Introduction to Artificial Intelligence

A Basic Introduction to Learning

Andres Mendez-Vazquez

March 8, 2019

Outline

Outline

Statistical Learning

Clearly, there are many problems important for us

- Predict whether a patient, hospitalized due to a heart attack, will have a second heart attack,
- Predict the price of a stock in 6 months from now,
- Given a market population what products to recommend to them,
- How to recognize in a video a car or person,
- How to predict maintenance in a factory,
- etc.

Statistical Learning

Clearly, there are many problems important for us

- Predict whether a patient, hospitalized due to a heart attack, will have a second heart attack,
- Predict the price of a stock in 6 months from now,
- Given a market population what products to recommend to them,
- How to recognize in a video a car or person,
- How to predict maintenance in a factory,
- etc.

Statistical Learning

Clearly, there are many problems important for us

- Predict whether a patient, hospitalized due to a heart attack, will have a second heart attack,
- Predict the price of a stock in 6 months from now,
- Given a market population what products to recommend to them,
- How to recognize in a video a car or person,
- How to predict maintenance in a factory,
- etc.

Statistical Learning

Clearly, there are many problems important for us

- Predict whether a patient, hospitalized due to a heart attack, will have a second heart attack,
- Predict the price of a stock in 6 months from now,
- Given a market population what products to recommend to them,
- How to recognize in a video a car or person,
- How to predict maintenance in a factory,
- etc.

Statistical Learning

Clearly, there are many problems important for us

- Predict whether a patient, hospitalized due to a heart attack, will have a second heart attack,
- Predict the price of a stock in 6 months from now,
- Given a market population what products to recommend to them,
- How to recognize in a video a car or person,
- How to predict maintenance in a factory,
- etc.

Statistical Learning

Clearly, there are many problems important for us

- Predict whether a patient, hospitalized due to a heart attack, will have a second heart attack,
- Predict the price of a stock in 6 months from now,
- Given a market population what products to recommend to them,
- How to recognize in a video a car or person,
- How to predict maintenance in a factory,
- etc.

Example

Given a sample on frequency of the most common words in a series of 4601 emails

	george	you	your	hp	free	hpl	!	our	re	edu
Spam	0.00	2.26	1.38	0.002	0.52	0.01	0.51	0.51	0.13	0.01
email	1.27	1.27	0.44	0.90	0.07	0.43	0.11	0.18	0.42	0.29

Example

Given a sample on frequency of the most common words in a series of 4601 emails

	george	you	your	hp	free	hpl	!	our	re	edu
Spam	0.00	2.26	1.38	0.002	0.52	0.01	0.51	0.51	0.13	0.01
email	1.27	1.27	0.44	0.90	0.07	0.43	0.11	0.18	0.42	0.29

We want to design a series of rules to guess when you have a Spam or a genuine email

$$f_1\left(message\right) = \begin{cases} \%george < 0.6 \text{ and } \%you > 1.5 & spam \\ \text{Otherwhise} & email \end{cases}$$

$$f_2\left(message\right) = \begin{cases} 0.2 \times \%you - 0.3 \times \%george > 1.5 & spam \\ \text{Otherwhise} & email \end{cases}$$

Example

Given a sample on frequency of the most common words in a series of 4601 emails

	george	you	your	hp	free	hpl	!	our	re	edu
Spam	0.00	2.26	1.38	0.002	0.52	0.01	0.51	0.51	0.13	0.01
email	1.27	1.27	0.44	0.90	0.07	0.43	0.11	0.18	0.42	0.29

We want to design a series of rules to guess when you have a Spam or a genuine email

$$f_1\left(message\right) = \begin{cases} \%george < 0.6 \text{ and } \%you > 1.5 & spam \\ \text{Otherwhise} & email \end{cases}$$

$$f_2\left(message\right) = \begin{cases} 0.2 \times \%you - 0.3 \times \%george > 1.5 & spam \\ \text{Otherwhise} & email \end{cases}$$

Outline

Therefore

Let $X \in \mathbb{R}^d$ a real valued random input and $Y \in \mathbb{R}$ a real valued output

