Data mining & Machine Learning

CS 373
Purdue University

Dan Goldwasser
dgoldwas@purdue.edu

Today's Lecture

*A deeper look into Hypothesis spaces and .. **Your first learning algorithm!***

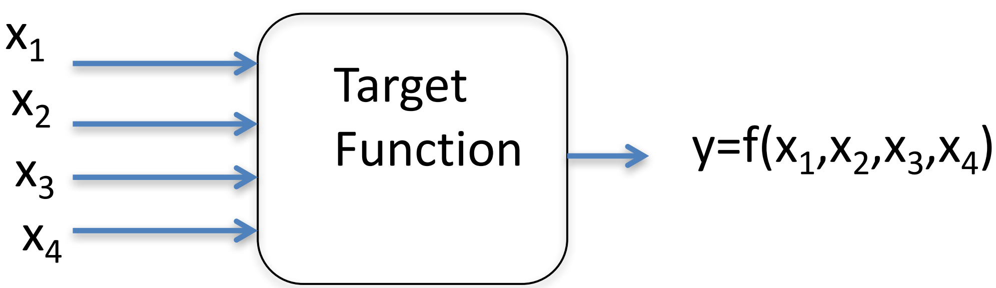
- *Hypothesis space – **over what do we search?***
  - *Underlying question - Can learning really work?*
  - *If the answer is "no", it will be a shorter class…*
- *Your first learning algorithm: KNN.*
  - *Not really learning, but it is, but not really..*
  - *Key idea: Complexity and Expressivity in KNN*

Reminder: Learning Algorithm

- Learning Algorithms generate a **model**, they work under the settings of a specific **protocol**
- **Learning is essentially search —**
  - Given the space of possible models in our hypothesis space
  - Search for the best model
    - Define a procedure for efficiently search the model space.
    - *Best* is defined as maximizing some scoring function, defined w.r.t the training data and other properties

Why is learning possible?

- We want to find the target function based on the training examples



x_1	x_2	x_3	x_4	y
0	0	1	0	0
0	1	0	0	0
0	0	1	1	1
1	0	0	1	1
0	1	1	0	0
1	1	0	0	0
0	1	0	1	0

Why is learning possible?

- How many Boolean functions are there over 4 inputs?
- 2^{16} = 65536 functions (**why?**)
  - 16 possible outputs.
  - Two possibilities for each output
- Without any data, 2^{16} options
- ***Does the data identify the right function?***
- The training data contains 7 examples
  - We still have 2^9 options

x_1	x_2	x_3	x_4	y
0	0	0	0	?
0	0	0	1	?
0	0	1	0	**0** $\leftarrow$
0	0	1	1	**1** $\leftarrow$
0	1	0	0	**0** $\leftarrow$
0	1	0	1	**0** $\leftarrow$
0	1	1	0	**0** $\leftarrow$
0	1	1	1	?
1	0	0	0	?
1	0	0	1	**1** $\leftarrow$
1	0	1	0	?
1	0	1	1	?
1	1	0	0	**0** $\leftarrow$
1	1	0	1	?
1	1	1	0	?
1	1	1	1	?

Is learning even possible?

Hypothesis/Model Space

- **A *hypothesis space* is the set of possible functions we consider**
  - We were looking at the space of **all** Boolean functions
  - *Instead choose a hypothesis space that is smaller than the space of all Boolean functions*
    - Only simple conjunctions (with four variables, there are only 16 conjunctions without negations)
    - Simple disjunctions
    - m-of-n rules: Fix a set of n variables. At least m of them must be true
    - Linear functions

Take 2

- **Simple Conjunctions**: *very small subset of Boolean functions*
  - Only 16 possible conjunction of the form: $y = x_i \wedge \ldots x_j$
    - **Why?**
  - ***Can you find a consistent Hypothesis in this space?***

x_1	x_2	x_3	x_4	y
0	0	1	0	0
0	1	0	0	0
0	0	1	1	1
1	0	0	1	1
0	1	1	0	0
1	1	0	0	0
0	1	0	1	0

Simple Conjunctions

Rule	Counterexample
$y=c$	
x_1	1100 0
x_2	0100 0
x_3	0110 0
x_4	0101 1
$x_1 \wedge x_2$	1100 0
$x_1 \wedge x_3$	0011 1
$x_1 \wedge x_4$	0011 1

x_1	x_2	x_3	x_4	y
0	0	1	0	0
0	1	0	0	0
0	0	1	1	1
1	0	0	1	1
0	1	1	0	0
1	1	0	0	0
0	1	0	1	0

Rule	Counterexample
$x_2 \wedge x_3$	0011 1
$x_2 \wedge x_4$	0011 1
$x_3 \wedge x_4$	1001 1
$x_1 \wedge x_2 \wedge x_3$	0011 1
	1
$x_1 \wedge x_3 \wedge x_4$	0011 1
$x_2 \wedge x_3 \wedge x_4$	0011 1
$x_1 \wedge x_2 \wedge x_3 \wedge x_4$	0011 1

No simple conjunction can explain this data!

