CS 461: Machine Learning
Lecture 1

Dr. Kiri Wagstaff
kiri.wagstaff@calstatela.edu

Introduction

- Artificial Intelligence
  - Computers demonstrate human-level cognition
  - Play chess, drive cars, fly planes
- Machine Learning
  - Computers learn from their past experience
  - Adapt to new environments or tasks
  - Recognize faces, recognize speech, filter spam

How Do We Learn?

How Do We Learn?

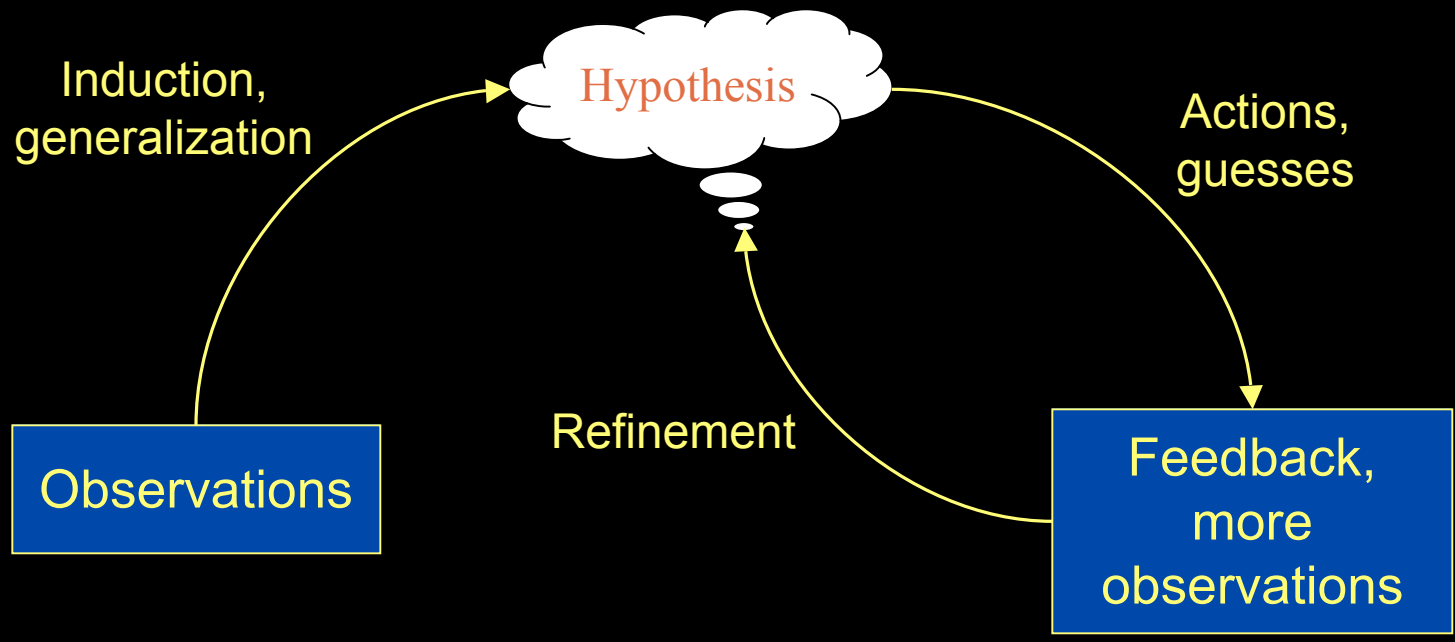
Human	Machine
Memorize	k-Nearest Neighbors, Case-based learning
Observe someone else, then repeat	Supervised Learning, Learning by Demonstration
Keep trying until it works (riding a bike)	Reinforcement Learning
20 Questions	Decision Tree
Pattern matching (faces, voices, languages)	Pattern Recognition
Guess that current trend will continue (stock market, real estate prices)	Regression

Inductive Learning from Grazeeb

(Example from Josh Tenenbaum, MIT)

"tufa"

General Inductive Learning



Machine Learning

- Optimize a criterion (reach a goal) using example data or past experience
- Infer or generalize to new situations
  - Statistics: inference from a (small) sample
  - Probability: distributions and models
  - Computer Science:
    - Algorithms: solve the optimization problem efficiently
    - Data structures: represent the learned model

Why use Machine Learning?

- We cannot write the program ourselves
- We don't have the expertise (circuit design)
- We cannot explain how (speech recognition)
- Problem changes over time (packet routing)
- Need customized solutions (spam filtering)

Machine Learning in Action

- Face, speech, handwriting recognition
  - Pattern recognition
- Spam filtering, terrain navigability (rovers)
  - Classification
- Credit risk assessment, weather forecasting, stock market prediction
  - Regression
- Future: Self-driving cars? Translating phones?

Your First Assignment (part 1)

