Artificial Intelligence and Machine Learning

Barbara Caputo

Bayes Decision Rule

x is an observation for which:

Bayes Decision Rule

x is an observation for which:

if $P(\omega_1 \mid x) > P(\omega_2 \mid x) \Rightarrow$ True state of nature $= \omega_1$

if $P(\omega_1 \mid x) < P(\omega_2 \mid x) \Rightarrow$ True state of nature $= \omega_2$

Bayes Decision Rule

x is an observation for which:

if $P(\omega_1 \mid x) > P(\omega_2 \mid x)$ $\Rightarrow$ True state of nature = ω_1
if $P(\omega_1 \mid x) < P(\omega_2 \mid x)$ $\Rightarrow$ True state of nature = ω_2

Therefore:

whenever we observe a particular x, the probability of error is :

Bayes Decision Rule

x is an observation for which:

if $P(\omega_1 \mid x) > P(\omega_2 \mid x) \Rightarrow$ True state of nature = ω_1

if $P(\omega_1 \mid x) < P(\omega_2 \mid x) \Rightarrow$ True state of nature = ω_2

Therefore:

whenever we observe a particular x, the probability of error is :

$$P(error \mid x) = P(\omega_1 \mid x) \text{ if we decide } \omega_2$$

$$P(error \mid x) = P(\omega_2 \mid x) \text{ if we decide } \omega_1$$

Bayes Decision Rule minimizes probability of error

Bayes Decision Rule minimizes probability of error

Decide ω_1 *if* $P(\omega_1 \mid x) > P(\omega_2 \mid x)$;
otherwise decide ω_2

Bayes Decision Rule minimizes probability of error

Decide ω_1 *if* $P(\omega_1 \mid x) > P(\omega_2 \mid x)$;
otherwise decide ω_2

Therefore:

$$P(error \mid x) = min\ [\ P(\omega_1 \mid x),\ \ P(\omega_2 \mid x)\ ]$$

(Bayes decision)

Bayes Decision Theory – Continuous Features

- Generalization of the preceding ideas
  - Use of more than one feature
  - Use more than two states of nature
  - Allowing actions and not only decide on the state of nature
  - Introduce a loss of function which is more general than the probability of error

Loss Function

- Allowing actions other than classification primarily allows the possibility of rejection
- Refusing to make a decision in close or bad cases!
- The loss function states how costly each action taken is

Loss Function Definition

Loss Function Definition

Let $\{\omega_1, \omega_2, \ldots, \omega_c\}$ be the set of c states of nature (or "categories")

Loss Function Definition

Let $\{\omega_1, \omega_2, \ldots, \omega_c\}$ be the set of c states of nature (or "categories")

Let $\{\alpha_1, \alpha_2, \ldots, \alpha_a\}$ be the set of possible actions

Loss Function Definition

Let $\{\omega_1, \omega_2, \ldots, \omega_c\}$ be the set of c states of nature (or "categories")

Let $\{\alpha_1, \alpha_2, \ldots, \alpha_a\}$ be the set of possible actions

Let $\lambda(\alpha_i \mid \omega_j)$ be the loss incurred for taking

action α_i when the state of nature is ω_j

Overall Risk

Overall Risk

R = Sum of all $R(\alpha_i \mid x)$ *for* $i = 1,\ldots,a$

Overall Risk

$$R = \text{Sum of all } \underbrace{R(\alpha_i \mid x)}_{\textbf{Conditional risk}} \text{ for } i = 1,\ldots,a$$

Minimizing R $\Longleftrightarrow$ Minimizing $R(\alpha_i \mid x)$ for $i = 1,\ldots, a$

Overall Risk

R = Sum of all $\underbrace{R(\alpha_i \mid x)}_{\text{Conditional risk}}$ *for i = 1,…,a*

Minimizing R ⟺ Minimizing $R(\alpha_i \mid x)$ *for i = 1,…, a*

Expected Loss with action i

$$R(\alpha_i \mid x) = \sum_{j=1}^{j=c} \lambda(\alpha_i \mid \omega_j) P(\omega_j \mid x)$$

Overall Risk

R = Sum of all $\underbrace{R(\alpha_i \mid x)}_{\text{Conditional risk}}$ for $i = 1,\ldots,a$

