CS532 ANN

Machine learning crash course

PROBABILITY

Bayes' theorem

$$P(x, y) = P(y|x)P(x) = P(x|y)P(y)$$

likelihood

prior

$$P(y|x) = \frac{P(x|y)P(y)}{P(x)}$$

posterior

Prior - belief before making a particular obs.

Posterior - belief after making the obs.

Posterior is the prior for the next observation

- Intrinsically incremental

Learning

- “A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P , if its performance at tasks in T , as measured by P , improves with experience E” (Mitchell, 1997).

The Task, T

- Classification:

$f : Rn \rightarrow \{1, . . . , k\}$

- Regression

$f : Rn \rightarrow R$

- Transcription: Unstructured -> discrete
- Density or probability function estimation

Why "Learn" ?

- Machine learning is programming computers to optimize a performance criterion using example data or past experience.
- There is no need to "learn" to calculate payroll
- Learning is used when:
  - Human expertise does not exist (navigating on Mars),
  - Humans are unable to explain their expertise (speech recognition)
  - Solution changes in time (routing on a computer network)
  - Solution needs to be adapted to particular cases (user biometrics)

What We Talk About When We Talk About"Learning"

- Learning general models from a data of particular examples
- Data is cheap and abundant (data warehouses, data marts); knowledge is expensive and scarce.
- Example in retail: Customer transactions to consumer behavior:

    People who bought "Blink" also bought "Outliers" (www.amazon.com)

- Build a model that is *a good and useful approximation* to the data.

Data Mining

- Retail: Market basket analysis, Customer relationship management (CRM)
- Finance: Credit scoring, fraud detection
- Manufacturing: Control, robotics, troubleshooting
- Medicine: Medical diagnosis
- Telecommunications: Spam filters, intrusion detection
- Bioinformatics: Motifs, alignment
- Web mining: Search engines
- ...

What is Machine Learning?

- Optimize a performance criterion using example data or past experience.
- Role of Statistics: Inference from a sample
- Role of Computer science: Efficient algorithms to
  - Solve the optimization problem
  - Representing and evaluating the model for inference

Applications

- Association
- Supervised Learning
  - Classification
  - Regression
- Unsupervised Learning
- Reinforcement Learning

Learning Associations

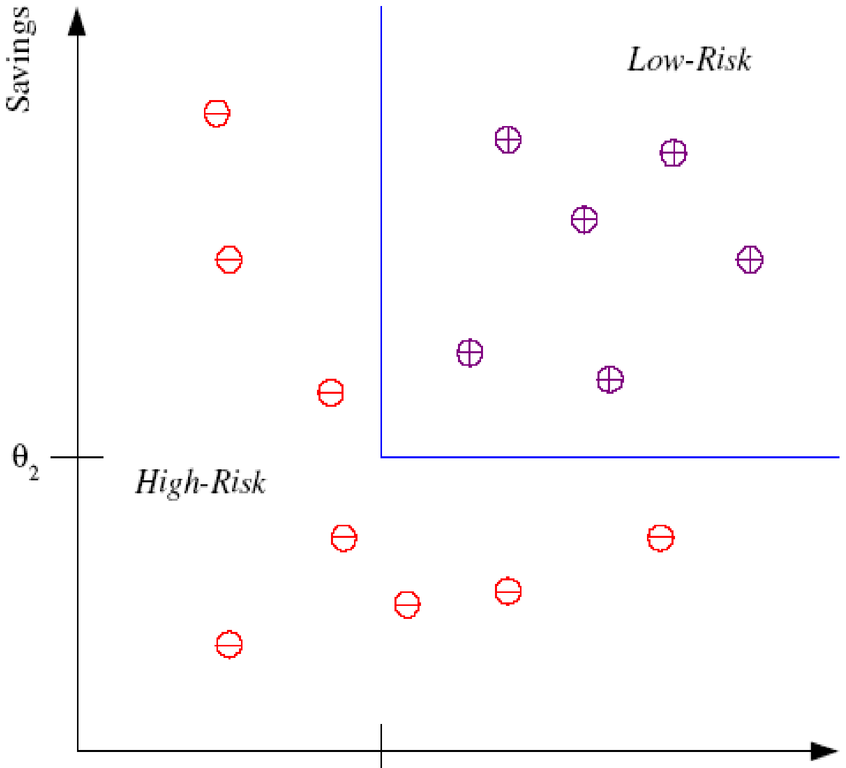
- Basket analysis:

  $P(Y \mid X)$ probability that somebody who buys X also buys Y where X and Y are products/services.

  Example: $P(\text{chips} \mid \text{beer}) = 0.7$

Classification

- Example: Credit scoring
- Differentiating between low-risk and high-risk customers from their *income* and *savings*



Discriminant: IF *income* > θ_1 AND *savings* > θ_2

THEN low-risk ELSE high-risk

Classification: Applications

- Aka Pattern recognition
- Face recognition: Pose, lighting, occlusion (glasses, beard), make-up, hair style
- Character recognition: Different handwriting styles.
- Speech recognition: Temporal dependency.
- Medical diagnosis: From symptoms to illnesses
- Biometrics: Recognition/authentication using physical and/or behavioral characteristics: Face, iris, signature, etc
- ...

Face Recognition

Training examples of a person

Test images

ORL dataset,
AT&T Laboratories, Cambridge UK

Regression

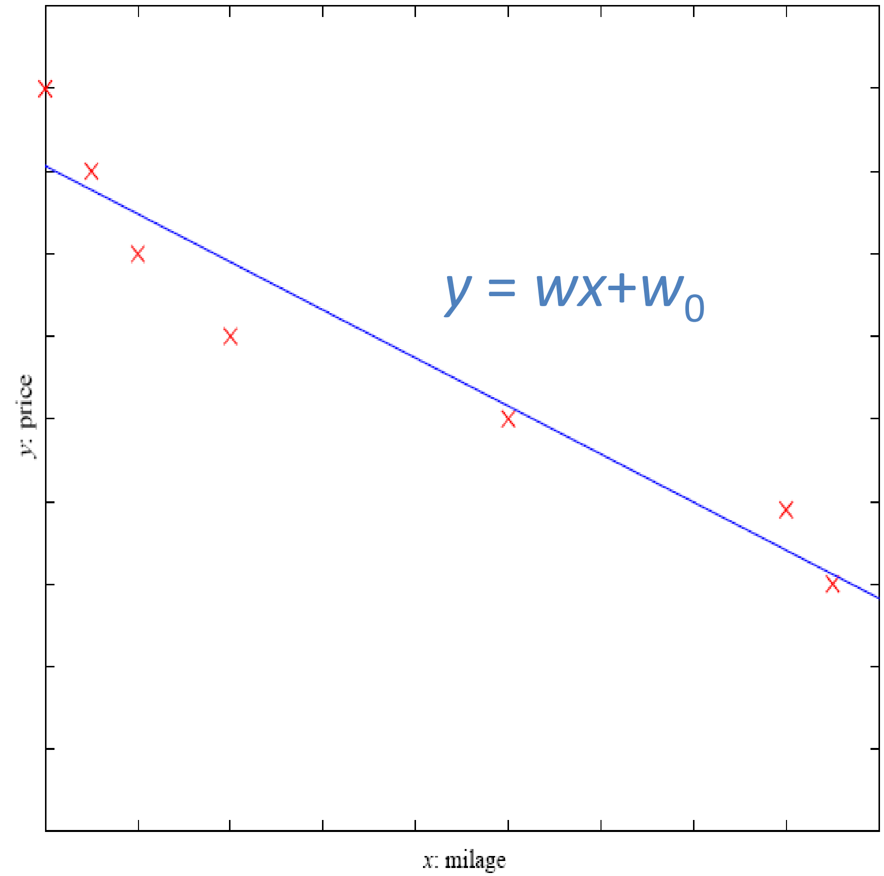
- Example: Price of a used car
- x : car attributes

  y : price

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

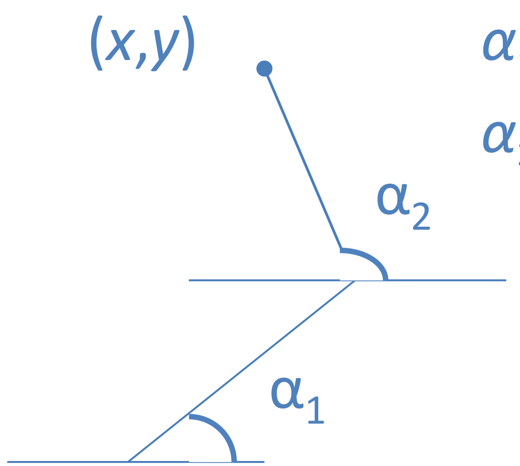
  $g(\,)$ model,

  θ parameters



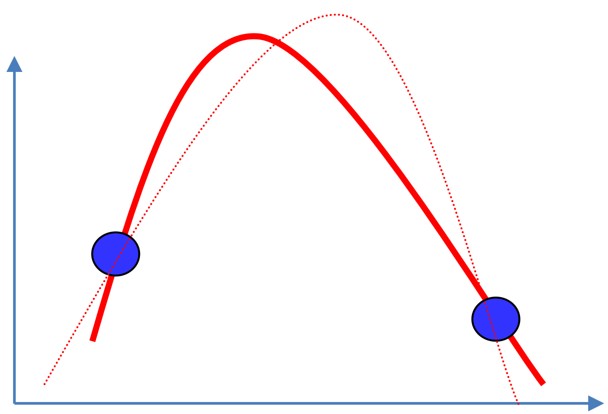
Regression Applications

- Navigating a car: Angle of the steering
- Kinematics of a robot arm



$$\alpha_1 = g_1(x,y)$$
$$\alpha_2 = g_2(x,y)$$

- Response surface design



Supervised Learning: Uses

- Prediction of future cases: Use the rule to predict the output for future inputs
- Knowledge extraction: The rule is easy to understand
- Compression: The rule is simpler than the data it explains
- Outlier detection: Exceptions that are not covered by the rule, e.g., fraud

Unsupervised Learning

- Learning “what normally happens”
- No output
- Clustering: Grouping similar instances
- Example applications
  - Customer segmentation in CRM
  - Image compression: Color quantization
  - Bioinformatics: Learning motifs

SUPERVISED LEARNING

Learning a Class from Examples

- Class C of a “family car”
  - Prediction: Is car x a family car?
  - Knowledge extraction: What do people expect from a family car?
- Output:

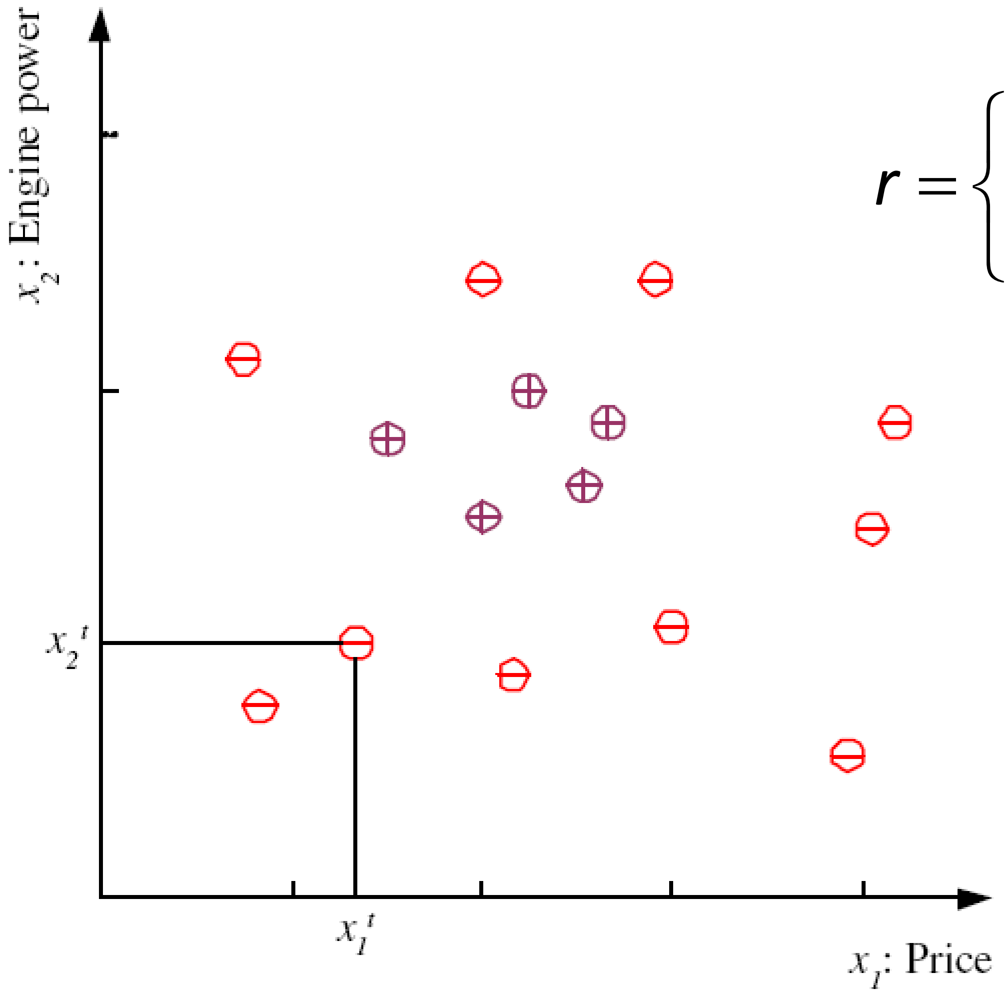
  Positive (+) and negative (–) examples
- Input representation:

  x_1: price, x_2 : engine power

Training set $\mathcal{X}$

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

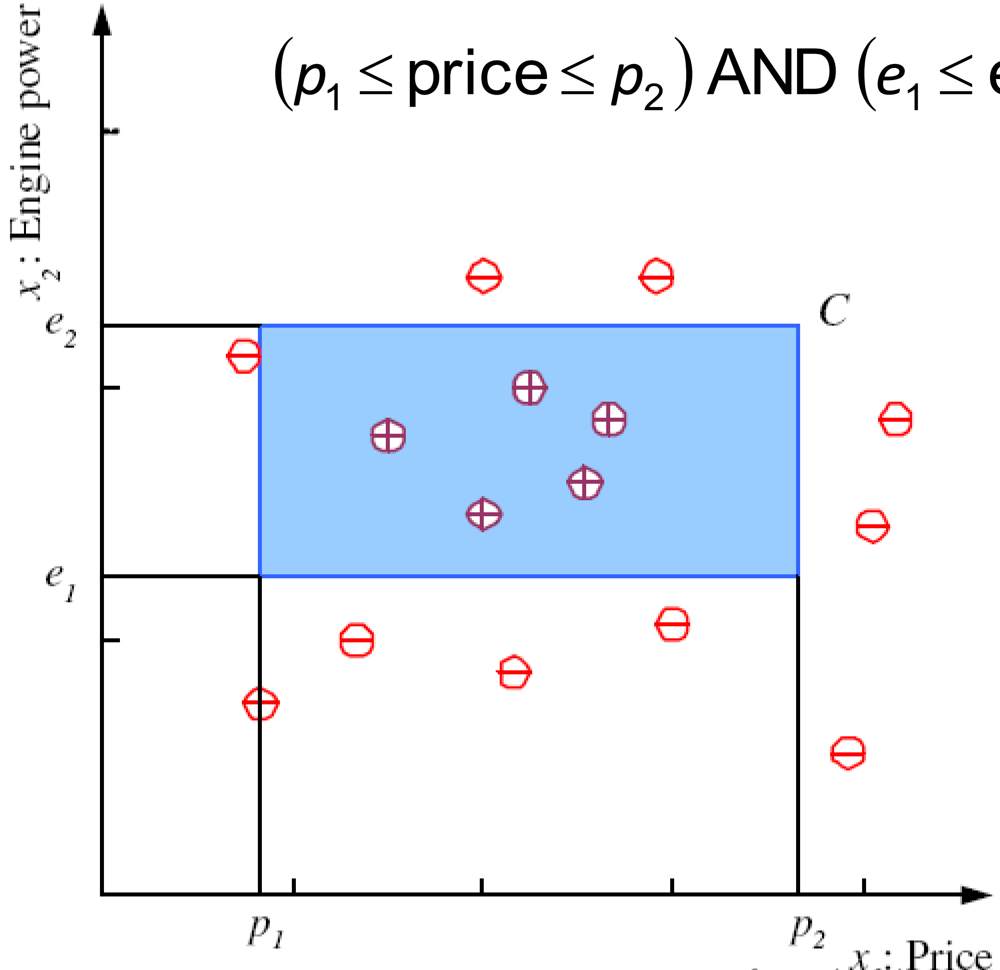
$$r = \begin{cases} 1 \text{ if } \mathbf{x} \text{ is positive} \\ 0 \text{ if } \mathbf{x} \text{ is negative} \end{cases}$$



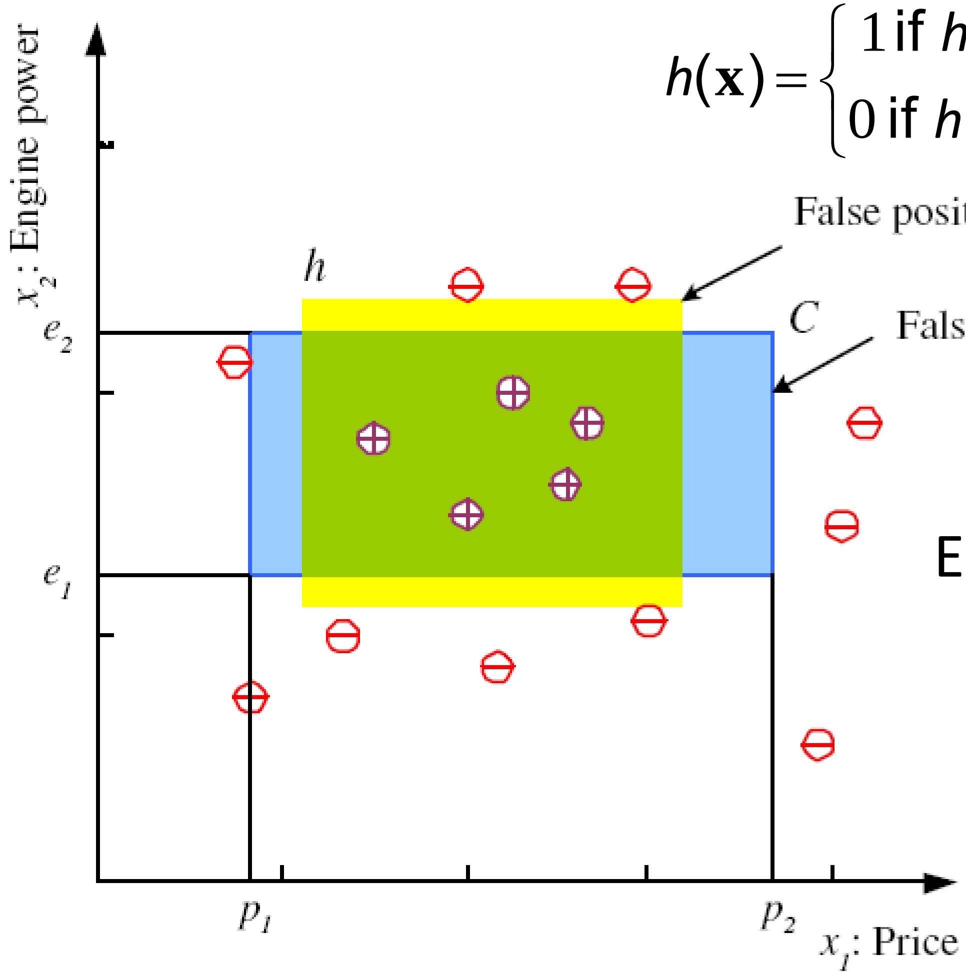
$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

Class C

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



Hypothesis class $\mathcal{H}$



$$h(\mathbf{x}) = \begin{cases} 1 \text{ if } h \text{ says } \mathbf{x} \text{ is positive} \\ 0 \text{ if } h \text{ says } \mathbf{x} \text{ is negative} \end{cases}$$

