleaning strategy) should be evaluated.
  - k-fold cross validation is the most common approach, by far.
- If an algorithm evaluates poorly, we need to figure out what to do:
  - Improve the dataset somehow
  - Change our data cleaning strategy
  - Change some parameters of our algorithm
  - Change our algorithm entirely

Panel 8

## 1-NN and k-NN

Recall 1-NN: For some new problem p, find $r = \arg\min_i d(\vec{x}^{(i)}, \vec{p})$ and return $y^{(r)}$.

So 1-NN looks at the single nearest neighbor, and returns the corresponding classification.

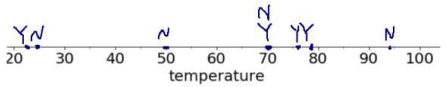
k-NN, in contrast, looks at the k nearest neighbors, for whatever positive k you want to use.
The k classifications of the k nearest neighbors would need to be aggregated somehow (e.g., majority vote).

Note that the "k" in k-NN is not necessarily the same as the "k" in k-fold cross validation.
That's why sometimes it's called v-fold cross validation: because everyone likes to use k for all kinds of stuff.

Panel 9

## Example: 3-NN in 1 dimension

| Temp (F) | Nice Outside? |
|---|---|
| 70 | Y |
| 50 | N |
| 25 | N |
| 78 | Y |
| 94 | N |
| 76 | Y |
| 23 | Y |
| 70 | N |




By 3-NN with a majority vote, what is the classification of a 68-degree day?

Reminders:
- k-NN, for any integer k > 0, can be done on data in *any* number of dimensions
- The "k" in k-NN need not (and usually won't) match the "k" in k-fold cross validation.

You're now ready for hwo8!

Panel 10

## Classification versus Regression

Our k-NN examples have all been _____ problems.
That is, the dependent variable is a _____ type
(e.g. Y/N, boolean, low-medium-high, integers 1-10, etc.)

Algorithms that work on classification problems are often called
_____, which build _____.

But we could just as easily imagine a problem where the dependent variable is a
_____ type. This is a _____ problem, for which we can
apply _____, which build _____.

Note:
- _____ is a common regression algorithm.
- What's the classification version? You might guess "linear classification", but no, it's _____.
- Therefore, sometimes the phrase "regression algorithm" refers to these two algorithms (one continuous, one discrete), but in other contexts it means any algorithm on a continuous dependent variable.

Panel 11

## k-NN Regression

Suppose we want to predict housing prices.

We have a dataset of homes with various attributes (square feet, # bedrooms, etc.)

We can build a k-NN regressor:
- Given a new target home, find the nearest homes in the dataset.
- Aggregate the nearest homes' prices somehow
  - Often an _____average_____,
    or maybe an average weighted by distance from the target home.

So in the standard k-NN algorithms, there's not much that's different between building a k-NN classifier and a k-NN regressor.

It's just a matter of how to aggregate the neighbors' predictions.

Panel 12

## Data science fundamentals: Big picture so far

What we've discussed and what we haven't

- An algorithm can develop a predictive model on a dataset. We hope the model generalizes well.
  - k-NN classification and regression
  - There are many other algorithms.
- Before giving a dataset to a model, the dataset should be "cleaned" or preprocessed.
  - One tool is standardization or normalization.
  - There are often other useful data cleaning actions to take too. **<---- More, up next!**
- This (the algorithm, the dataset, and the data cleaning strategy) should be evaluated.
  - k-fold cross validation is the most common approach, by far.
- If an algorithm evaluates poorly, we need to figure out what to do:
  - Improve the dataset somehow
  - Change our data cleaning strategy
  - Change some parameters of our algorithm
  - Change our algorithm entirely

Panel 13

**Just a little bit of measurement theory**

measurement theory is the study of how measurements can be interpreted. One key insight in measurement theory came from psychologist Stanley Smith Stevens in 1946. He classified data into four types:

1. Nominal
2. Ordinal
3. Interval
4.

The type of data determines what statistical techniques can be meaningfully applied (but we won't get into that much in this course).

Some dispute the usefulness of these classifications, but I see them used very frequently in data science discussions, so we'll consider them here.

Panel 14

**Nominal Values**

The word "nominal" comes from latin "nom", meaning "name".
An attribute of a nominal type just has "names" for values.

Examples:
- Steve, Jimmy, Samantha, Alice
- brick, stone, plastic, wood
- red, yellow, purple, blue

There's no ordering, and you can't do mathematical operations like +, -, *, /.