New Model space: *M-of-N rules*

- **The class of simple conjunctions is not expressive enough for our functions**
- **How can we pick a better space?**
  - *Prior knowledge about the problem*
  - *Sufficiently "flexible"*
- Let's try another space – ***m-of-n rules***
  - Rules of the form "y = 1 if and only if at least m of the following n variables are 1"
    - How many are there for 4 Boolean variables?
    - **Is there a consistent hypothesis?**

Why is learning possible?

x_1	x_2	x_3	x_4	y
0	0	1	0	0
0	1	0	0	0
0	0	1	1	1
1	0	0	1	1
0	1	1	0	0
1	1	0	0	0
0	1	0	1	0

- m-of-n rules
  - Examples:
    - 1 out of {x1}
    - 2 out of {x1, x3}
    - …
- **Is there a consistent hypothesis?**
  - Check!
  - For example: Let's try checking for "2 out of {x1,x2,x3,x4}"

➔ Exactly one hypothesis is consistent with the data!

Why is learning possible?

- **Learning is removal of remaining uncertainty**
  – If we know that the function is a "m-out-of-n", data can help find a function from that class
- **Finding a good hypothesis class is essential!**
  – You can start small, and enlarge it until you can find a hypothesis that fits the data

Why is learning possible?

- **Learning is removal of remaining uncertainty**
  - If we know that the function is a “m-out-of-n”, data can help find a function from that class
- **Finding a good hypothesis class is essential!**
  - You can start small, and enlarge it until you can find a hypothesis that fits the data

Question: *Can there be more than one function that is consistent with the data*?

How do you choose between them?

And now to something completely different

Your first classifier!

Your First Classifier!

- Let's consider one of the simplest classifiers out there.
- Assume we have a training set $(x_1,y_1)\ldots(x_n,y_n)$
- Now we get a new instance x_{new} ,
- how can we classify it?
  - Example: Can you recommend a movie, based on user's movie reviews?

Your First Classifier!

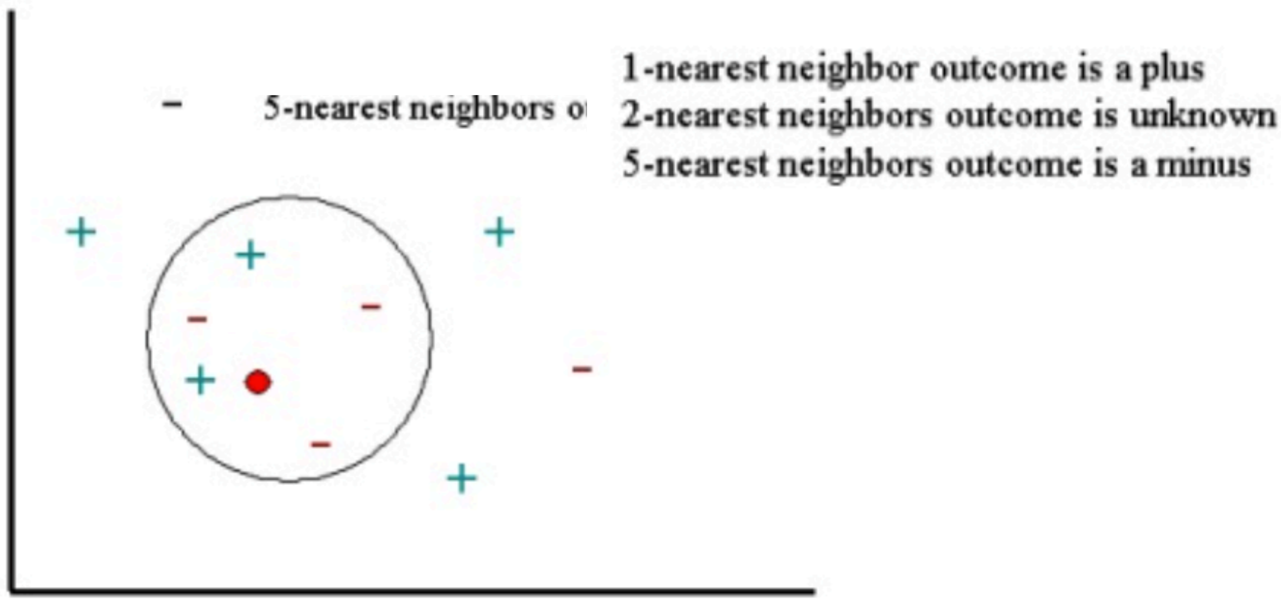
- **Simple Solution**:
  - Find the most similar example (x,y) in the training data and predict the same
    - If you liked "*Fast and Furious*" you'll like "*2 fast 2 furious*"
- Only a single decision is needed: distance metric to compute similarity