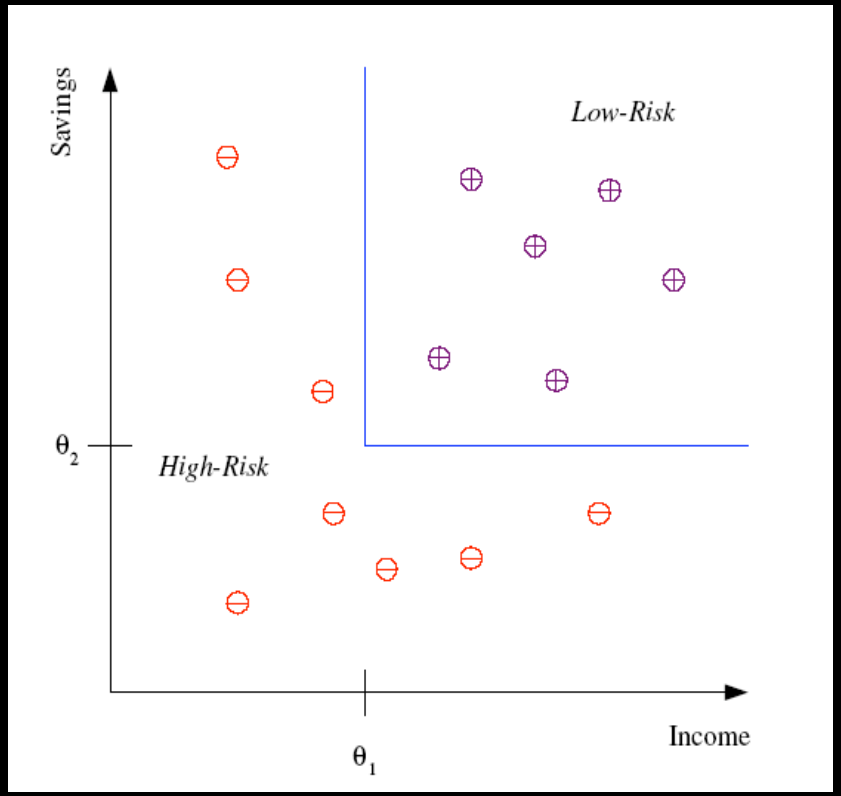
- Find:
  - news article,
  - press release, or
  - product advertisement
  - … about machine learning
- Write 1 paragraph each:
  - Summary of the machine learning component
  - Your opinion, thoughts, assessment
- Due January 10, midnight
  - (submit through CSNS)

Association Rules

- Market basket analysis
  - Basket 1: { apples, banana, chocolate }
  - Basket 2: { chips, steak, BBQ sauce }
- P(Y|X): probability of buying Y given that X was bought
  - Example: P(chips | beer) = 0.7
  - High probability: association rule

Classification

- Credit scoring
- Goal: label each person as "high risk" or "low risk"
- Input features: Income and Savings
- Learned discriminant:
  - If Income > θ_1 AND Savings > θ_2 THEN low-risk ELSE high-risk



Classification: Emotion Recognition

[See movie on website]

Classification Methods in this course

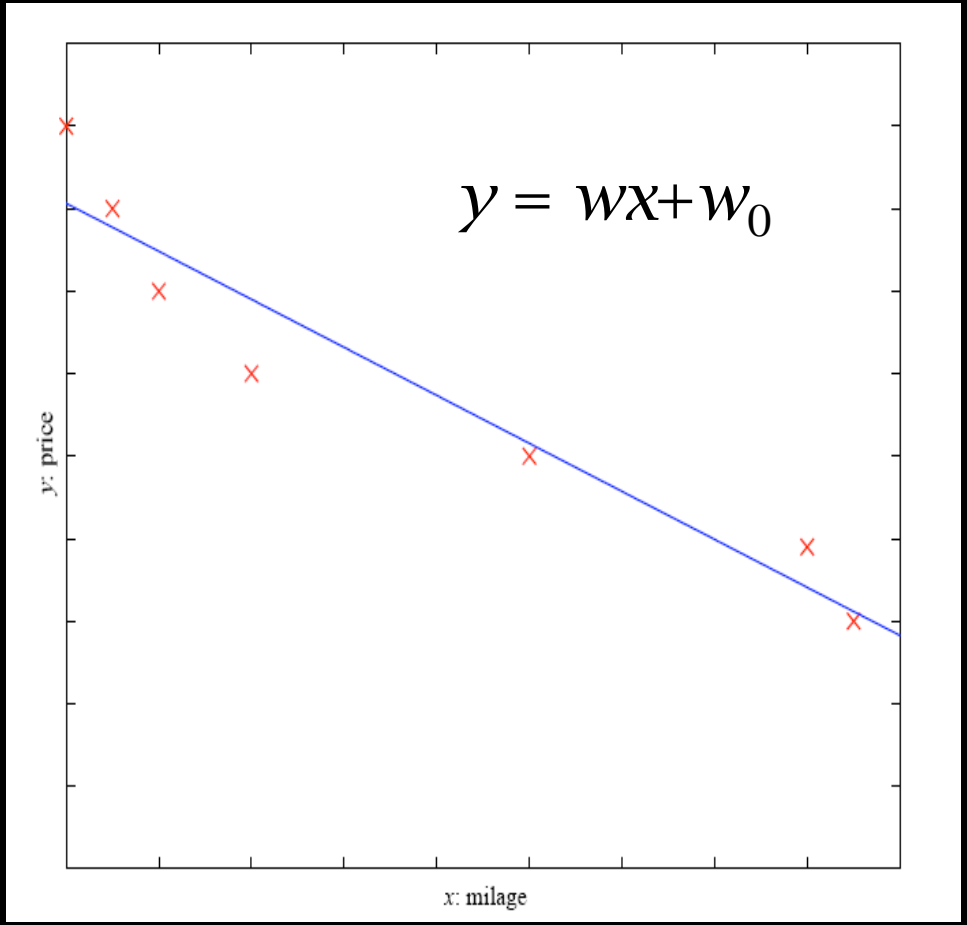
- k-Nearest Neighbor
- Decision Trees
- Support Vector Machines
- Neural Networks
- Naïve Bayes

Regression

- Predict price of used car (y)
- Input feature: mileage (x)
- Learned:

$y = g(x \mid \theta)$

$g(\ )$ model,
θ parameters



Regression: Angle of steering wheel
(2007 DARPA Grand Challenge, MIT)

[See movie on website]

Regression Methods in this course

- k-Nearest Neighbors
- Support Vector Machines
- Neural Networks
- Bayes Estimator