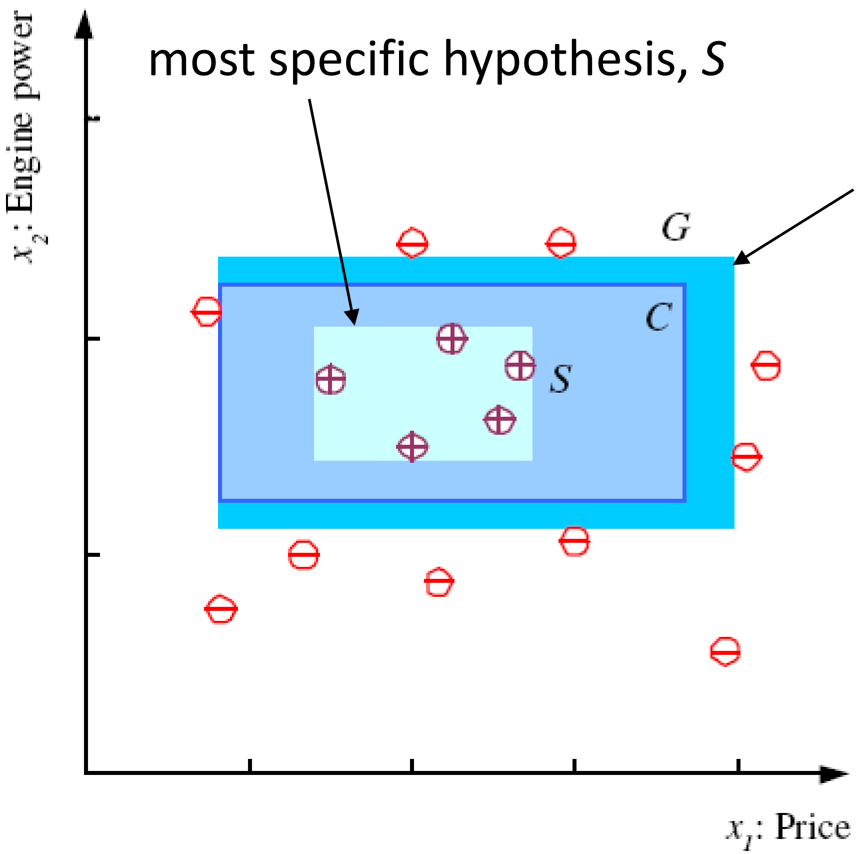
Error of h on $\mathcal{H}$

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

S, G, and the Version Space

most specific hypothesis, *S*

most general hypothesis, *G*

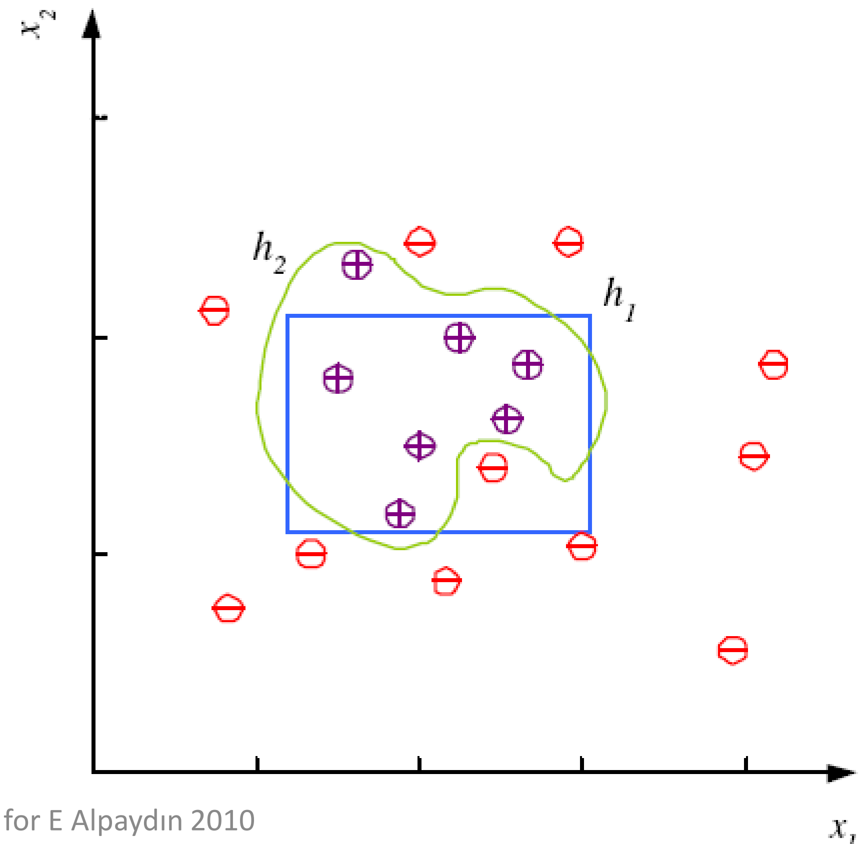


h ∈ H, between *S* and *G* is consistent and make up the version space (Mitchell, 1997)

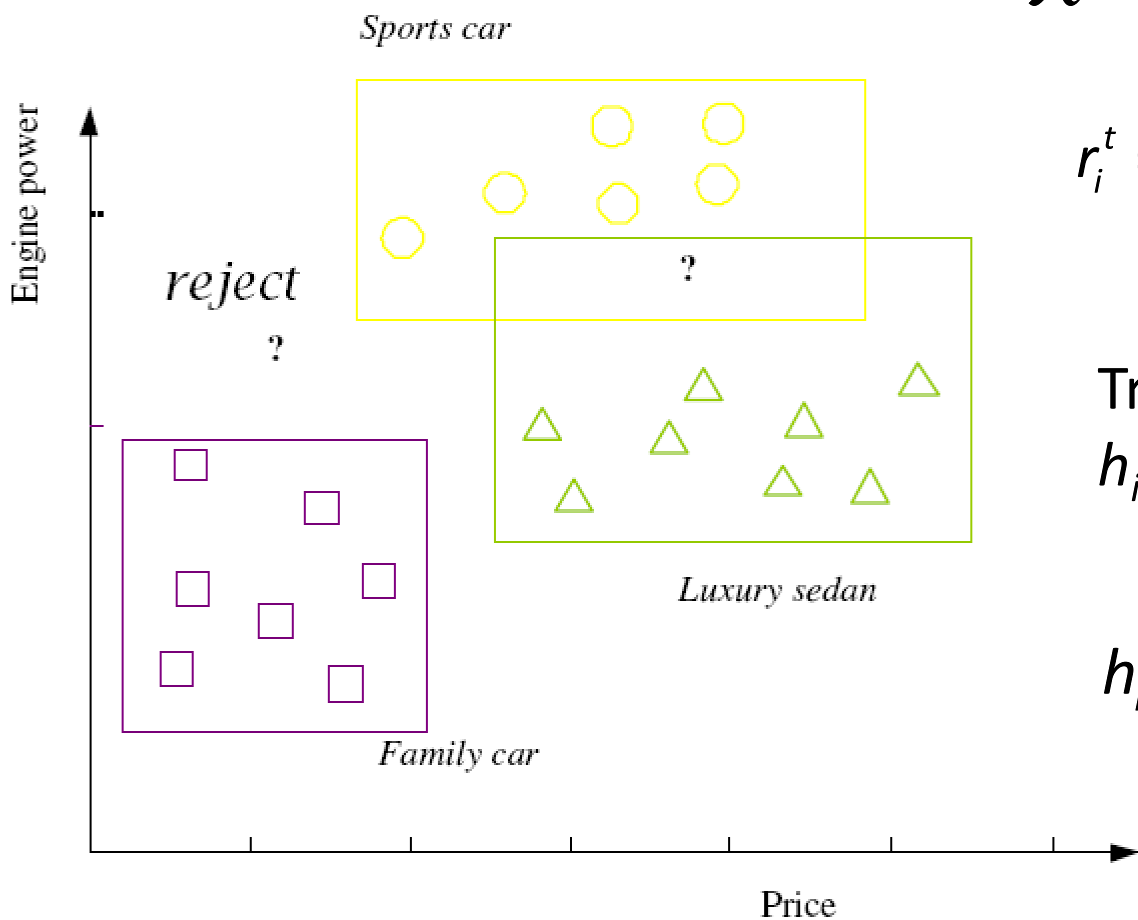
Noise and Model Complexity

Use the simpler one because

- Simpler to use (lower computational complexity)
- Easier to train (lower space complexity)
- Easier to explain (more interpretable)
- Generalizes better (lower variance - Ockham's razor)



Multiple Classes, C_i i=1,...,K



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

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

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

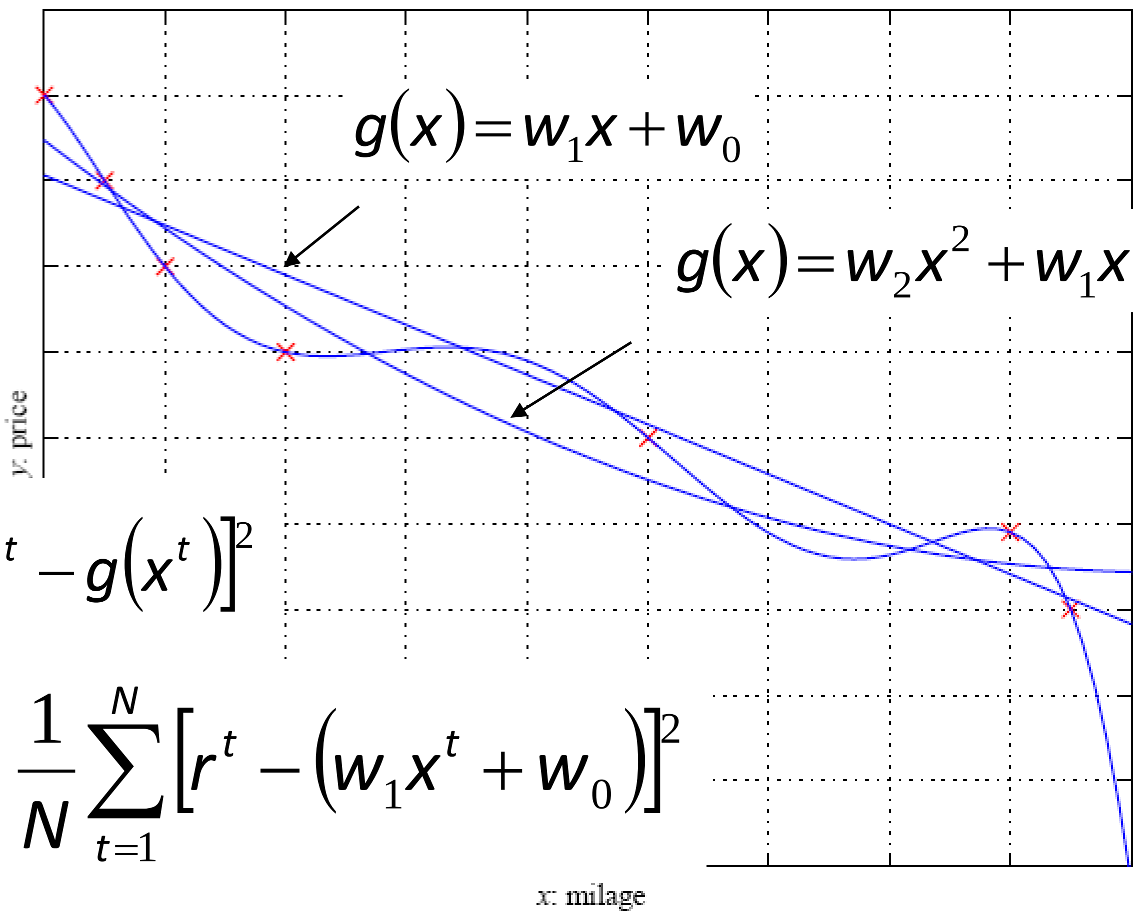
$$h_i\left(\mathbf{x}^t\right) = \begin{cases} 1 \text{ if } \mathbf{x}^t \in C_i \\ 0 \text{ if } \mathbf{x}^t \in C_j, j \neq i \end{cases}$$

Regression

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

$$r^t \in \Re$$

$$r^t = f(x^t) + \varepsilon$$



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

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

Model Selection & Generalization

- Learning is an ill-posed problem; data is not sufficient to find a unique solution
- The need for inductive bias, assumptions about $\mathcal{H}$
- Generalization: How well a model performs on new data
- Overfitting: $\mathcal{H}$ more complex than C or f
- Underfitting: $\mathcal{H}$ less complex than C or f

Triple Trade-Off

- There is a trade-off between three factors (Dietterich, 2003):
  1. Complexity of $\mathcal{H}$, $c(\mathcal{H})$,
  2. Training set size, N,
  3. Generalization error, E, on new data

☐ As N, $E\downarrow$

☐ As $c(\mathcal{H})$, first $E\downarrow$ and then E

Cross-Validation

- To estimate generalization error, we need data unseen during training. We split the data as
  - Training set (50%)
  - Validation set (25%)
  - Test (publication) set (25%)
- Resampling when there is few data

Dimensions of a Supervised Learner

1. Model: $g(\mathbf{x} \mid \theta)$
2. Loss function: $E(\theta \mid \mathcal{X}) = \sum_t L\left(r^t, g\left(\mathbf{x}^t \mid \theta\right)\right)$
3. Optimization procedure:

$$\theta^* = \arg\min_\theta E(\theta \mid \mathcal{X})$$

BAYSIAN DESCISION THEORY

Probability and Inference

- Result of tossing a coin is $\in$ {Heads,Tails}
- Random var $X \in \{1,0\}$

  Bernoulli: $P\{X=1\} = p_o^{X}(1 - p_o)^{(1 - X)}$
- Sample: $\boldsymbol{X} = \{x^t\}_{t=1}^{N}$

  Estimation: $p_o = \#\{\text{Heads}\}/\#\{\text{Tosses}\} = \sum_t x^t / N$
- Prediction of next toss:

  Heads if $p_o > ½$, Tails otherwise

Classification

- Credit scoring: Inputs are income and savings.