$$d(x_1, x_2) = 1 - \frac{x_1 \bigcap x_2}{x_1 \bigcup x_2}$$

$$d(x_1, x_2) = \sqrt[2]{(x_1 - x_2)^2}$$

K Nearest Neighbors

– Can you thing about a better way?
– We can make the decision by looking at several near examples, not just one. **Why would it be better**?



K Nearest Neighbors

- **Learning**: just storing the training examples
- **Prediction**:
  - Find the K training example closest to **x**
- **Predict a label:**
  - Classification: majority vote
  - Regression: mean value
- KNN is a type of *instance based learning*
- This is called *lazy* learning, since most of the computation is done at prediction time

KNN analysis

- ***What are the advantages and disadvantages of KNN?***
  - *What should we care about when answering this question?*
- ***Complexity***
  - ***Space*** *(how memory efficient is the algorithm?)*
    - *Why should we care?*
  - ***Time*** *(computational complexity)*
    - *Both at training time and at test (prediction) time*
- ***Expressivity***
  - *What kind of functions can we learn?*

KNN analysis

- ***What are the advantages and disadvantages of KNN?***
  – *What should we care about when answering this question?*

KNN needs to maintain all training examples!
-Datasets can be HUGE

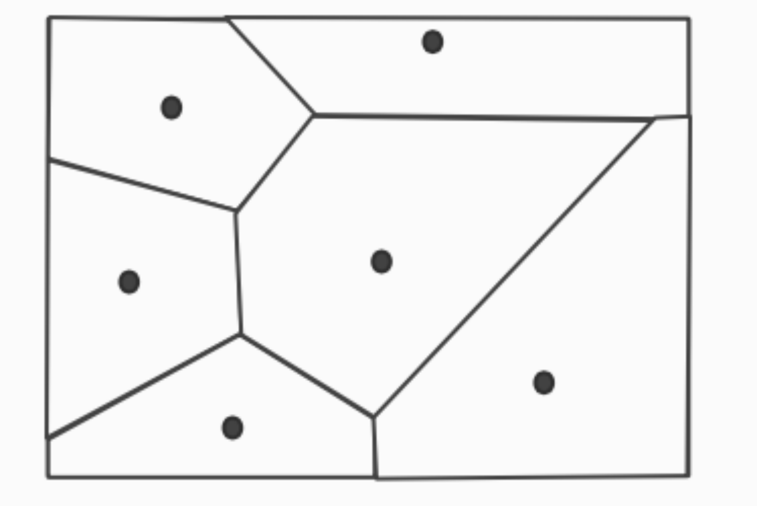
- ***Complexity***
  – ***Space*** *(how memory efficient is the algorithm?)*
    - *Why should we care?*
  – ***Time*** *(computational complexity)*
    - *Both at training time and at test (prediction) time*

Training is very fast! But *prediction is slow*
- O(dN) for N examples with d attributes
- *increases* with the number of examples!