Unsupervised Learning

- No labels or feedback
- Learn trends, patterns

- Applications
  - Customer segmentation: e.g., targeted mailings
  - Image compression
  - Image segmentation: find objects

- This course
  - k-means and EM clustering
  - Hierarchical clustering

Reinforcement Learning

- Learn a policy: sequence of actions
- Delayed reward

- Applications
  - Game playing
  - Balancing a pole
  - Solving a maze

- This course
  - Temporal difference learning

What you should know

- What is inductive learning?
- Why/when do we use machine learning?
- Some learning paradigms
  - Association rules
  - Classification
  - Regression
  - Clustering
  - Reinforcement Learning

Supervised Learning

Chapter 2

Slides adapted from Alpaydin and Dietterich

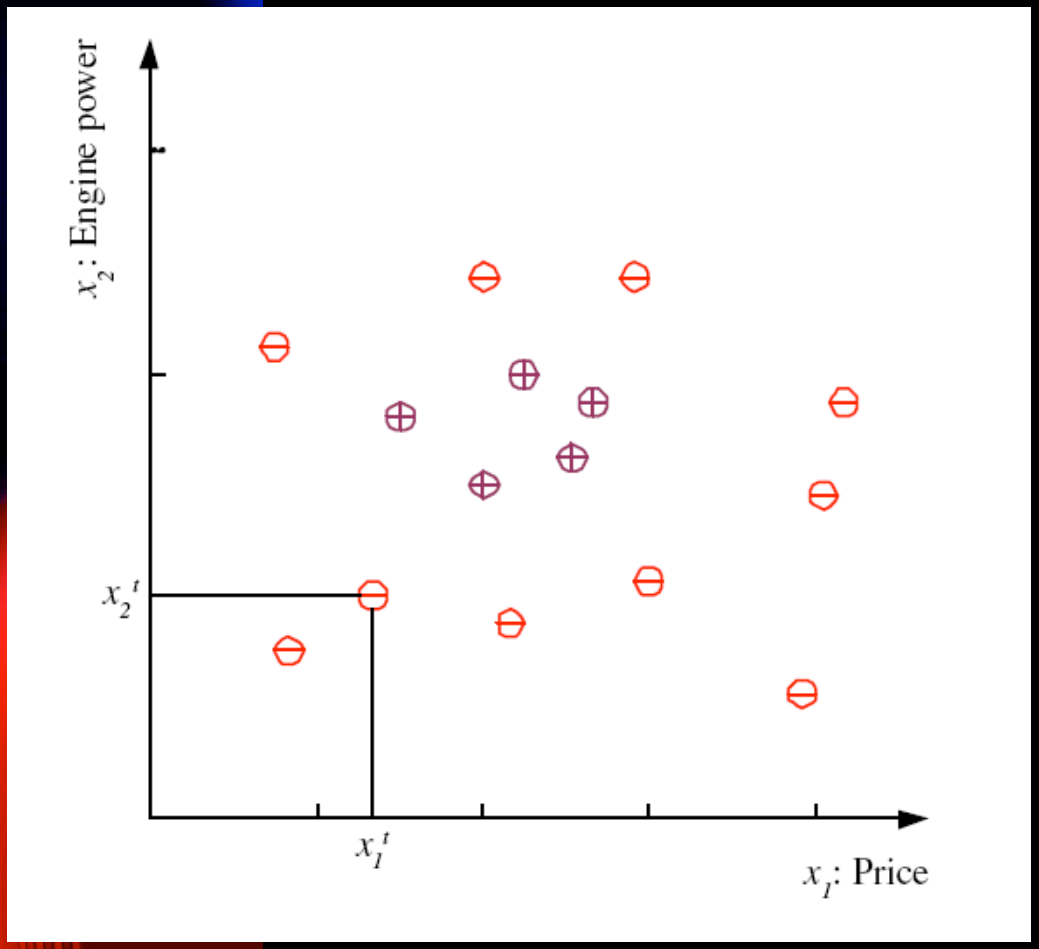
Supervised Learning

- Goal: given <input *x*, output *g(x)*> pairs, learn a good approximation to *g*
  - Minimize number of errors on new *x*'s
- Input: N labeled examples
- Representation: descriptive features
  - These define the "feature space"
- Learning a concept C from examples
  - Family car (vs. sports cars, etc.)
  - "A" student (vs. all other students)
  - Blockbuster movie (vs. all other movies)
- (Also: classification, regression…)

Supervised Learning: Examples

- Handwriting Recognition
  - Input: data from pen motion
  - Output: letter of the alphabet
- Disease Diagnosis
  - Input: patient data (symptoms, lab test results)
  - Output: disease (or recommended therapy)
- Face Recognition
  - Input: bitmap picture of person's face
  - Output: person's name
- Spam Filtering
  - Input: email message
  - Output: "spam" or "not spam"