  Output is low-risk vs high-risk
- Input: $\boldsymbol{x} = [x_1, x_2]^T$,Output: C Î {0,1}
- Prediction:

$$\text{choose: } \begin{cases} C = 1 \text{ if } P(C = 1 \mid x_1, x_2) > 0.5 \\ C = 0 \text{ otherwise} \end{cases}$$

or

$$\text{choose } \begin{cases} C = 1 \text{ if } P(C = 1 \mid x_1, x_2) > P(C = 0 \mid x_1, x_2) \\ C = 0 \text{ otherwise} \end{cases}$$

Bayes' Rule

prior *likelihood*

posterior

$$P(C \mid \mathbf{x}) = \frac{P(C)\,p(\mathbf{x} \mid C)}{p(\mathbf{x})}$$

evidence

$$P(C=0) + P(C=1) = 1$$

$$p(\mathbf{x}) = p(\mathbf{x} \mid C=1)P(C=1) + p(\mathbf{x} \mid C=0)P(C=0)$$

$$p(C=0 \mid \mathbf{x}) + P(C=1 \mid \mathbf{x}) = 1$$

Bayes' Rule: *K*>2 Classes

$$P(C_i \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid C_i)P(C_i)}{p(\mathbf{x})}$$

$$= \frac{p(\mathbf{x} \mid C_i)P(C_i)}{\sum_{k=1}^{K} p(\mathbf{x} \mid C_k)P(C_k)}$$

$P(C_i) \geq 0$ and $\sum_{i=1}^{K} P(C_i) = 1$

choose C_i if $P(C_i \mid \mathbf{x}) = \max_k P(C_k \mid \mathbf{x})$

Losses and Risks

- Actions: α_i
- Loss of α_i when the state is C_k : λ_{ik}
- Expected risk (Duda and Hart, 1973)

$$R(\alpha_i \mid \mathbf{x}) = \sum_{k=1}^{K} \lambda_{ik} P(C_k \mid \mathbf{x})$$

$$\text{choose } \alpha_i \text{ if } R(\alpha_i \mid \mathbf{x}) = \min_k R(\alpha_k \mid \mathbf{x})$$

Losses and Risks: 0/1 Loss

$$\lambda_{ik} = \begin{cases} 0 \text{ if } i = k \\ 1 \text{ if } i \neq k \end{cases}$$

$$\begin{aligned} R(\alpha_i \mid \mathbf{x}) &= \sum_{k=1}^{K} \lambda_{ik} P(C_k \mid \mathbf{x}) \\ &= \sum_{k \neq i} P(C_k \mid \mathbf{x}) \\ &= 1 - P(C_i \mid \mathbf{x}) \end{aligned}$$

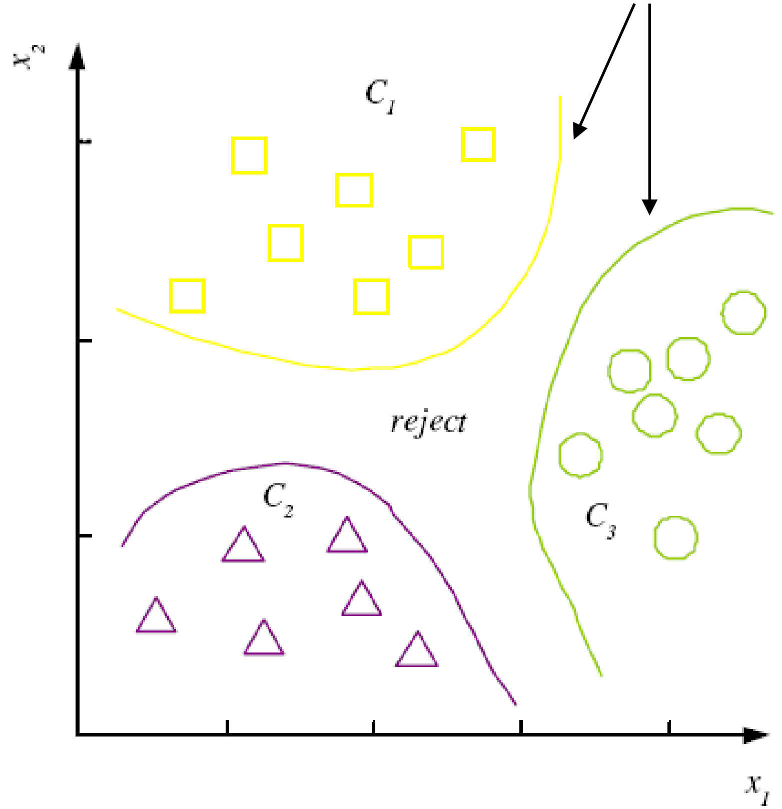
For minimum risk, choose the most probable class

Discriminant Functions

choose C_i if $g_i(\mathbf{x}) = \max_k g_k(\mathbf{x})$

$g_i(\mathbf{x}), i = 1, \ldots, K$

$$g_i(\mathbf{x}) = \begin{cases} -R(\alpha_i \mid \mathbf{x}) \\ P(C_i \mid \mathbf{x}) \\ p(\mathbf{x} \mid C_i) P(C_i) \end{cases}$$



K decision regions $\mathcal{R}_1, \ldots, \mathcal{R}_K$

$$\mathcal{R}_i = \{\mathbf{x} \mid g_i(\mathbf{x}) = \max_k g_k(\mathbf{x})\}$$

K=2 Classes

- Dichotomizer (K=2) vs Polychotomizer (K>2)
- $g(\boldsymbol{x}) = g_1(\boldsymbol{x}) - g_2(\boldsymbol{x})$

$$\text{choose } \begin{cases} C_1 \text{ if } g(\mathbf{x}) > 0 \\ C_2 \text{ otherwise} \end{cases}$$

- *Log odds:* $\log \frac{P(C_1 \mid \mathbf{x})}{P(C_2 \mid \mathbf{x})}$

PARAMETRIC METHODS

Parametric Estimation

- $\mathcal{X} = \{ x^t \}_t$ where $x^t \sim p(x)$
- Parametric estimation:

  Assume a form for $p(x \mid \theta)$ and estimate θ, its sufficient statistics, using X

  e.g., N (μ, σ^2) where $\theta = \{ \mu, \sigma^2 \}$

Maximum Likelihood Estimation

- Likelihood of θ given the sample $\mathcal{X}$

  $l(\vartheta|\mathcal{X}) = p(\mathcal{X}|\vartheta) = \prod_t p(x^t|\vartheta)$

- Log likelihood

  $\mathcal{L}(\vartheta|\mathcal{X}) = \log l(\vartheta|\mathcal{X}) = \sum_t \log p(x^t|\vartheta)$

- Maximum likelihood estimator (MLE)

  $\vartheta^* = \mathrm{argmax}_\vartheta \mathcal{L}(\vartheta|\mathcal{X})$

Examples: Bernoulli/Multinomial

- Bernoulli: Two states, failure/success, x in {0,1}

  $P(x) = p_o^{x}(1 - p_o)^{(1 - x)}$

  $\mathcal{L}(p_o | \mathcal{X}) = \log \prod_t p_o^{x^t}(1 - p_o)^{(1 - x^t)}$

  MLE: $p_o = \sum_t x^t / N$

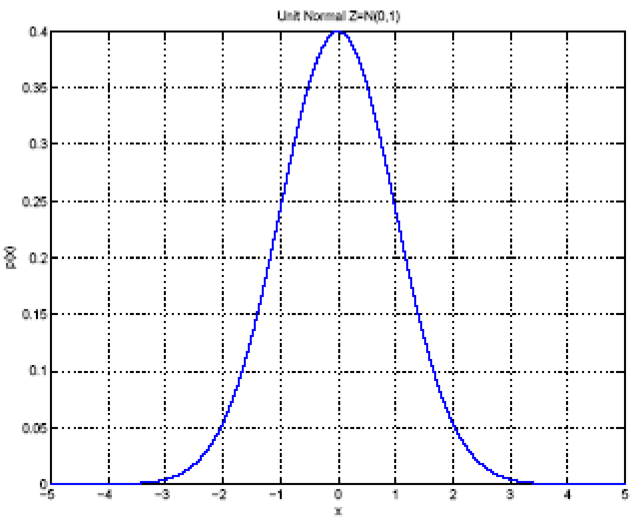
- Multinomial: K>2 states, x_i in {0,1}

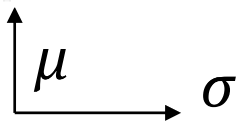
  $P(x_1, x_2, \ldots, x_K) = \prod_i p_i^{x_i}$

  $\mathcal{L}(p_1, p_2, \ldots, p_K | \mathcal{X}) = \log \prod_t \prod_i p_i^{x_i^t}$

  MLE: $p_i = \sum_t x_i^t / N$

Gaussian (Normal) Distribution





- $p(x) = \mathcal{N}(\mu, \sigma^2)$

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right]$$

- MLE for μ and σ^2:

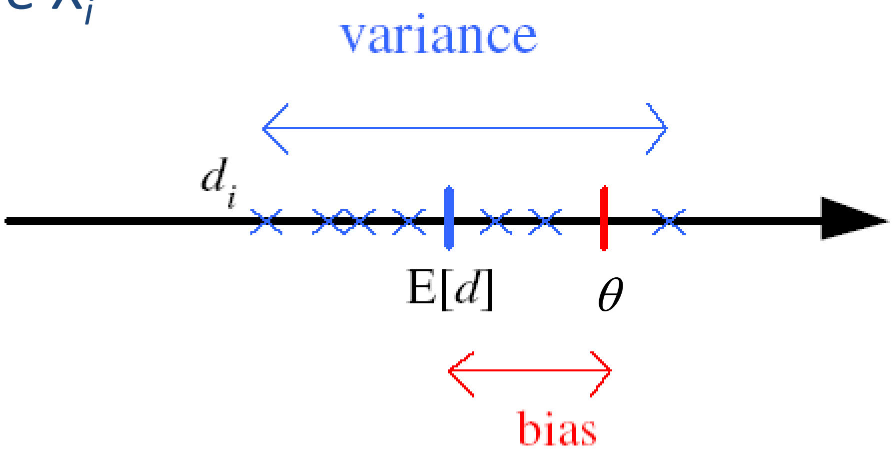
$$m = \frac{\sum_t x^t}{N}$$

$$s^2 = \frac{\sum_t \left(x^t - m\right)^2}{N}$$

Bias and Variance

Unknown parameter θ
Estimator $d_i = d(X_i)$ on sample X_i

Bias: $b_\theta(d) = E[d] - \theta$
Variance: $E[(d-E[d])^2]$



Mean square error:
$r(d,\theta) = E[(d-\theta)^2]$
$= (E[d] - \theta)^2 + E[(d-E[d])^2]$
$=$ Bias^2 + Variance