Panel 15

**Convert Nominal Values --> Numeric**

Example: Roof Style
(flat, gable, gambrel, hip, mansard, shed)

-->
(5, 4, 3, 2, 1, 0)

or also normalized to [0,1], for example:
(1, 0.8, 0.6, 0.4, 0.2 0.0)

Is this ok?

Panel 16

**Convert Nominal Values --> Numeric**

Example: Roof Style (flat, gable, gambrel, hip, mansard, shed)

one-hot encoding (aka dummy variables, indicator variables):
Replace the Roof Style attribute with 6 new attributes (one per nominal value).
Only one attribute will be nonzero for any given example.

| RoofStyle | RS_flat | RS_gable | RS_gambrel | RS_hip | RS_mansard | RS_shed |
|---|---|---|---|---|---|---|
| flat | 1 | 0 | 0 | 0 | 0 | 0 |
| gable | 0 | 1 | 0 | 0 | 0 | 0 |
| gambrel | 0 | 0 | 1 | 0 | 0 | 0 |
| hip | 0 | 0 | 0 | 1 | 0 | 0 |
| mansard | 0 | 0 | 0 | 0 | 1 | 0 |
| shed | 0 | 0 | 0 | 0 | 0 | 1 |

So, house X with sq. ft. of 2000 and a mansard roof would change from:

| | sq. ft. | RoofStyle |
|---|---|---|
| X | 2000 | mansard |

to:

| | sq. ft. | RS_flat | RS_gable | RS_gambrel | RS_hip | RS_mansard | RS_shed |
|---|---|---|---|---|---|---|---|
| X | 2000 | 0 | 0 | 0 | 0 | 1 | 0 |

Panel 17

**Convert Nominal Values --> Numeric**

One-hot encoding
Advantages:

It's numeric!
No implied ordering or values (i.e., (

Disadvantages:

(In terms of modeling, no "grey are

If you want to do this in Python, I recommend you look into the Pandas

_____ function.

Panel 18

**Convert Nominal Values --> Numeric**

Alternative to one-hot encoding: bina

| RoofStyle | RS_2 | RS_1 | RS_0 |
|---|---|---|---|
| flat | 0 | 0 | 0 |
| gable | 0 | 0 | 1 |
| gambrel | 0 | 1 | 0 |
| hip | 0 | 1 | 1 |
| mansard | 1 | 0 | 0 |
| shed | 1 | 0 | 1 |

No or
It's nu
Not a

| | sq. ft. | RoofStyle |
|---|---|---|
| X | 2000 | mansard |

becomes:

| | sq. ft. | RS_2 | RS_1 | RS_0 |
|---|---|---|---|---|
| X | 2000 | 1 | 0 | 0 |

Panel 19

**Ordinal Values**

An attribute of an ordinal type has names, just like nominal types, but the names are Ordered.

Examples:

- small, medium, large
- poor, fair, average, good, excellent
- red, orange, yellow, green, blue, indigo, violet

Panel 20

**Convert Ordinal Values --> Numeric**

Example: Exterior Quality
(excellent, good, average, fair, poor)
-->
(4, 3, 2, 1, 0)

or also normalized to [0,1], for example:
(1, 0.75, 0.5, 0.25, 0)

Is this ok?

We could say "that's fine" and use this approach anyway.
Or we could use one-hot encoding or binary encoding (but cause increased dimensionality).

Panel 21

**Nominal and Ordinal Attributes**

Nominal and ordinal attributes are collectively called _____ or ___qualitative___ attributes.

Panel 22

**Interval and Ratio Values**

An attribute of an interval type is continuous, and differences have meaning.

An attribute of a ratio type is like an interval type, but it also has a meaningful 0, so ratios make sense.

Interval and ratio attributes are collectively called _____ or ___quantitative___ attributes.

For our purposes, we're going to think of interval and ratio values as simply "numerical", and apply standardization or normalization to them.

There's much more to the story, though, if you want to study statistics in depth (a good idea!).