Car Feature Space and Data Set



Data Set

$$\mathcal{X} = \{\mathbf{x}^t, y^t\}_{t=1}^{N}$$

Data Item

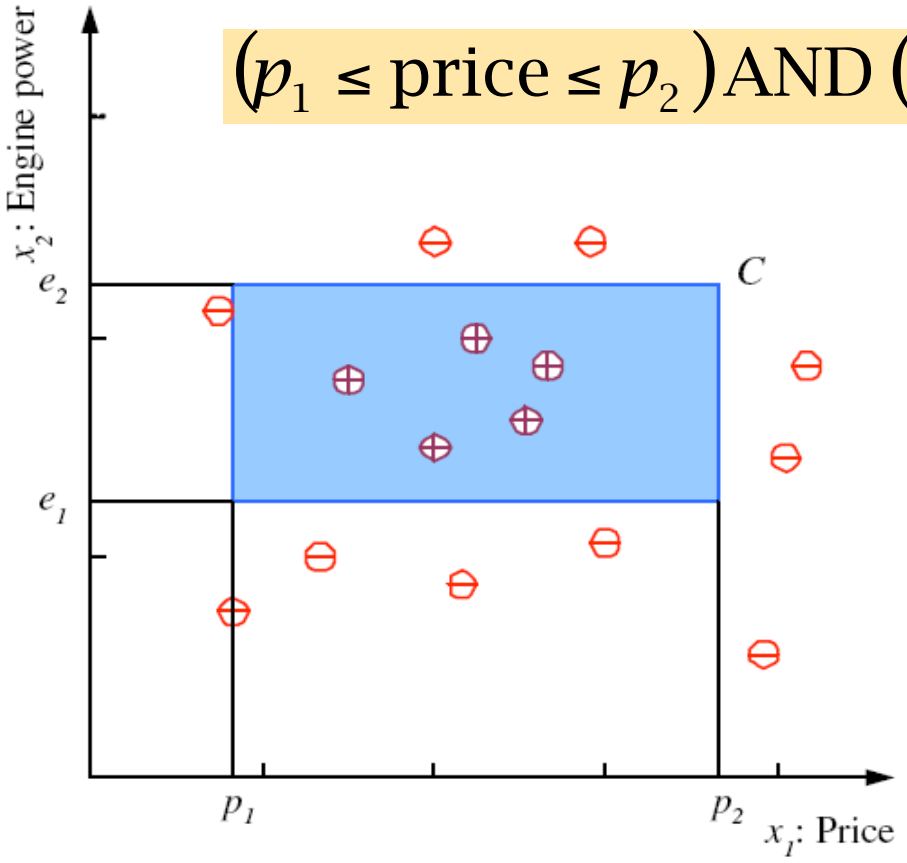
$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

Data Label

$$y = \begin{cases} 1 \text{ if } \mathbf{x} \text{ is positive} \\ 0 \text{ if } \mathbf{x} \text{ is negative} \end{cases}$$

Family Car Concept *C*

$$(p_1 \leq \text{price} \leq p_2) \text{ AND } (e_1 \leq \text{engine power} \leq e_2)$$

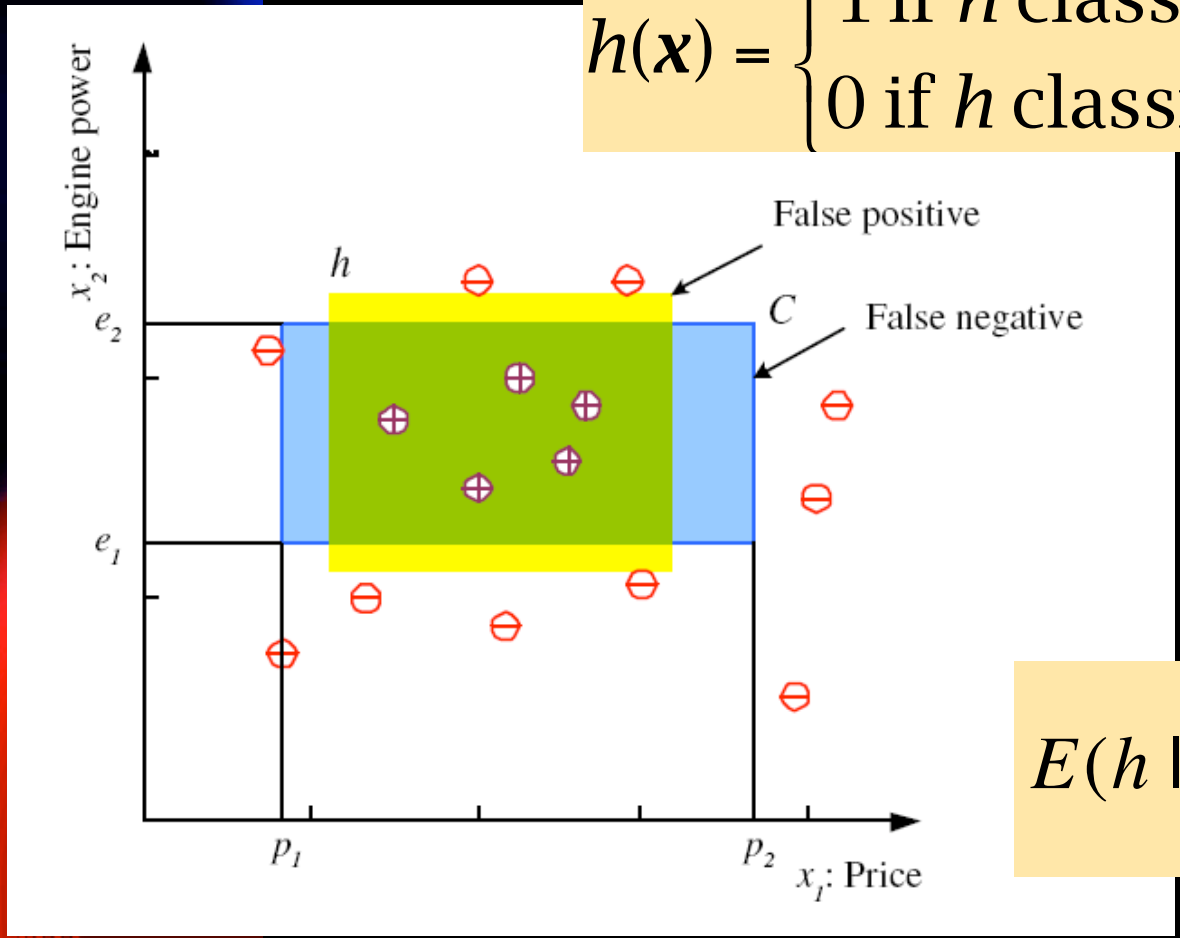


Hypothesis Space $\mathcal{H}$

- Includes all possible concepts of a certain form
  - All rectangles in the feature space
  - All polygons
  - All circles
  - All ellipses
  - …
- Parameters define a specific hypothesis from $\mathcal{H}$
  - Rectangle: 2 params per feature (min and max)
  - Polygon: *f* params per vertex (at least 3 vertices)
  - (Hyper-)Circle: *f* params (center) plus 1 (radius)
  - (Hyper-)Ellipse: *f* params (center) plus *f* (axes)