- ***Expressivity***
  – *What kind of functions can we learn?*

KNN analysis

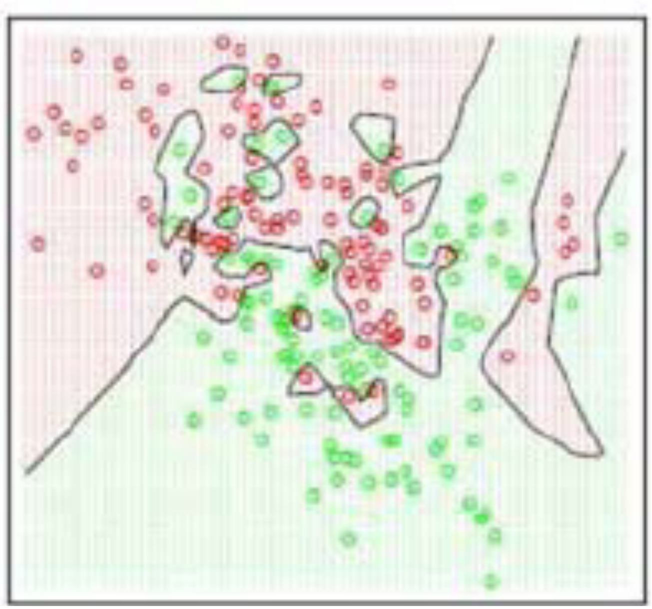
- We discussed the importance of the model space
  - Expressive (we can represent the right model)
  - Constrained (we can search effectively, using available data)
- Let's try to characterize the model space, by looking at the **decision boundary**
- **How would it look if K=1?**

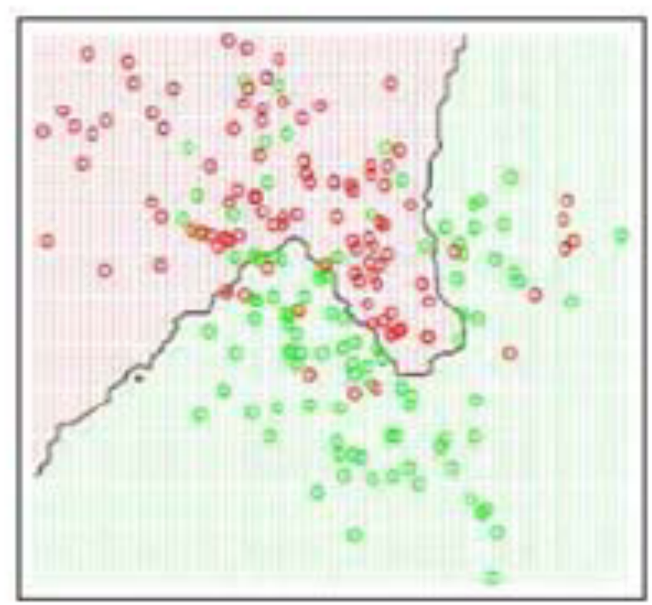


We define the model space to be our choice of K.
Does the complexity of the model space increase of decrease with K?

KNN analysis

- Which model has a higher K value?
- Which model is more sensitive to noise?





Questions

- We know higher K values result in a smoother decision boundary.
  - Less ”jagged” decision regions
  - Total number of regions will be smaller

What will happen if we keep increasing K, up to the point that K=n ?