Panel 23

**Options in Data Transformation**

- Categorical (nominal and ordinal):
  - Assign integers (beware assumed ordering and differences) and then normalize or standardize
  - One-hot encoding (beware high data dimensionality)
  - Boolean encoding (beware spurious attribute connections)
- Numerical (interval and ratio):
  - Normalize to [0,1] $x_{new} = \frac{x - x_{min}}{x_{max} - x_{min}}$
  - Normalize to [-1, 1] $x_{new} = \frac{x - x_{avg}}{x_{max} - x_{min}}$
  - Standardize $x_{new} = \frac{x - \mu_x}{\sigma_x}$ $\sigma_x = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(x^{(i)} - \mu_x)^2}$

Panel 24

**Data science fundamentals: Big picture so far**

What we've discussed and what we haven't

- An algorithm can develop a predictive model on a dataset. We hope the model generalizes well.
  - k-NN classification and regression
  - There are many other algorithms.
- Before giving a dataset to a model, the dataset should be "cleaned" or preprocessed.
  - Normalization, standardization, integer assignment, one-hot encoding, binary encoding
  - Discretization, missing values, inconsistent data, feature engineering, dimensionality reduction **<---- Quick overview, up next!**
- This (the algorithm, the dataset, and the data cleaning strategy) should be evaluated.
  - k-fold cross validation is the most common approach, by far.
- If an algorithm evaluates poorly, we need to figure out what to do:
  - Improve the dataset somehow
  - Change our data cleaning strategy
  - Change some parameters of our algorithm
  - Change our algorithm entirely

Panel 25

**Data Preprocessing: Discretization**

Some algorithms require discrete values.

- ___discretization___ converts values from a continuous type into a discrete type
  - Selecting ranges for each discrete value
    Example: Convert values in [0, 30) to
    [0, 10) --> 1
    [10, 20) --> 2
    [20, 30) --> 3
  - More sophisticated techniques use information theory to determine what size intervals are useful at different parts of the range.
    Example: Body temperature
    Below 94 --> 0 (very low)
    [94, 97) --> 1 (low)
    [97, 99) --> 2 (normal)
    [99, 100) --> 3 (mild fever)
    [100, 102) --> 4 (fever)
    [102, 105) --> 5 (higher fever)
    Above 105 (dangerous fever

Panel 26

**Data Preprocessing: Missing Values**

Some examples might be missing values for some attributes:

- Unknown value
- Attribute not applicable
- Some assumed default
- Data corrupted

Most machine learning algorithms can't handle this. Preprocessing options:

- Get more measurements / find that data
- Drop examples that have any missing attributes
- Drop the attributes with missing values, across the entire dataset
- Replace missing values with a default
- Replace missing values with the aggregate value across the dataset
- Replace a missing value with the aggregate of the k neighbors' values

Panel 27

**Data Preprocessing: Inconsistent data**

- Examples:
  - garage square feet = 0
    but
    number of cars in garage = 2
  - N/A, NA, na, none, (missing), 0, zero
- No easy answers.
  - Determine what's right, and fix the inconsistencies
  - Get more data
  - Drop examples or attributes with inconsistent data

Panel 28

**Data Preprocessing: Feature Engineering**

- Or "attribute engineering": devise _____ based on what you already have
- Examples:
  - Convert "year built" into "years old" or "decades old"
  - Calculate (yard size) = (lot size) - (ground floor size)
  - Use automated techniques like neural networks to learn new attributes that are combinations of existing attributes

Panel 29

## Data Preprocessing: Dimensionality Reduction

- Large numbers of attributes can be problematic for machine learning algorithms.
- Dimensionality reduction: take a large number of attributes, and pare them down:
  - Drop attributes deemed less useful
    - Lower correlation with the dependent variable
    - High correlation with another attribute
    - Translate the attributes into another space via a technique like _____ (PCA) or _____ (SVD)

Panel 30

## Data science fundamentals: Big picture so far

What we've discussed and what we haven't

- An algorithm can develop a predictive model on a dataset. We hope the algorithm generalizes well.
  - k-NN classification and regression
  - There are many other algorithms. <-- Soon: Linear regression and many more core machine learning ideas
- Before giving a dataset to a model, the dataset should be "cleaned" or preprocessed.
  - Normalization, standardization, integer assignment, one-hot encoding, binary encoding, discretization, missing values, inconsistent data, feature engineering, dimensionality reduction
- This (the algorithm, the dataset, and the data cleaning strategy) should be evaluated.
  - k-fold cross validation is the most common approach, by far.
- If an algorithm evaluates poorly, we need to figure out what to do: <---- You'll practice all this in the semester project!
  - Improve the dataset somehow
  - Change our data cleaning strategy
  - Change some parameters of our algorithm
  - Change our algorithm entirely