Minimizing R $\Leftrightarrow$ Minimizing $R(\alpha_i \mid x)$ for $i = 1,\ldots, a$

Expected Loss with action i

$$R(\alpha_i \mid x) = \sum_{j=1}^{j=c} \lambda(\alpha_i \mid \omega_j) P(\omega_j \mid x)$$

Select the action α_i for which $R(\alpha_i \mid x)$ is minimum

R is minimum and R in this case is called the Risk

Bayes risk = best performance that can be achieved

Two-category classification

Two-category classification

α_1 : *deciding* ω_1

α_2 : *deciding* ω_2

$$\lambda_{ij} = \lambda(\alpha_i \mid \omega_j)$$

Two-category classification

α_1 : deciding ω_1

α_2 : deciding ω_2

$$\lambda_{ij} = \lambda(\alpha_i \mid \omega_j)$$

loss incurred for deciding ω_i when the true state of nature is ω_j

Two-category classification

α_1 : deciding ω_1

α_2 : deciding ω_2

$$\lambda_{ij} = \lambda(\alpha_i \mid \omega_j)$$

loss incurred for deciding ω_i when the true state of nature is ω_j

Conditional risk:

$$R(\alpha_1 \mid x) = \lambda_{11}P(\omega_1 \mid x) + \lambda_{12}P(\omega_2 \mid x)$$
$$R(\alpha_2 \mid x) = \lambda_{21}P(\omega_1 \mid x) + \lambda_{22}P(\omega_2 \mid x)$$

Minimum Risk Decision Rule

Minimum Risk Decision Rule

Our rule is the following:

if $R(\alpha_1 \mid x) < R(\alpha_2 \mid x)$

action α_1: "decide ω_1" is taken

Minimum Risk Decision Rule

Our rule is the following:

if $R(\alpha_1 \mid x) < R(\alpha_2 \mid x)$

action α_1: "decide ω_1" is taken

This results in the equivalent rule :

decide ω_1 if:

Minimum Risk Decision Rule

Our rule is the following:

if $R(\alpha_1 \mid x) < R(\alpha_2 \mid x)$

action α_1: "decide ω_1" is taken

This results in the equivalent rule :

decide ω_1 if:

$$(\lambda_{21} - \lambda_{11})\, P(x \mid \omega_1)\, P(\omega_1) > (\lambda_{12} - \lambda_{22})\, P(x \mid \omega_2)\, P(\omega_2)$$

and decide ω_2 otherwise

Minimum-Error-Rate Classification

- Actions are decisions on classes

  If action α_i is taken and the true state of nature is ω_j then: decision is correct if $i = j$ and in error if $i \neq j$

  Seek a decision rule that minimizes the *probability of error* which is the *error rate*

Let’s take a break!