Hypothesis h

$$h(\boldsymbol{x}) = \begin{cases} 1 \text{ if } h \text{ classifies } \boldsymbol{x} \text{ as positive} \\ 0 \text{ if } h \text{ classifies } \boldsymbol{x} \text{ as negative} \end{cases}$$

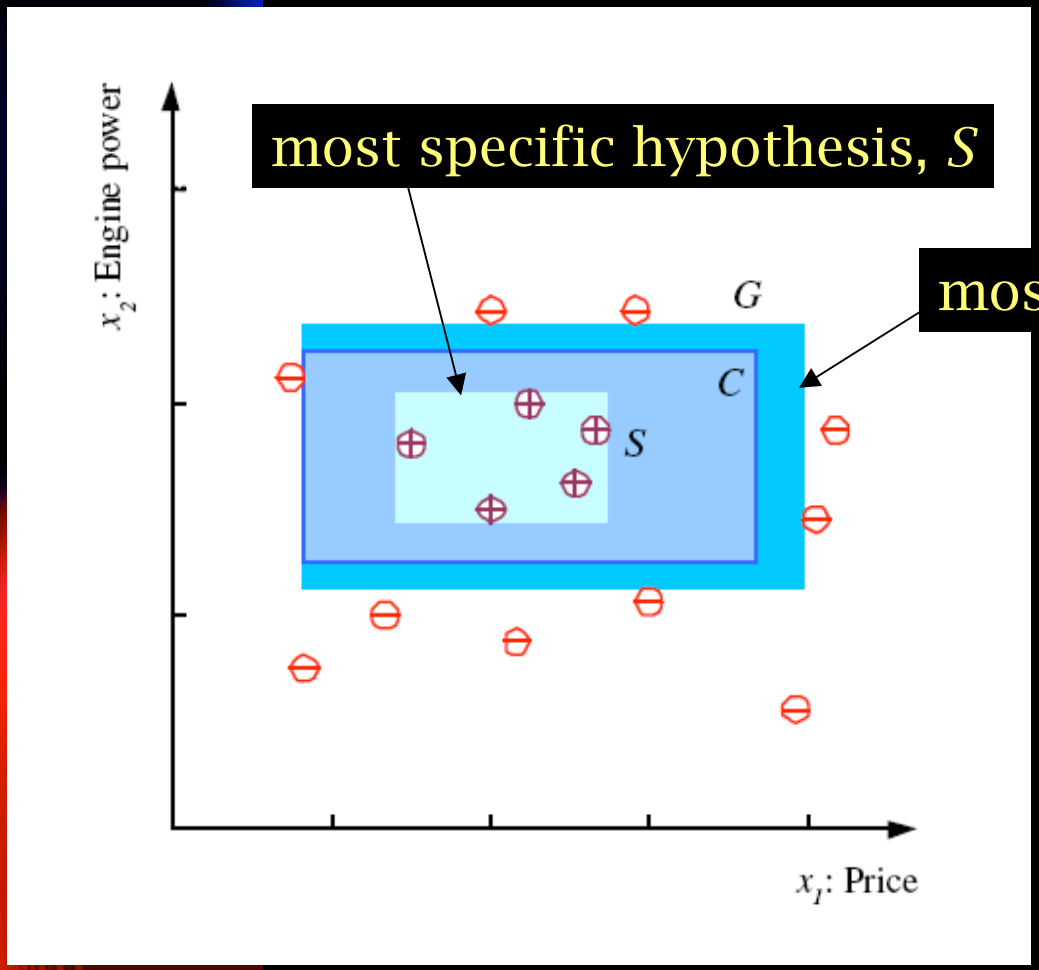


Error of h on $\mathcal{X}$

$$E(h \mid \mathcal{X}) = \sum_{t=1}^{N} 1\left(h\left(\mathbf{x}^t\right) \neq y^t\right)$$

(Minimize this!)