Bayes' Estimator

- Treat ϑ as a random var with prior $p\ (\vartheta)$
- Bayes' rule: $p\ (\vartheta|\mathcal{X}) = p(\mathcal{X}|\vartheta)\ p(\vartheta)\ /\ p(\mathcal{X})$
- Full: $p(x|\mathcal{X}) = \int p(x|\vartheta)\ p(\vartheta|\mathcal{X})\ d\vartheta$
- Maximum a Posteriori (MAP): $\vartheta_{\mathrm{MAP}} = \mathrm{argmax}_{\vartheta}\ p(\vartheta|\mathcal{X})$
- Maximum Likelihood (ML): $\vartheta_{\mathrm{ML}} = \mathrm{argmax}_{\vartheta}\ p(\mathcal{X}|\vartheta)$
- Bayes': $\vartheta_{\mathrm{Bayes'}} = \mathrm{E}[\vartheta|\mathcal{X}] = \int \vartheta\ p(\vartheta|\mathcal{X})\ d\vartheta$

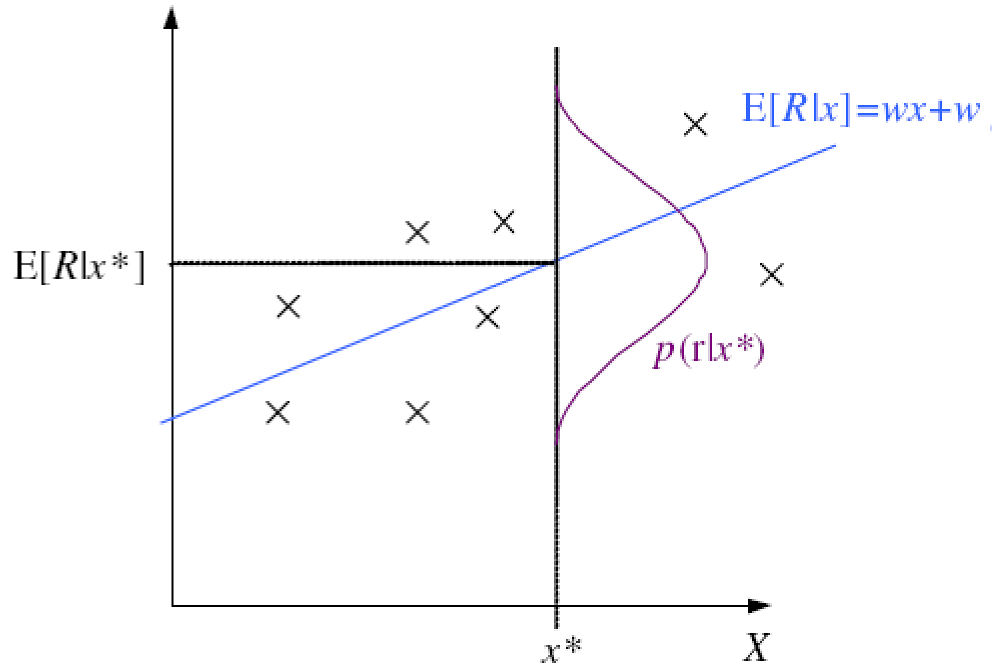
Regression

$$r = f(x) + \varepsilon$$

$$\text{estimator}: g(x \mid \theta)$$

$$\varepsilon \sim \mathcal{N}(0, \sigma^2)$$

$$p(r \mid x) \sim \mathcal{N}(g(x \mid \theta), \sigma$$



$$\mathcal{L}(\theta \mid \mathcal{X}) = \log \prod_{t=1}^{N} p(x \;\; , r \;\;)$$

$$= \log \prod_{t=1}^{N} p(r^t \mid x^t) + \log \prod_{t=1}^{N} p(x^t)$$

Regression: From LogL to Error

$$\mathcal{L}(\theta|\mathcal{X}) = \log \prod_{t=1}^{N} \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{\left[r^t - g(x^t|\theta)\right]^2}{2\sigma^2}\right]$$

$$= -N \log \sqrt{2\pi}\sigma - \frac{1}{2\sigma^2} \sum_{t=1}^{N} \left[r^t - g(x^t|\theta)\right]^2$$

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

Linear Regression

$$g\left(x^t \mid w_1, w_0\right) = w_1 x^t + w_0$$

$$\sum_t r^t = N w_0 + w_1 \sum_t x^t$$

$$\sum_t r^t x^t = w_0 \sum_t x^t + w_1 \sum_t \left(x^t\right)^2$$

$$\mathbf{A} = \begin{bmatrix} N & \sum_t x^t \\ \sum_t x^t & \sum_t \left(x^t\right)^2 \end{bmatrix} \quad \mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} \quad \mathbf{y} = \begin{bmatrix} \sum_t r^t \\ \sum_t r^t x^t \end{bmatrix}$$

$$\mathbf{w} = \mathbf{A}^{-1}\mathbf{y}$$

Polynomial Regression

$$g\left(x^t \mid w_k,\ldots,w_2,w_1,w_0\right)=w_k\left(x^t\right)^k+\cdots+w_2\left(x^t\right)^2+w_1x^t+w_0$$

$$\mathbf{D}=\begin{bmatrix} 1 & x^1 & \left(x^1\right)^2 & \cdots & \left(x^1\right)^k \\ 1 & x^2 & \left(x^2\right)^2 & \cdots & \left(x^2\right)^k \\ \vdots & & & & \\ 1 & x^N & \left(x^N\right)^2 & \cdots & \left(x^N\right)^2 \end{bmatrix} \quad \mathbf{r}=\begin{bmatrix} r^1 \\ r^2 \\ \vdots \\ r^N \end{bmatrix}$$

$$\mathbf{w}=\left(\mathbf{D}^T\mathbf{D}\right)^{-1}\mathbf{D}^T\mathbf{r}$$

Other Error Measures

- Square Error: $$E(\theta|\mathcal{X}) = \frac{1}{2}\sum_{t=1}^{N}\left[r^t - g\left(x^t|\theta\right)\right]^2$$
- Relative Square Error: $$E(\theta|\mathcal{X}) = \frac{\sum_{t=1}^{N}\left[r^t - g\left(x^t|\theta\right)\right]^2}{\sum_{t=1}^{N}\left[r^t - \bar{r}\right]^2}$$
- Absolute Error: $E(\vartheta|X) = \sum_t |r^t - g(x^t|\vartheta)|$
- ε-sensitive Error:

$$E(\vartheta|X) = \sum_t 1(|r^t - g(x^t|\vartheta)| > \varepsilon)\,(|r^t - g(x^t|\vartheta)| - \varepsilon)$$

Model Selection

- Cross-validation: Measure generalization accuracy by testing on data unused during training
- Regularization: Penalize complex models

  E'=error on data + λ model complexity

  Akaike's information criterion (AIC), Bayesian information criterion (BIC)
- Minimum description length (MDL): Kolmogorov complexity, shortest description of data
- Structural risk minimization (SRM)

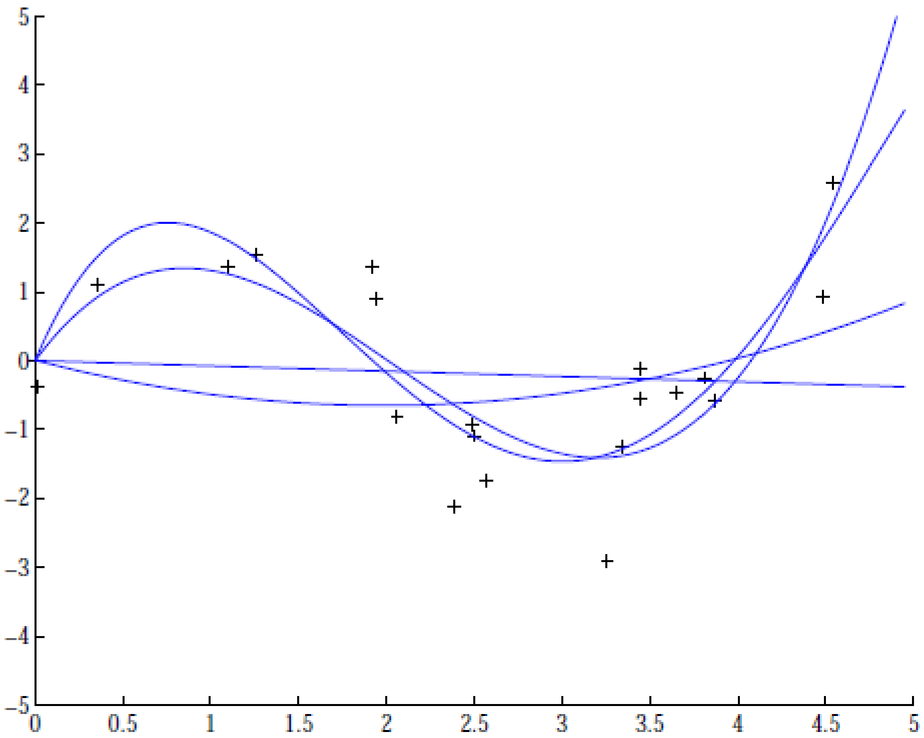
Bayesian Model Selection

- Prior on models, $p(\text{model})$

$$p(\text{model}|\text{data}) = \frac{p(\text{data}|\text{model})\,p(\text{model})}{p(\text{data})}$$

- Regularization, when prior favors simpler models
- Bayes, MAP of the posterior, $p(\text{model}|\text{data})$
- Average over a number of models with high posterior (voting, ensembles: Chapter 17)

Regression example



Coefficients increase in magnitude as order increases:
1: [-0.0769, 0.0016]
2: [0.1682, -0.6657, 0.0080]
3: [0.4238, -2.5778, 3.4675, -0.0002
4: [-0.1093, 1.4356, -5.5007, 6.0454, -0.0019]

$$regularization: E(\mathbf{w} \mid \mathcal{X}) = \frac{1}{2}\sum_{t=1}^{N}\left[r^t - g\left(x^t \mid \mathbf{w}\right)\right]^2 + \lambda \sum_i w_i^2$$

	$M=0$	$M=1$	$M=3$	$M=9$
w_0	0.19	0.82	0.31	0.35
w_1		-1.27	7.99	232.37
w_2			-25.43	-5321.83
w_3			17.37	48568.31
w_4				-231639.30
w_5				640042.26
w_6				-1061800.52
w_7				1042400.18
w_8				-557682.99
w_9				125201.43