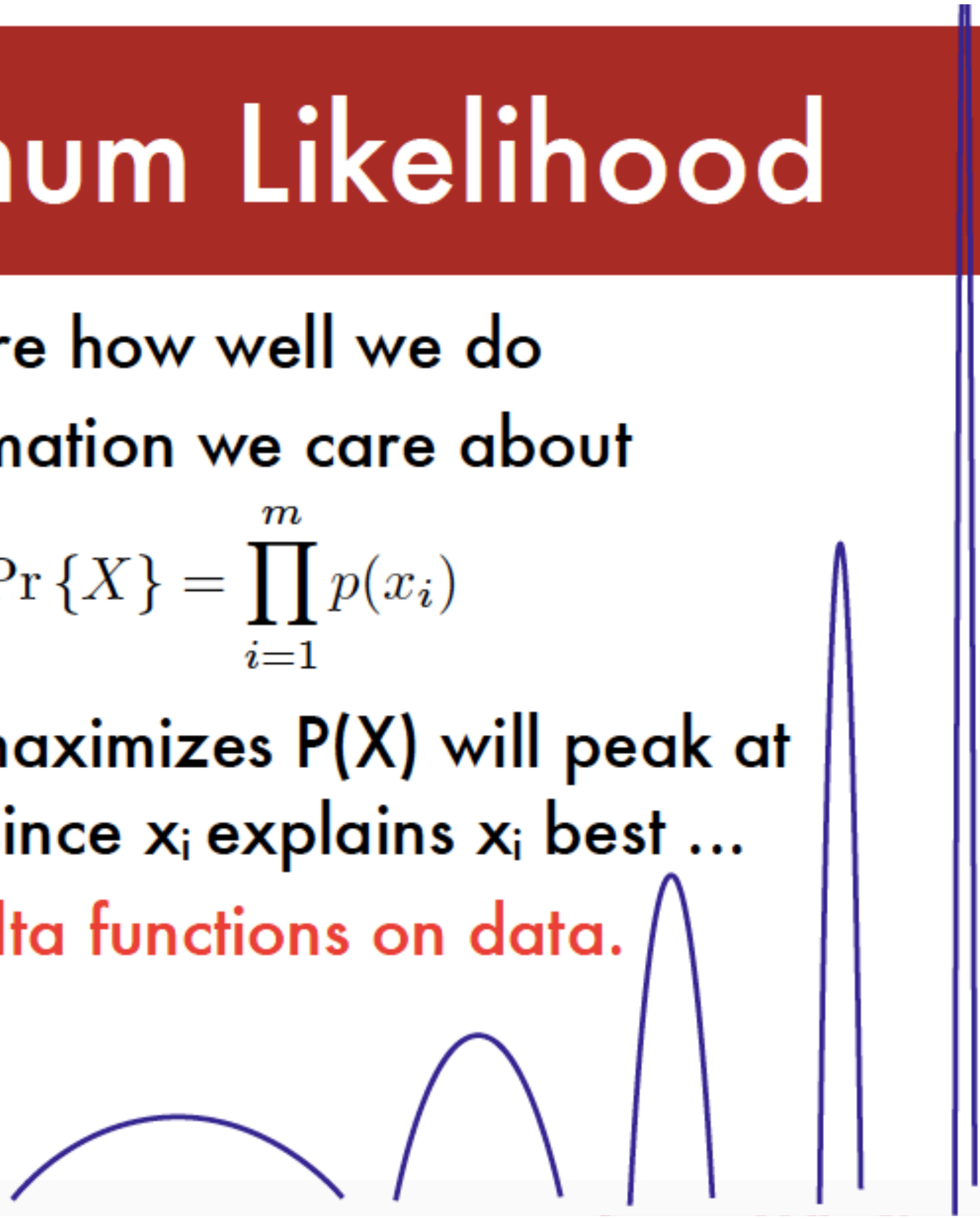
Model Selection

Maximum Likelihood

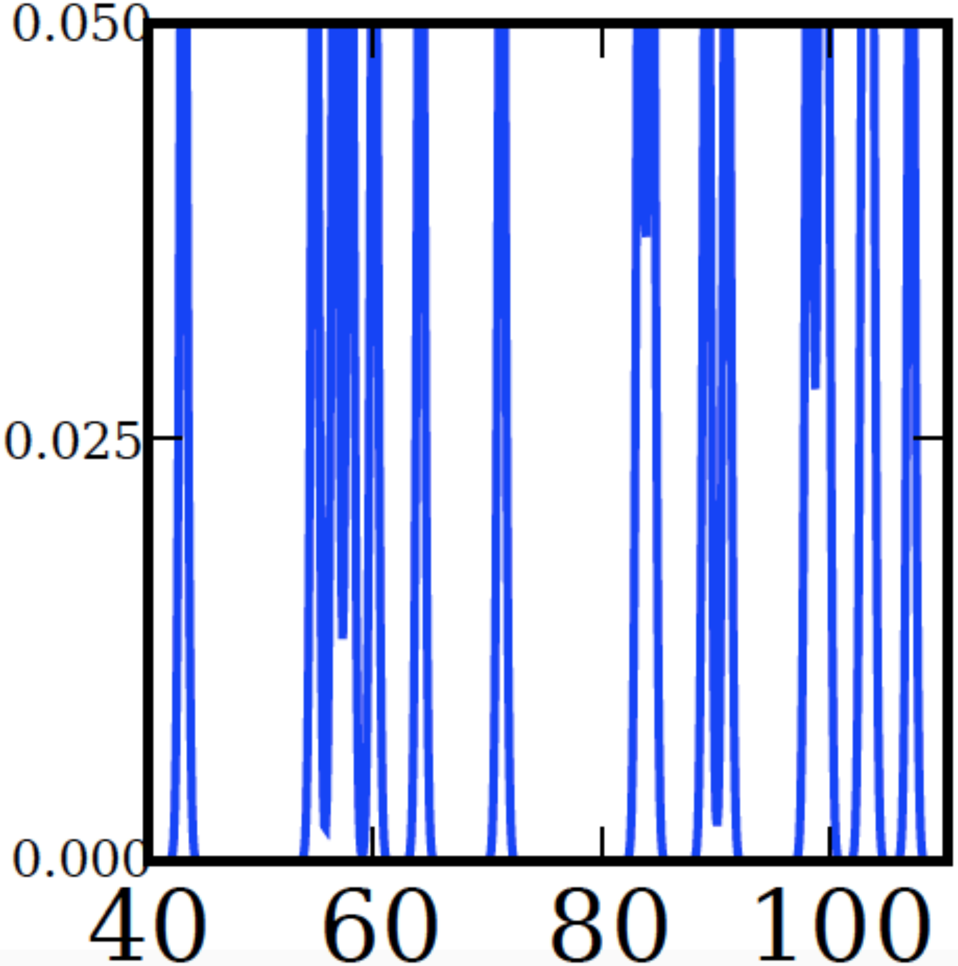
- Need to measure how well we do
- For density estimation we care about