Version space: h consistent with $\mathcal{X}$

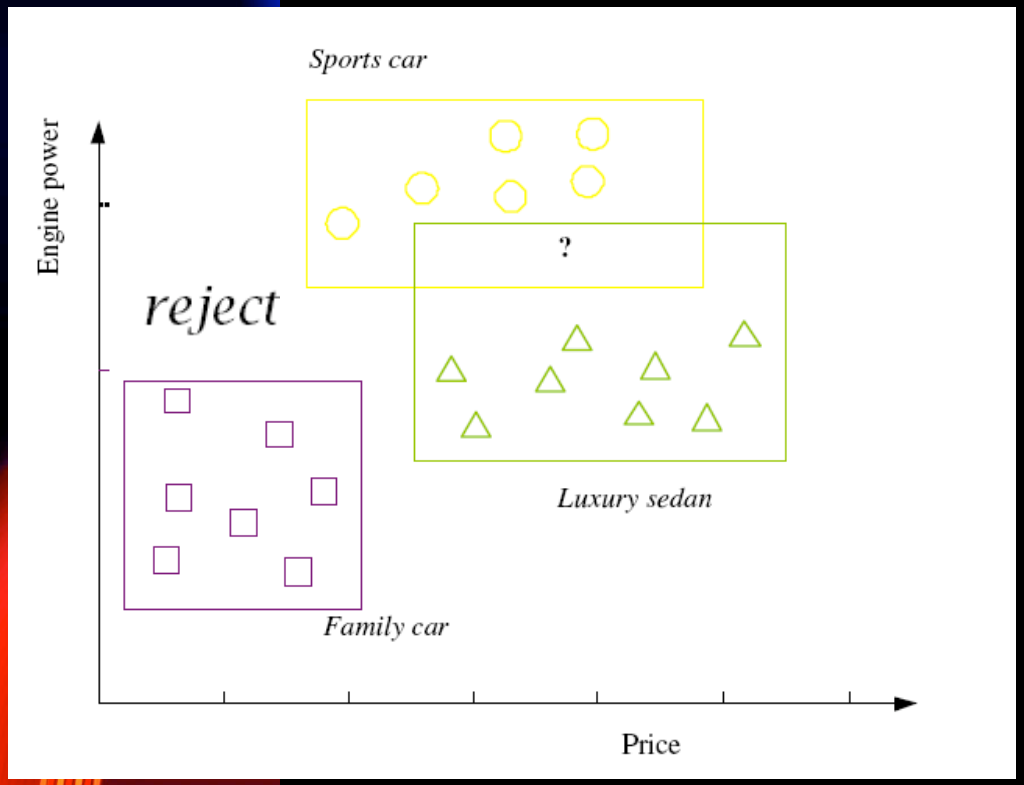


$h \in \mathcal{H}$, between S and G, are consistent with $\mathcal{X}$ (no errors)

They make up the version space

(Mitchell, 1997)

Learning Multiple Classes



$$\mathcal{X} = \{\mathbf{x}^t, y^t\}_{t=1}^N$$

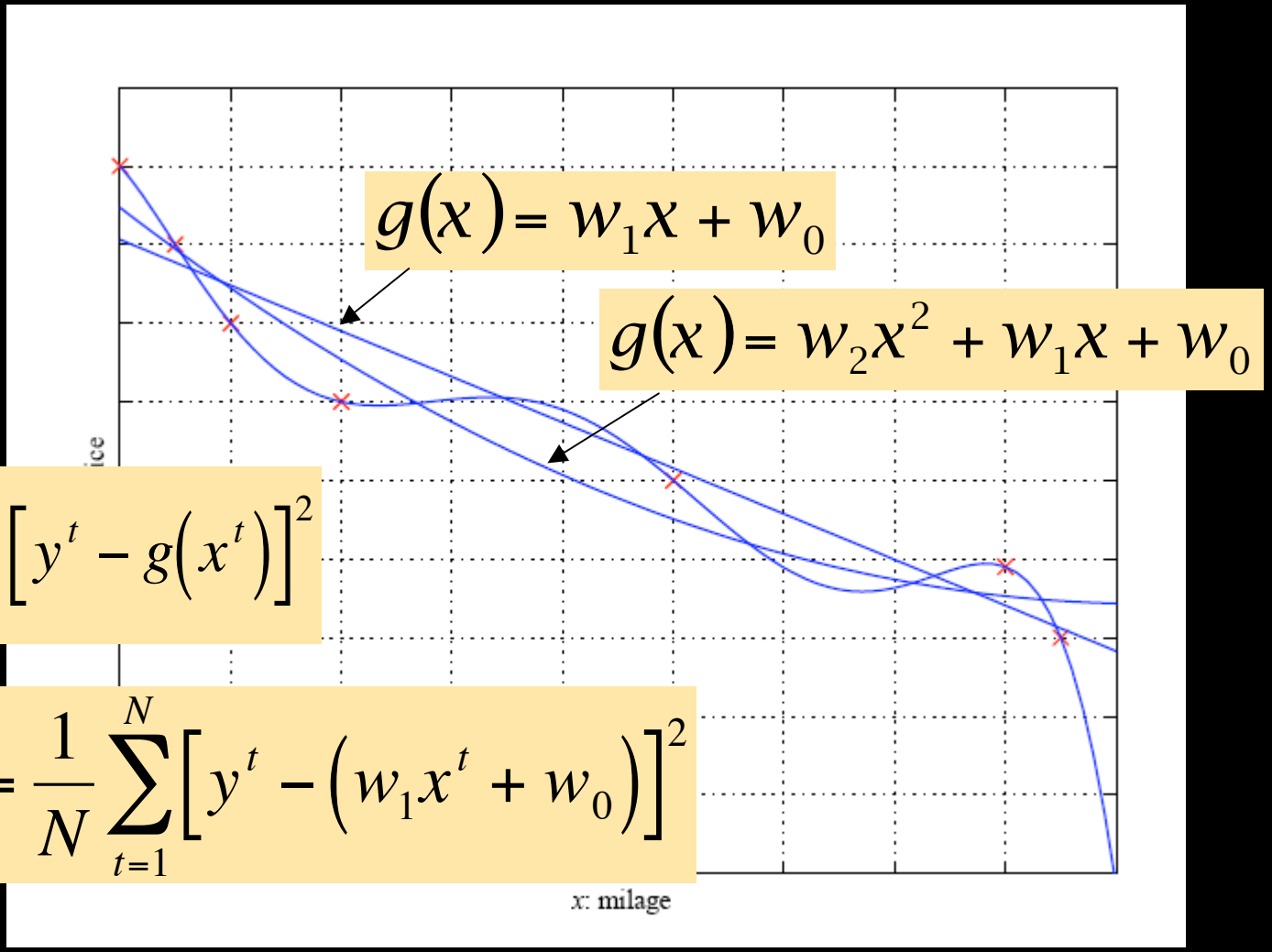
$$y_i^t = \begin{cases} 1 \text{ if } \mathbf{x}^t \in C_i \\ 0 \text{ if } \mathbf{x}^t \in C_j, j \neq i \end{cases}$$