DESIGN AND ANALYSIS OF MACHINE LEARNING EXPERIMENTS

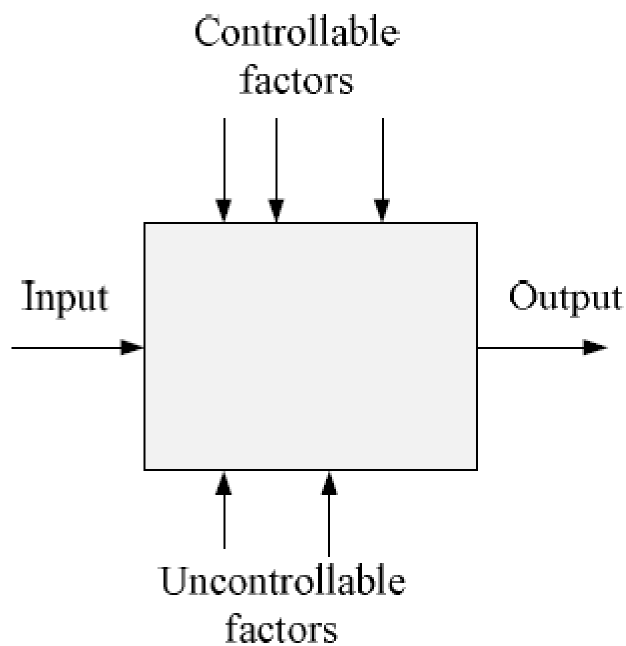
Introduction

- Questions:
  - Assessment of the expected error of a learning algorithm: Is the error rate of 1-NN less than 2%?
  - Comparing the expected errors of two algorithms: Is *k*-NN more accurate than MLP ?
- Training/validation/test sets
- Resampling methods: *K*-fold cross-validation

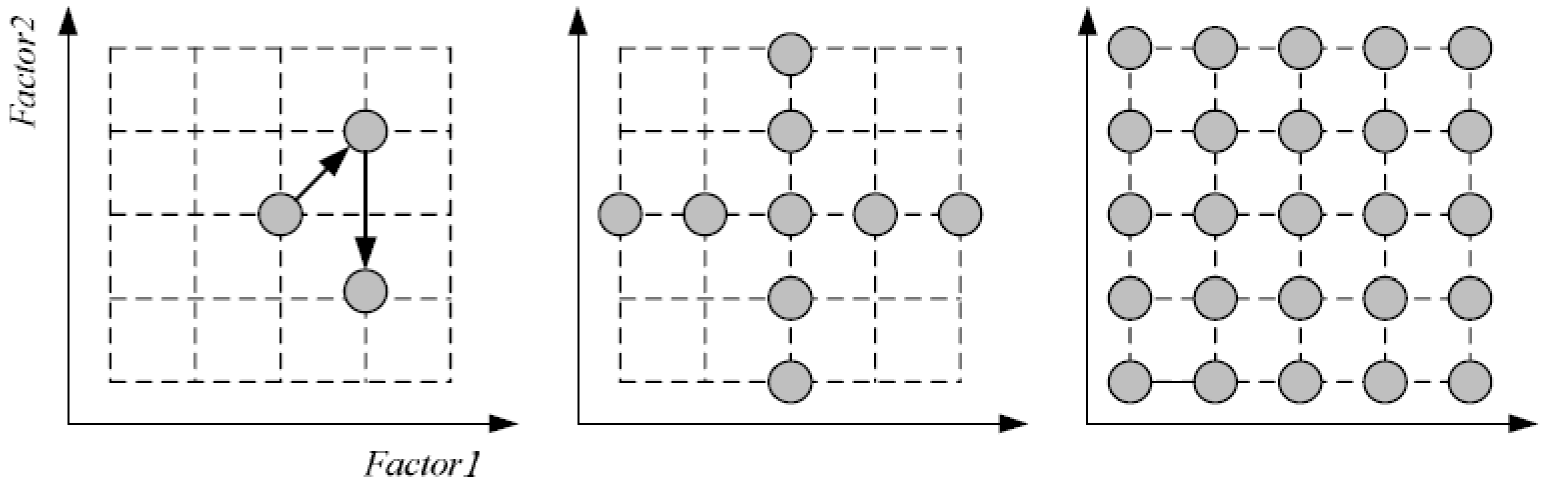
Algorithm Preference

- Criteria (Application-dependent):
  - Misclassification error, or risk (loss functions)
  - Training time/space complexity
  - Testing time/space complexity
  - Interpretability
  - Easy programmability

Factors and Response



Strategies of Experimentation



(a) Best guess (b) One factor at a time (c) Factorial design

Response surface design for approximating and maximizing the response function in terms of the controllable factors

Guidelines for ML experiments

A. Aim of the study
B. Selection of the response variable
C. Choice of factors and levels
D. Choice of experimental design
E. Performing the experiment
F. Statistical Analysis of the Data
G. Conclusions and Recommendations

Resampling and K-Fold Cross-Validation

- The need for multiple training/validation sets
  $\{X_i,V_i\}_i$: Training/validation sets of fold i
- K-fold cross-validation: Divide X into k, X_i,i=1,...,K

$$\begin{aligned}
&\mathcal{V}_1 = \mathcal{X}_1 \quad \mathcal{T}_1 = \mathcal{X}_2 \cup \mathcal{X}_3 \cup \cdots \cup \mathcal{X}_K \\
&\mathcal{V}_2 = \mathcal{X}_2 \quad \mathcal{T}_2 = \mathcal{X}_1 \cup \mathcal{X}_3 \cup \cdots \cup \mathcal{X}_K \\
&\quad\vdots \\
&\mathcal{V}_K = \mathcal{X}_K \quad \mathcal{T}_K = \mathcal{X}_1 \cup \mathcal{X}_2 \cup \cdots \cup \mathcal{X}_{K-1}
\end{aligned}$$

- T_i share K-2 parts

Bootstrapping

- Draw instances from a dataset *with replacement*
- Prob that we do not pick an instance after N draws

$$\left(1-\frac{1}{N}\right)^N \approx e^{-1} = 0.368$$

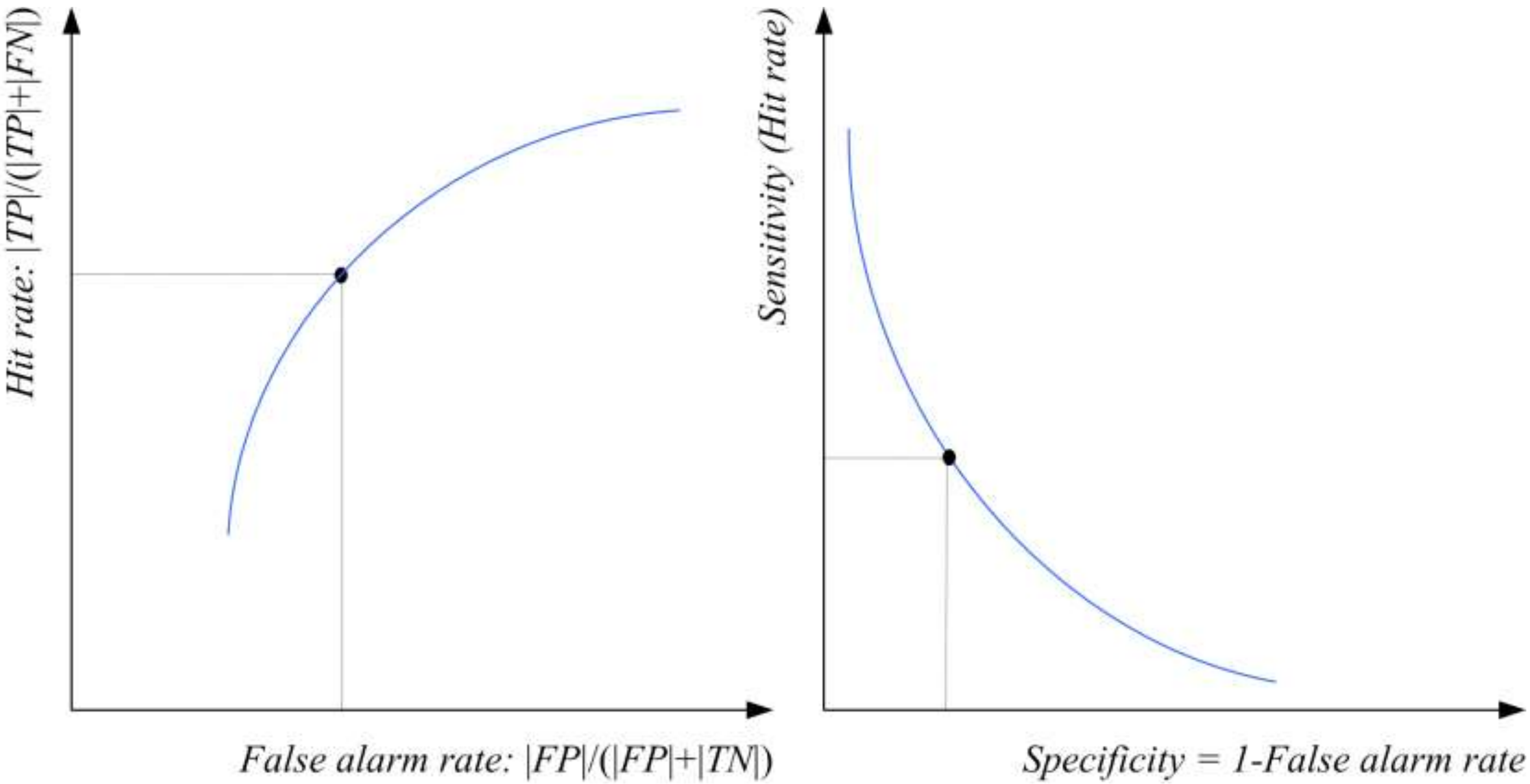
that is, only 36.8% is new!