$$\Pr\{X\} = \prod_{i=1}^{m} p(x_i)$$

- Finding a that maximizes P(X) will peak at all data points since x_i explains x_i best ...
- Maxima are delta functions on data.
- Overfitting!

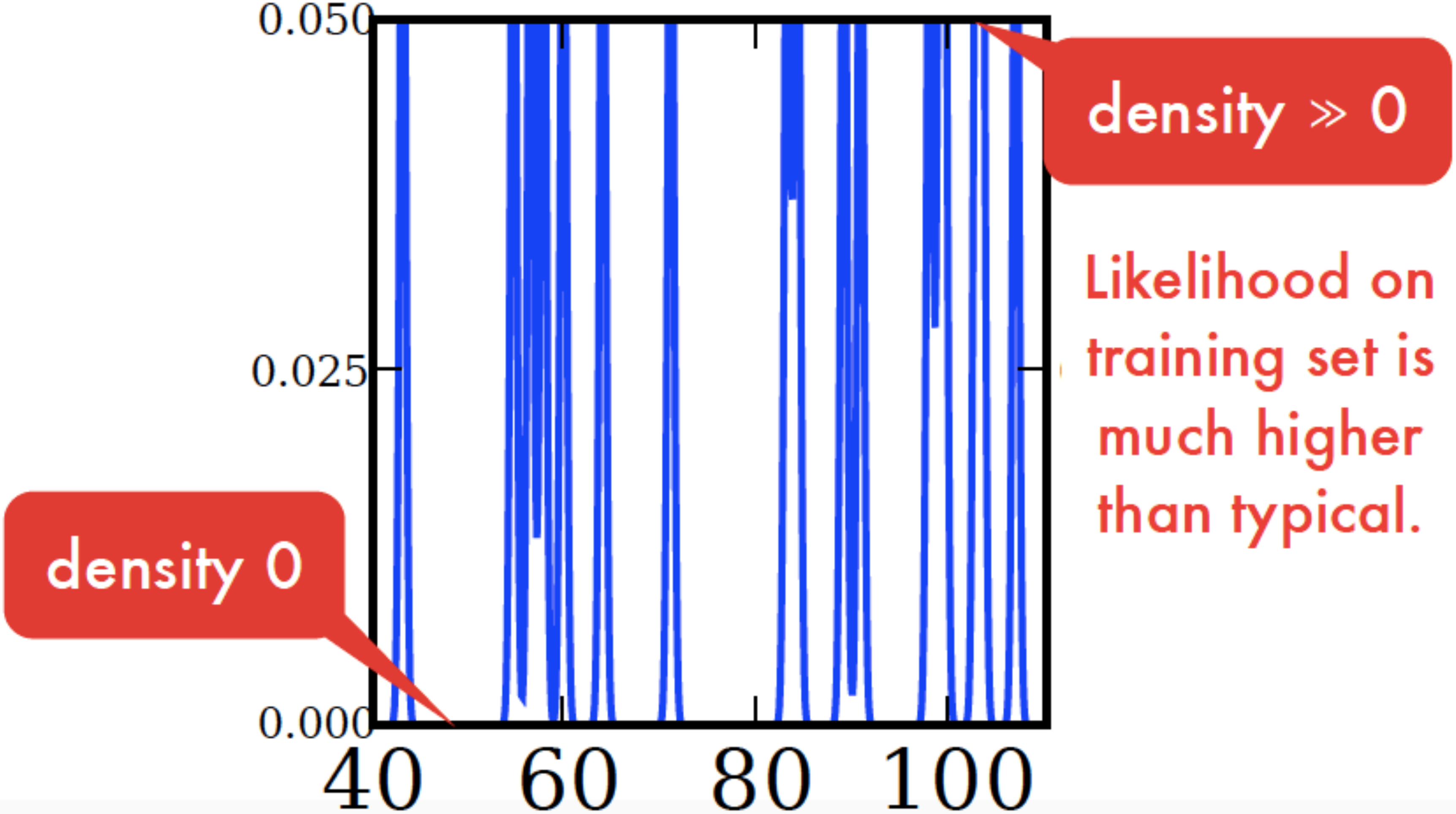


Overfitting