n = is the number of examples we have

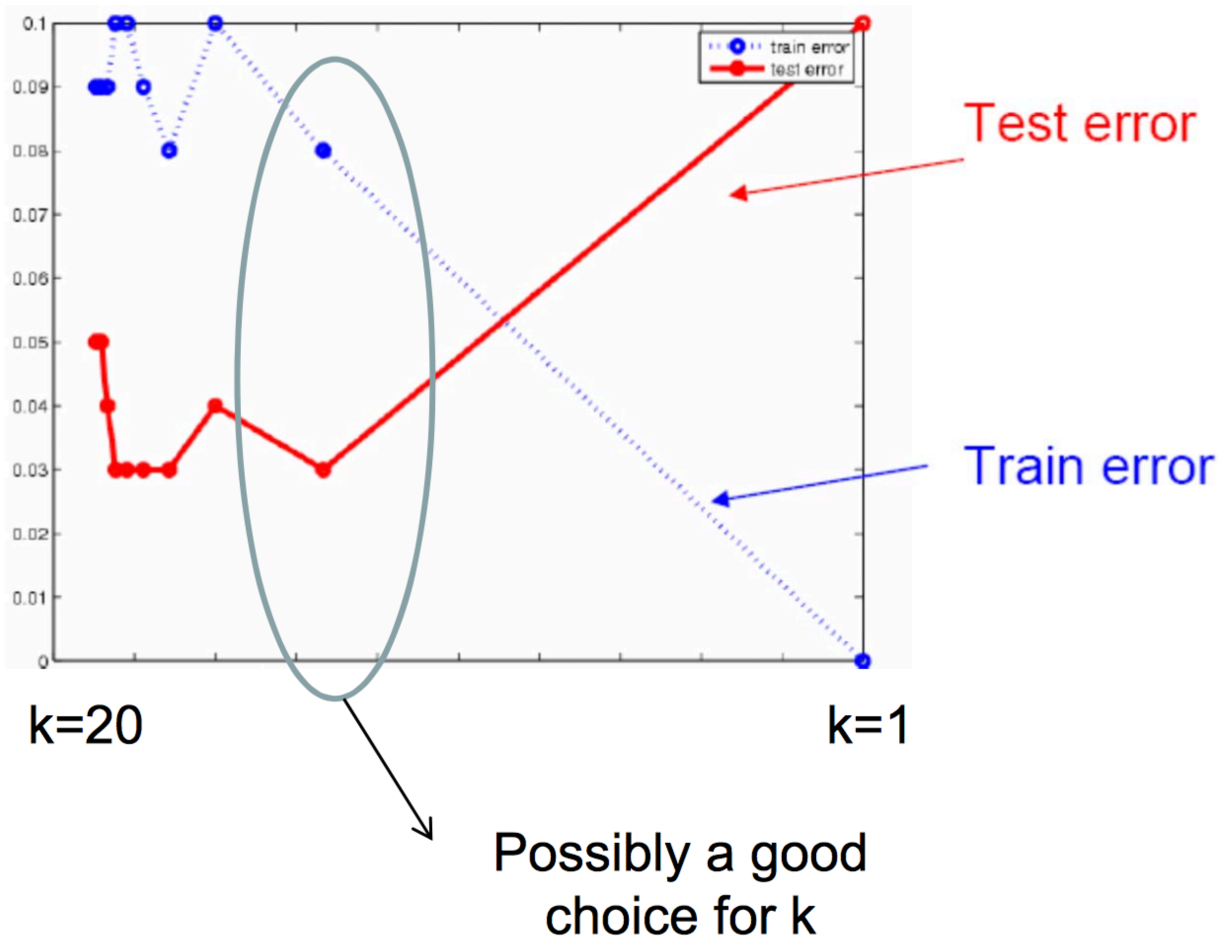
Determining the value of K

- Higher K result in less complex functions (less expressive)
- Lower K values are more complex (more expressive)
  - **How can we find the right balance between the two?**
- **<u>Option 1</u>**: *Find the K that minimizes the training error*.
  - <u>Training error</u>: after learning the classifier, what is the number of errors we get on the training data.
  - What will be this value for k=1, k=n, k=n/2?

Is this a good idea?

- **<u>Option 2</u>**: *Find K that minimizes the **validation error***.
  - <u>Validation error</u>: set aside some of the data (validation set). what is the number of errors we get on the validation data, after training the classifier.

Determining the value of K



In general – using the training error to tune parameters will always result in a more complex hypothesis! **(why?)**

Practical Considerations

- Finding the right representation is key
  - KNN is very sensitive to irrelevant attributes
- Choosing the right distance metric is important
  - Many options!
  - Popular choices:

– Euclidean distance

$$||\mathbf{x}_1 - \mathbf{x}_2||_2 = \sqrt{\sum_{i=1}^{n} (\mathbf{x}_{1,i} - \mathbf{x}_{2,i})^2}$$

– Manhattan distance

$$||\mathbf{x}_1 - \mathbf{x}_2||_1 = \sum_{i=1}^{n} |\mathbf{x}_{1,i} - \mathbf{x}_{2,i}|$$

– L_p-norm

- Euclidean = L_2
- Manhattan = L_1

$$||\mathbf{x}_1 - \mathbf{x}_2||_p = \left(\sum_{i=1}^{n} |\mathbf{x}_{1,i} - \mathbf{x}_{2,i}|^p\right)^{\frac{1}{p}}$$

Summary: Week 1

- Introduction to Machine Learning and Data mining
  - Why is data-centric computing interesting/relevant?
  - Where is it applicable?
  - What is the data-mining process? Where do you start? How do you know you are finished?
- Principles of Machine Learning
  - Model/Hypothesis space, Learning protocol, learning algorithm
  - Explain the tradeoff between complexity and expressiveness
  - KNN learning algorithm

Summary: Week 1

- Is KNN a supervised or unsupervised learning algorithm?
- If we increase K, would we get a more complex decision boundary?
- If we want to learn a Boolean function, what would be a simpler and complex model spaces?
- What can we do if the target function is not in our hypothesis space?
  - Does that even happen? How can we tell? Should we do something about it?
- I'm searching over an infinite size hypothesis space.
  - Would the search converge? Am I guaranteed that the target function is there?