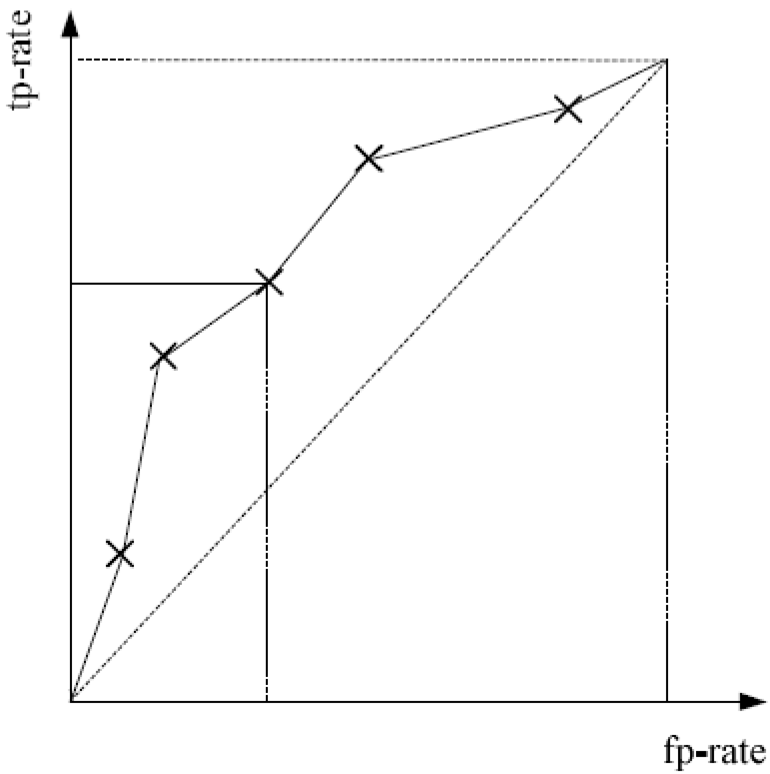
Measuring Error

	Predicted class	
True Class	Yes	No
Yes	TP: True Positive	FN: False Negative
No	FP: False Positive	TN: True Negative

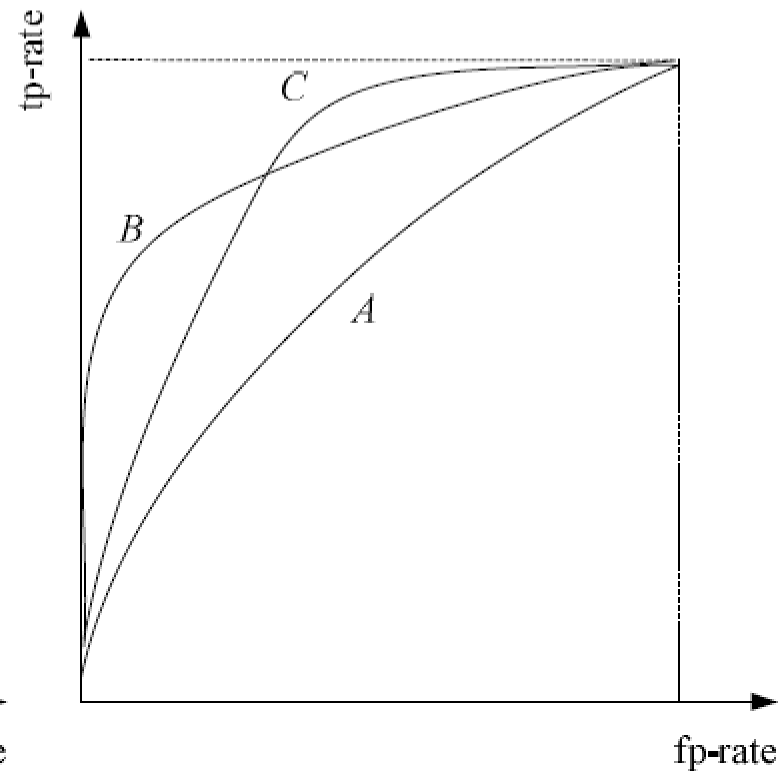
- Error rate = # of errors / # of instances = (FN+FP) / N
- Recall = # of found positives / # of positives
  = TP / (TP+FN) = sensitivity = hit rate
- Precision = # of found positives / # of found
  = TP / (TP+FP)
- Specificity = TN / (TN+FP)
- False alarm rate = FP / (FP+TN) = 1 - Specificity

ROC Curve


Hit rate: |TP|/(|TP|+|FN|)
False alarm rate: |FP|/(|FP|+|TN|)
Sensitivity (Hit rate)
Specificity = 1-False alarm rate



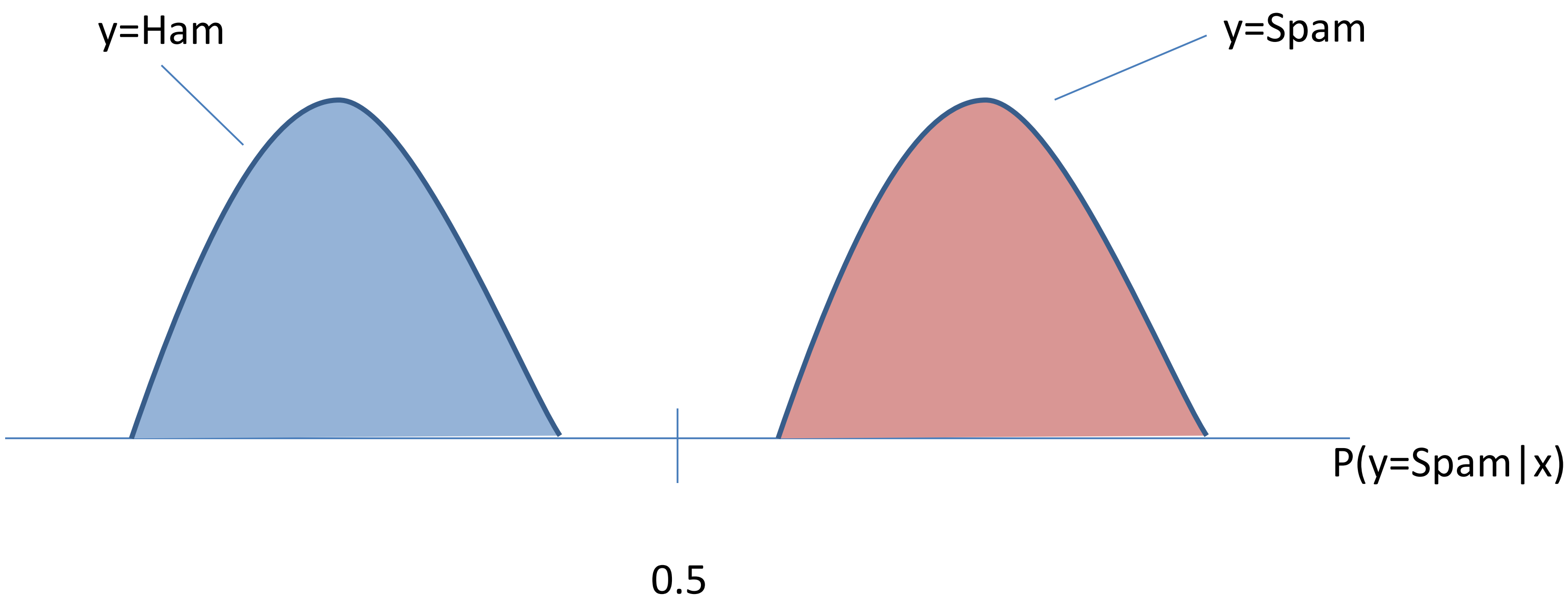
(a) Example ROC curve



(b) Different ROC curves for different classifiers

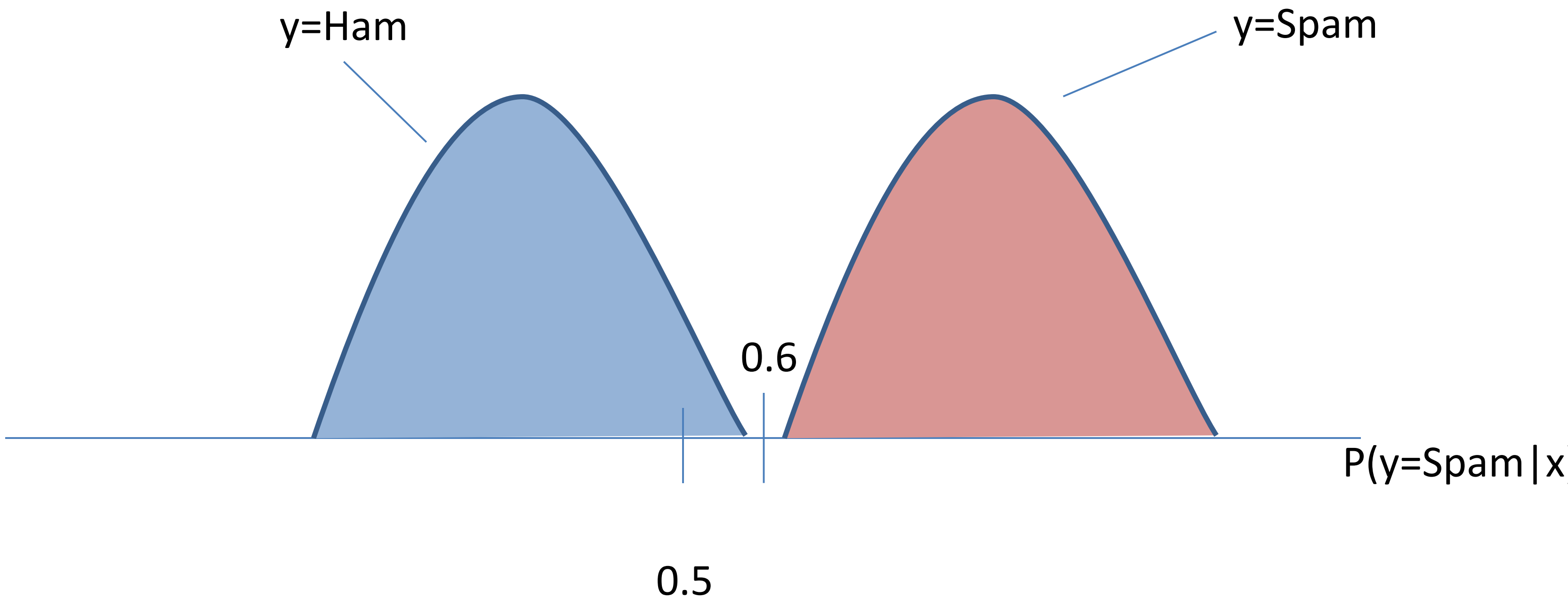
An example

- Consider class conditionals of output probabilities for training samples



An example

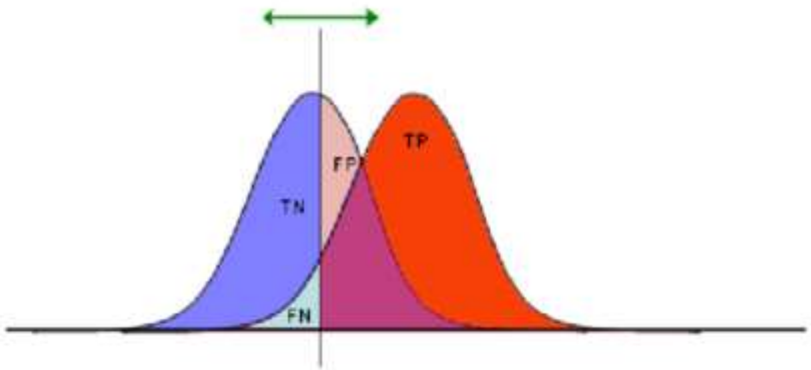
- Consider class conditionals of output probabilities for training samples



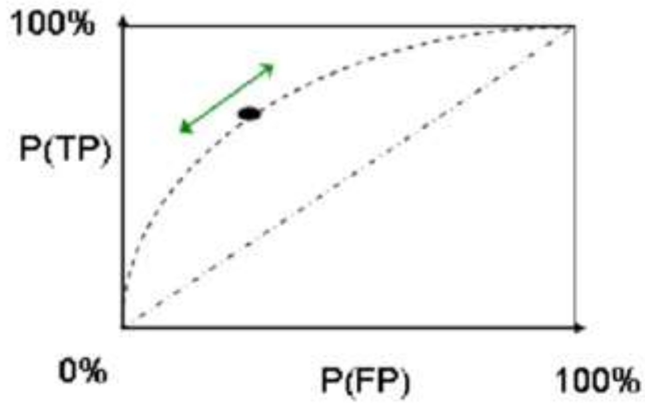
Binary classification

- Predicting one of two categories
  - Spam/Ham
  - Dead/Alive
  - Click on/Don’t click
- Hypothesis often outputs a number
  - Probability of the positive class
  - A real number to show confidence
- The **cutoff** that we choose gives us different results