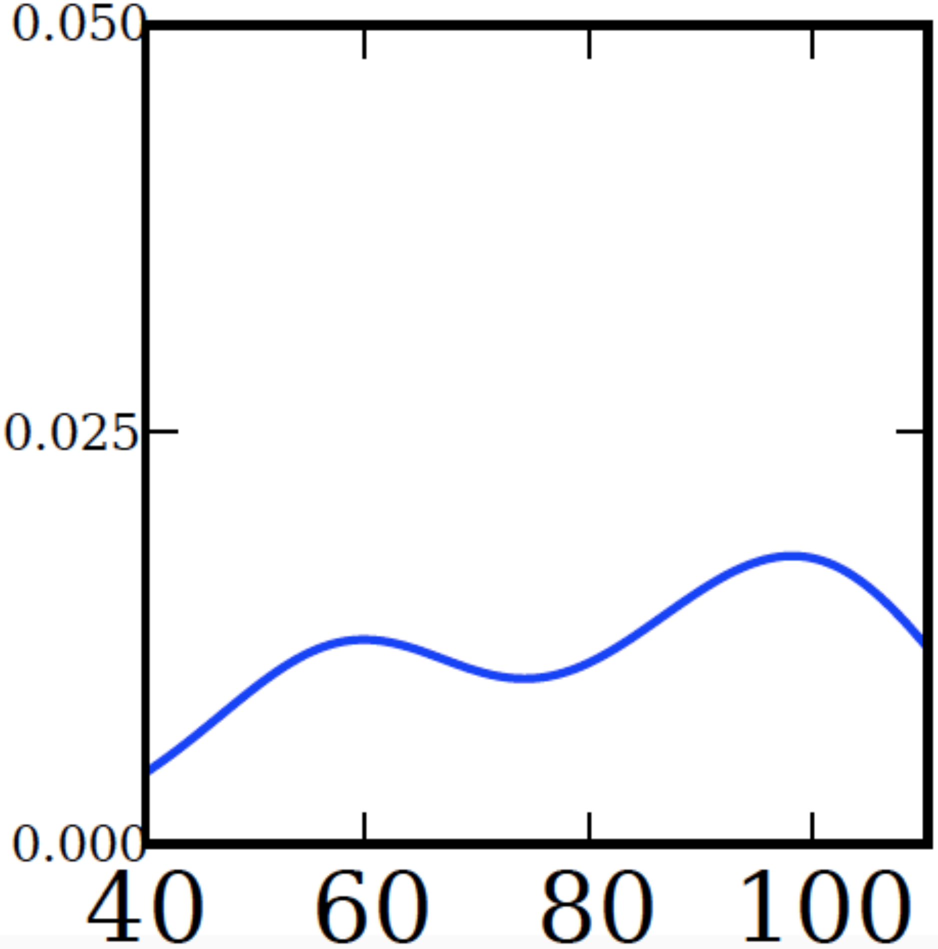
Likelihood on training set is much higher than typical.

Overfitting



Likelihood on training set is much higher than typical.

Underfitting



Likelihood on training set is very similar to typical one.

Too simple.