Train K hypotheses $h_i(\mathbf{x}),\ i = 1,\ldots,K$:

$$h_i(\mathbf{x}^t) = \begin{cases} 1 \text{ if } \mathbf{x}^t \in C_i \\ 0 \text{ if } \mathbf{x}^t \in C_j, j \neq i \end{cases}$$

Regression: predict real value (with noise)

$$\mathcal{X} = \left\{x^t, y^t\right\}_{t=1}^{N}$$
$$y^t \in \Re$$
$$y^t = g\left(x^t\right) + \varepsilon$$



$$E\left(g \mid \mathcal{X}\right) = \frac{1}{N}\sum_{t=1}^{N}\left[y^t - g\left(x^t\right)\right]^2$$

$$E\left(w_1, w_0 \mid \mathcal{X}\right) = \frac{1}{N}\sum_{t=1}^{N}\left[y^t - \left(w_1 x^t + w_0\right)\right]^2$$

Issues in Supervised Learning

1. Representation: which features to use?
2. Model Selection: complexity, noise, bias
3. Evaluation: how well does it perform?

What you should know

- What is supervised learning?
  - Create model by optimizing loss function
  - Examples of supervised learning problems
- Features / representation, feature space
- Hypothesis space
- Version space
- Classification with multiple classes
- Regression

Instance-Based Learning

Chapter 8

Chapter 8: Nonparametric Methods

- "Nonparametric methods": ?
  - No explicit "model" of the concept being learned
  - Key: keep all the data (memorize)
  - = "lazy" or "memory-based" or "instance-based" or "case-based" learning
- Parametric methods:
  - Concept model is specified with one or more parameters
  - Key: keep a compact model, throw away individual data points
  - E.g., a Gaussian distribution; params = mean, std dev

Instance-Based Learning

- Build a database of previous observations
- To make a prediction for a new item *x'*, find the most similar database item *x* and use its output *f(x)* for *f(x')*
- Provides a local approximation to target function or concept

(neighbor)

- You need:
  1. A distance metric (to determine similarity)
  2. Number of neighbors to consult
  3. Method for combining neighbors' outputs

1-Nearest Neighbor

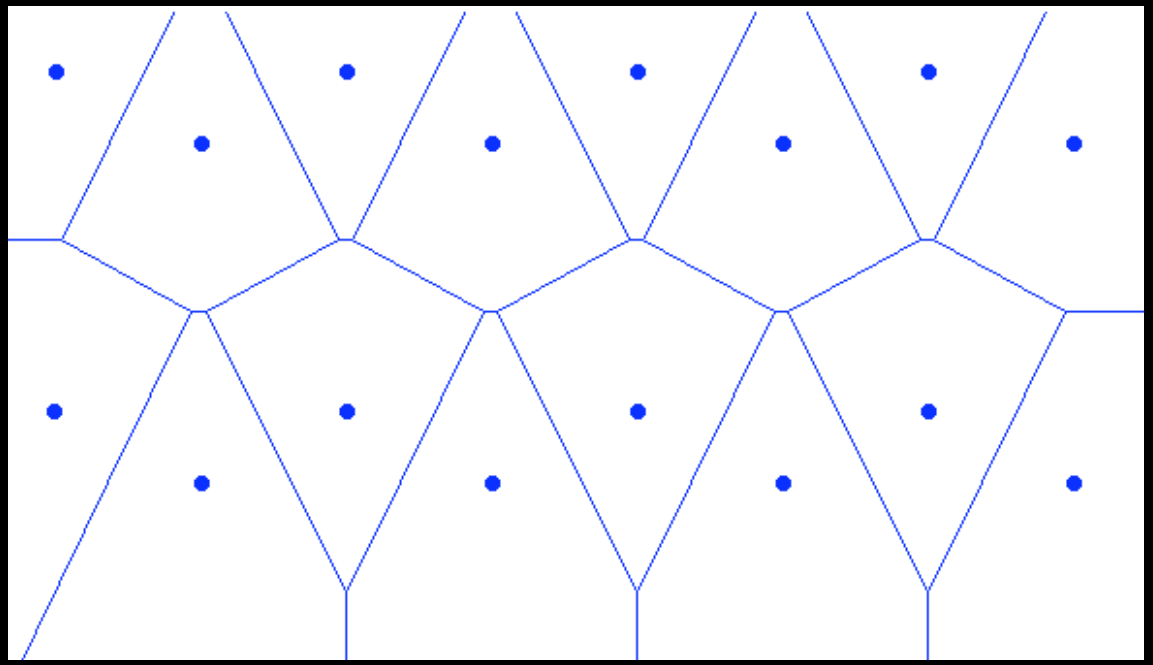
$$D(x, x') = \sqrt{\sum_{i=1}^{f} (x_i - x'_i)^2}$$

1. A distance metric: Euclidean
2. Number of neighbors to consult: 1
3. Combining neighbors' outputs: N/A

- Equivalent to memorizing everything you've ever seen and reporting the most similar result