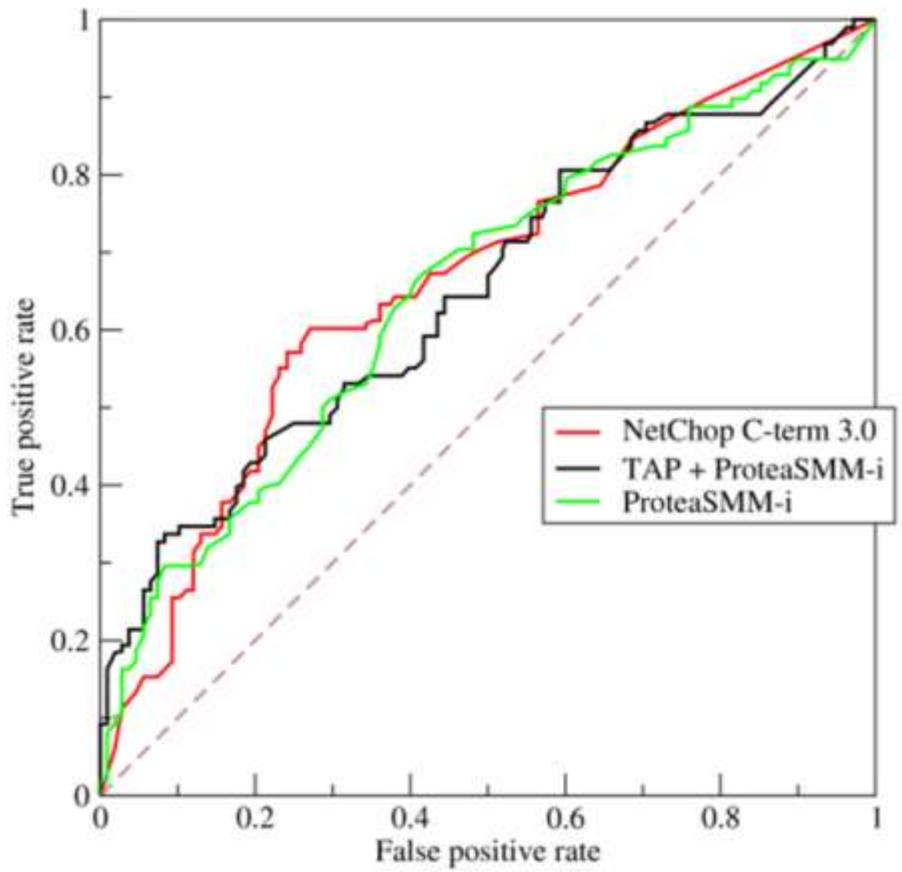
ROC Curves



TP	FP
FN	TN
1	1

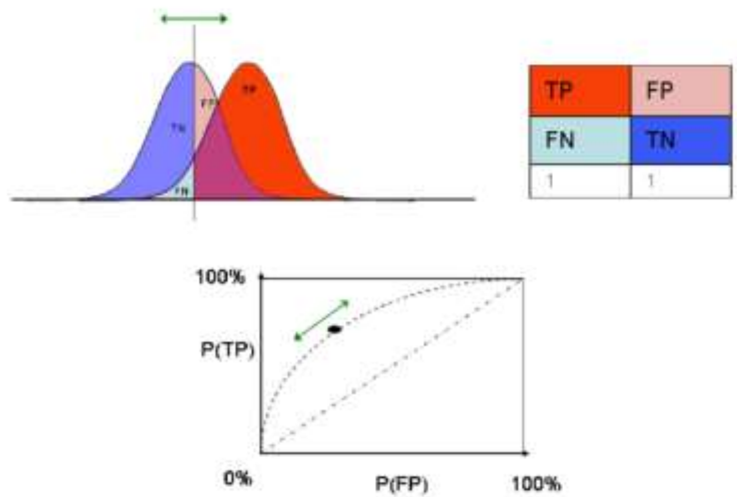


Example ROC



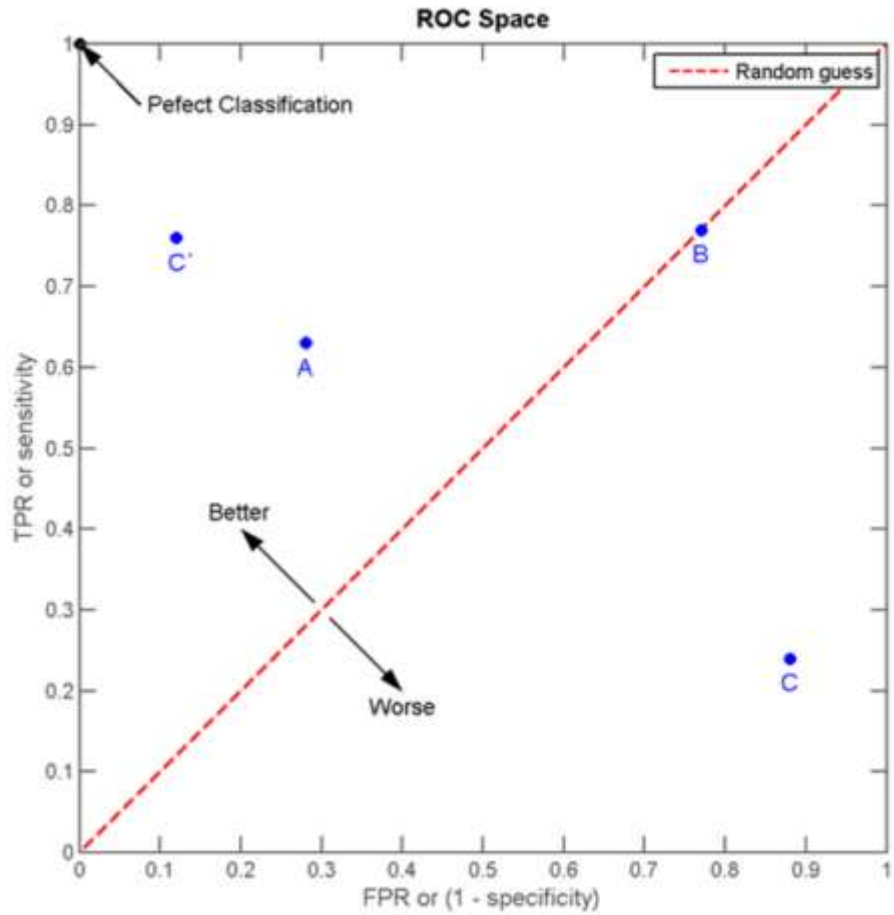
1
0.8
0.6
0.4
0.2
0
0
0.2
0.4
0.6
0.8
1
True positive rate
False positive rate
NetChop C-term 3.0
TAP + ProteaSMM-i
ProteaSMM-i

Area under the curve



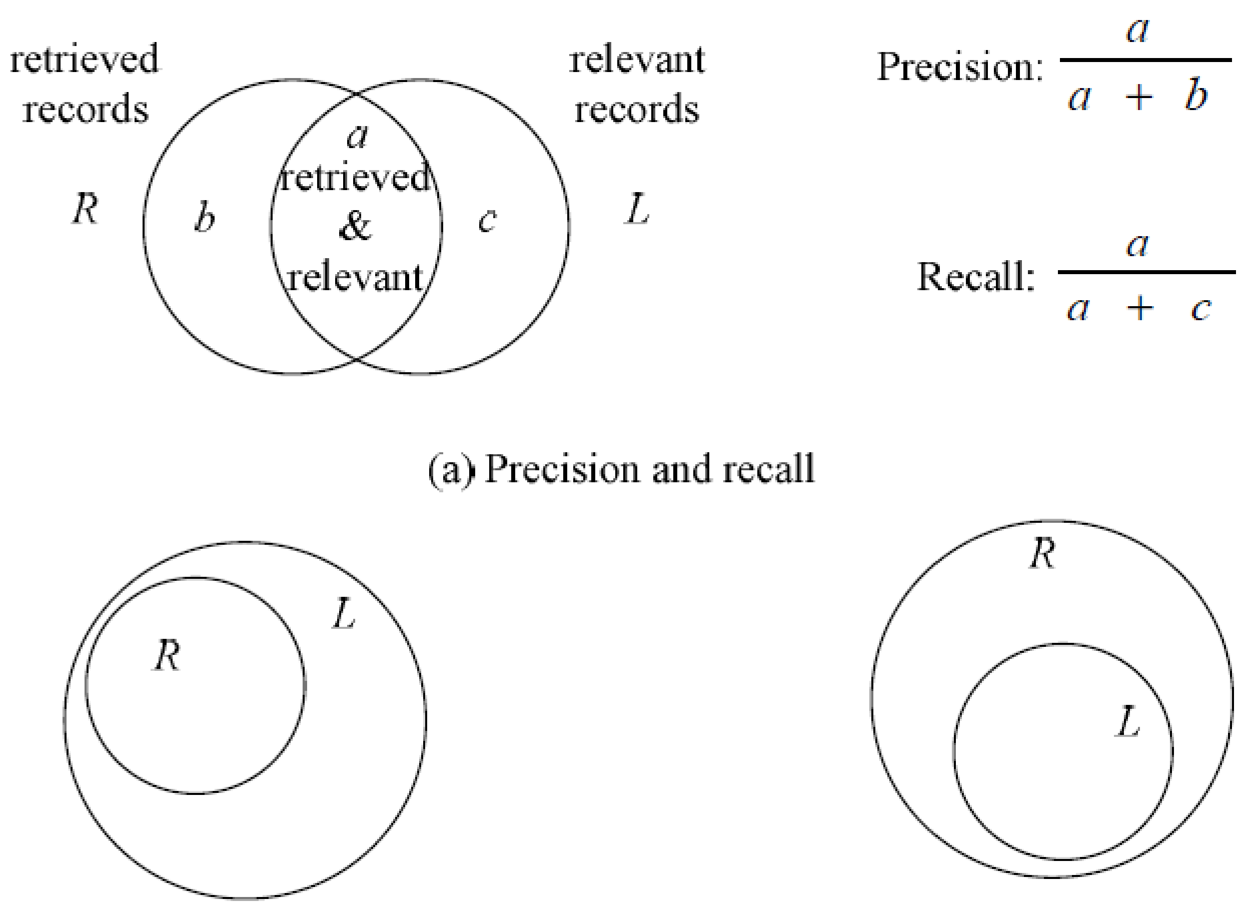
- AUC = 0.5 random guessing
- AUC = 1 perfect classifier
- In general AUC of above 0.8 considered "good"

What is good?



ROC Space
Random guess
Pefect Classification
C'
B
A
Better
Worse
C
TPR or sensitivity
FPR or (1 - specificity)
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

Precision and Recall



(a) Precision and recall

(b) Precision = 1

(c) Recall = 1