Model Selection

- Validation
  - Use some of the data to estimate density.
  - Use other part to evaluate how well it works
  - Pick the parameter that works best

Model Selection

- Validation
  - Use some of the data to estimate density.
  - Use other part to evaluate how well it works
  - Pick the parameter that works best

$$\mathcal{L}(X'|X) := \frac{1}{n'} \sum_{i=1}^{n'} \log \hat{p}(x_i')$$

Model Selection

- Leave-one-out Crossvalidation

Model Selection

- Leave-one-out Crossvalidation
  - Use almost all data to estimate density.

Model Selection

- Leave-one-out Crossvalidation
  - Use almost all data to estimate density.
  - Use single instance to estimate how well it works

Model Selection

- Leave-one-out Crossvalidation
  - Use almost all data to estimate density.
  - Use single instance to estimate how well it works

$$\log p(x_i | X \backslash x_i) = \log \frac{1}{n-1} \sum_{j \neq i} k(x_i, x_j)$$

Model Selection

- Leave-one-out Crossvalidation
  - Use almost all data to estimate density.
  - Use single instance to estimate how well it works

$$\log p(x_i|X\backslash x_i) = \log \frac{1}{n-1} \sum_{j \neq i} k(x_i, x_j)$$

  - This has huge variance

Model Selection

- Leave-one-out Crossvalidation
  - Use almost all data to estimate density.
  - Use single instance to estimate how well it works
$$\log p(x_i|X \backslash x_i) = \log \frac{1}{n-1} \sum_{j \neq i} k(x_i, x_j)$$
  - This has huge variance
  - Average over estimates for all training data

Model Selection

- Leave-one-out Crossvalidation
  - Use almost all data to estimate density.
  - Use single instance to estimate how well it works

$$\log p(x_i|X \backslash x_i) = \log \frac{1}{n-1} \sum_{j \neq i} k(x_i, x_j)$$

  - This has huge variance
  - Average over estimates for all training data
  - Pick the parameter that works best

Model Selection

- Leave-one-out Crossvalidation
  - Use almost all data to estimate density.
  - Use single instance to estimate how well it works
$$\log p(x_i|X\backslash x_i) = \log \frac{1}{n-1} \sum_{j \neq i} k(x_i, x_j)$$
  - This has huge variance
  - Average over estimates for all training data
  - Pick the parameter that works best
- Simple implementation
$$\frac{1}{n} \sum_{i=1}^{n} \log \left[\frac{n}{n-1} p(x_i) - \frac{1}{n-1} k(x_i, x_i) \right] \text{ where } p(x) = \frac{1}{n} \sum_{i=1}^{n} k(x_i, x)$$

Model Selection

- k-fold Crossvalidation

Model Selection

- k-fold Crossvalidation
  - Partition data into k blocks (typically 10)

Model Selection

- k-fold Crossvalidation
  - Partition data into k blocks (typically 10)
  - Use all but one block to compute estimate

Model Selection

- k-fold Crossvalidation
  - Partition data into k blocks (typically 10)
  - Use all but one block to compute estimate
  - Use remaining block as validation set

Model Selection

- k-fold Crossvalidation
  - Partition data into k blocks (typically 10)
  - Use all but one block to compute estimate
  - Use remaining block as validation set
  - Average over all validation estimates

Model Selection

- k-fold Crossvalidation
  - Partition data into k blocks (typically 10)
  - Use all but one block to compute estimate
  - Use remaining block as validation set
  - Average over all validation estimates

$$\frac{1}{k}\sum_{i=1}^{k} l(p(X_i|X\backslash X_i))$$

Model Selection

- k-fold Crossvalidation
  - Partition data into k blocks (typically 10)
  - Use all but one block to compute estimate
  - Use remaining block as validation set
  - Average over all validation estimates

$$\frac{1}{k}\sum_{i=1}^{k} l(p(X_i|X\backslash X_i))$$

- Almost unbiased (e.g. via Luntz and Brailovski, 1969) (error is for (k-1)/k sized set)

Model Selection

- k-fold Crossvalidation
    - Partition data into k blocks (typically 10)
    - Use all but one block to compute estimate
    - Use remaining block as validation set
    - Average over all validation estimates

$$\frac{1}{k}\sum_{i=1}^{k} l(p(X_i|X\backslash X_i))$$

- Almost unbiased (e.g. via Luntz and Brailovski, 1969) (error is for (k-1)/k sized set)
- Pick best parameter

That’s all