In Feature Space…

- We can draw the 1-nearest-neighbor region for each item: a Voronoi diagram



- http://hirak99.googlepages.com/voronoi

1-NN Algorithm

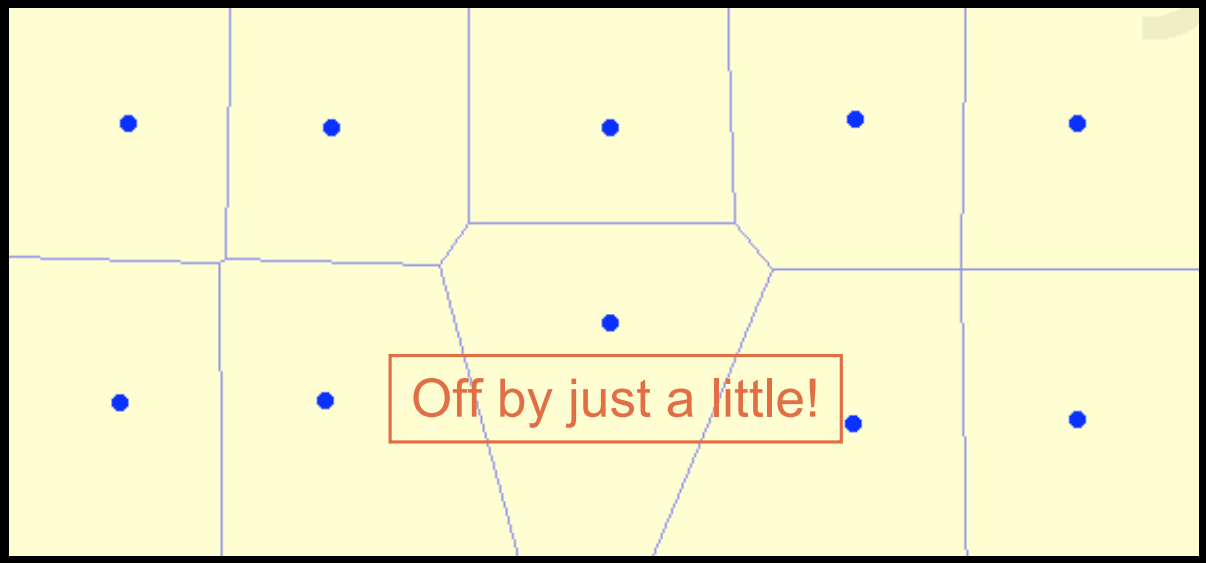
- Given training data (x_1, y_1) … (x_n, y_n), determine y_{new} for x_{new}

1. Find x' most similar to x_{new} using Euclidean dist
2. Assign $y_{new} = y'$

- Works for classification or regression

Drawbacks to 1-NN

- 1-NN fits the data exactly, including any noise



- May not generalize well to new data

k-Nearest Neighbors

1. A distance metric: Euclidean
2. Number of neighbors to consult: k
3. Combining neighbors' outputs:
   - Classification
     - Majority vote
     - Weighted majority vote: nearer have more influence
   - Regression
     - Average (real-valued)
     - Weighted average: nearer have more influence

$$y_{new} = \text{argmax}_c \sum_k (y_k = c)$$

$$y_{new} = \text{argmax}_c \sum_k w_k (y_k = c)$$

$$y_{new} = \sum_k \frac{1}{k} y_k$$

$$y_{new} = \sum_k w_k y_k$$

- Result: *Smoother*, more generalizable result

Choosing k

- K is a parameter of the k-NN algorithm
  - This does not make it "parametric". Confusing!
- Recall: set parameters using validation data set
  - Not the training set (overfitting)

Computational Complexity (cost)

- How expensive is it to perform k-NN on a new instance?
  - O(n) to find the nearest neighbor
  - The more you know, the longer it takes to make a decision!
  - Can be reduced to O(log n) using kd-trees

Summary of k-Nearest Neighbors

- Pros
  - k-NN is simple! (to understand, implement)
    - You'll get to try it out in Homework 1!
  - Often used as a baseline for other algorithms
  - "Training" is fast: just add new item to database
- Cons
  - Most work done at query time: may be expensive
  - Must store O(n) data for later queries
  - Performance is sensitive to choice of distance metric
    - And normalization of feature values

What you should know

- Parametric vs. nonparametric methods
- Instance-based learning
- 1-NN, k-NN
  - k-NN classification and regression
  - How to choose k?
- Pros and cons of nearest-neighbor approaches

Homework 1

Due Jan. 10, 2008
Midnight

Three parts

1. Find a newsworthy machine learning product or discovery online; write 2 paragraphs about it
2. Written questions
3. Programming (Java)
   - Implement 1-nearest-neighbor algorithm
   - Evaluate it on two data sets
   - Analyze the results

Final Project

Proposal due 1/19

Project due 3/8

1. Pick a problem that interests you

- Classification
  - Male vs. female?
  - Left-handed vs. right-handed?
  - Predict grade in a class?
  - Recommend a product (e.g., type of MP3 player)?
- Regression
  - Stock market prediction?
  - Rainfall prediction?

2. Create or obtain a data set

- Tons of data sets are available online… or you can create your own
  - Must have at least 100 instances
- What features will you use to represent the data?
  - Even if using an existing data set, you might select only the features that are relevant to your problem

3. Pick a machine learning algorithm to solve it

- Classification
  - k-nearest neighbors
  - Decision trees
  - Support Vector Machines
  - Neural Networks
- Regression
  - k-nearest neighbors
  - Support Vector Machines
  - Neural Networks
  - Naïve Bayes
- Justify your choice

4. Design experiments

- What metrics will you use?
  - We'll cover evaluation methods in Lectures 2 and 3
- What baseline algorithm will you compare to?
  - k-Nearest Neighbors is a good one
  - Classification: Predict most common class
  - Regression: Predict average output

Project Requirements

- Proposal (30 points):
  - Due midnight, Jan. 19
- Report (70 points):
  - Your choice:
    - Oral presentation (March 8) + 1-page report
    - 4-page report
  - Reports due midnight, March 8
  - Maximum of 15 oral presentations
- Project is 25% of your grade

Next Time

- Decision Trees (read Ch. 9)
- Rule Learning
- Evaluation (read Ch. 14.1-14.3, 14.6)
- Weka: Java machine learning library (read Weka Explorer Guide)