With joint distribution $P(X, Y)$

We are looking for a function that takes the variables in X to map them into Y

$f(X)$ predicting Y

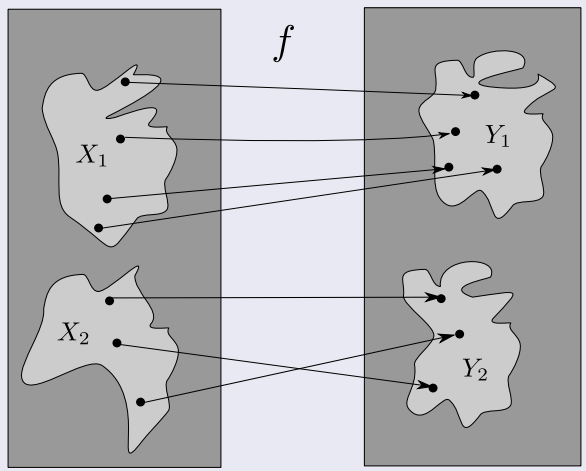
Therefore

Let $X \in \mathbb{R}^d$ a real valued random input and $Y \in \mathbb{R}$ a real valued output

With joint distribution $P(X, Y)$

We are looking for a function that takes the variables in X to map them into Y

$f(X)$ predicting Y



Outline

We have two main types

Quantitative Data

- They are measures of values or counts and are expressed as numbers.
  - Quantitative data are data about numeric variables (e.g. how many; how much; or how often).

We have two main types

Quantitative Data

- They are measures of values or counts and are expressed as numbers.
  - Quantitative data are data about numeric variables (e.g. how many; how much; or how often).

Qualitative Data

- They are measures of 'types' and may be represented by a name, symbol, or a number code.
  - Qualitative data are data about categorical variables (e.g. what type).

We have two main types

Quantitative Data

- They are measures of values or counts and are expressed as numbers.
  - Quantitative data are data about numeric variables (e.g. how many; how much; or how often).

Qualitative Data

- They are measures of 'types' and may be represented by a name, symbol, or a number code.
  - Qualitative data are data about categorical variables (e.g. what type).

We have two main types

Quantitative Data

- They are measures of values or counts and are expressed as numbers.
  - Quantitative data are data about numeric variables (e.g. how many; how much; or how often).

Qualitative Data

- They are measures of 'types' and may be represented by a name, symbol, or a number code.
  - Qualitative data are data about categorical variables (e.g. what type).

For Example (In the case of Outputs)

If we are classifying digits

The Outputs are Quantitative

$$\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$$

For Example (In the case of Outputs)

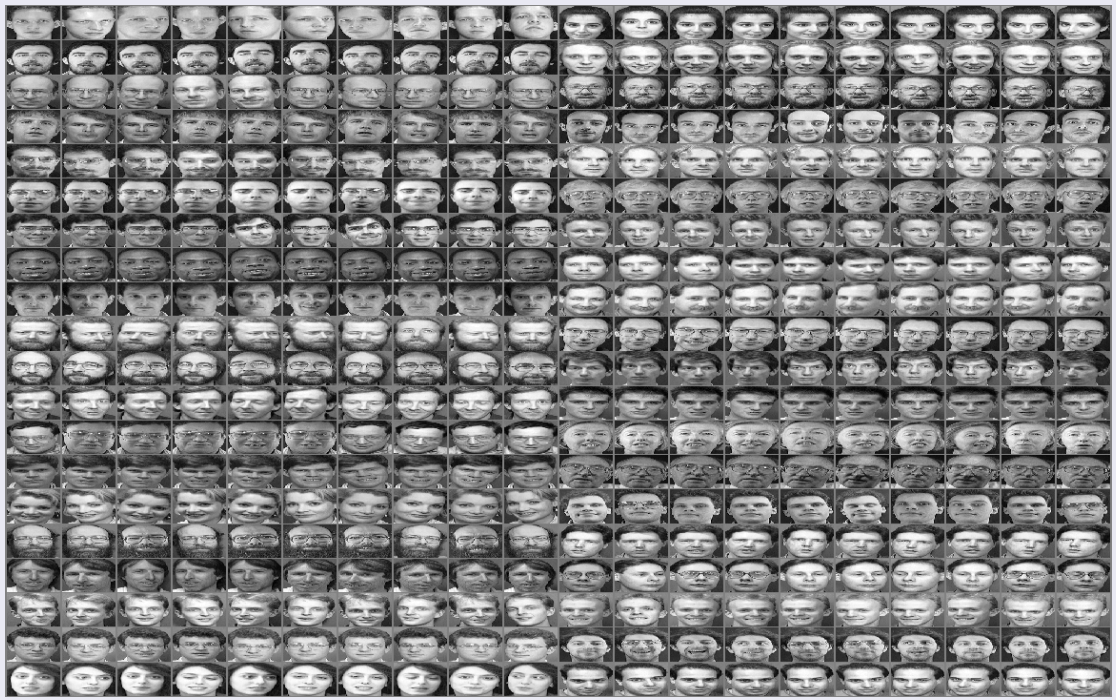
If we are classifying digits

The Outputs are Quantitative

$$\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$$

Therefore

We want to use the Quantitative or Qualitative variables

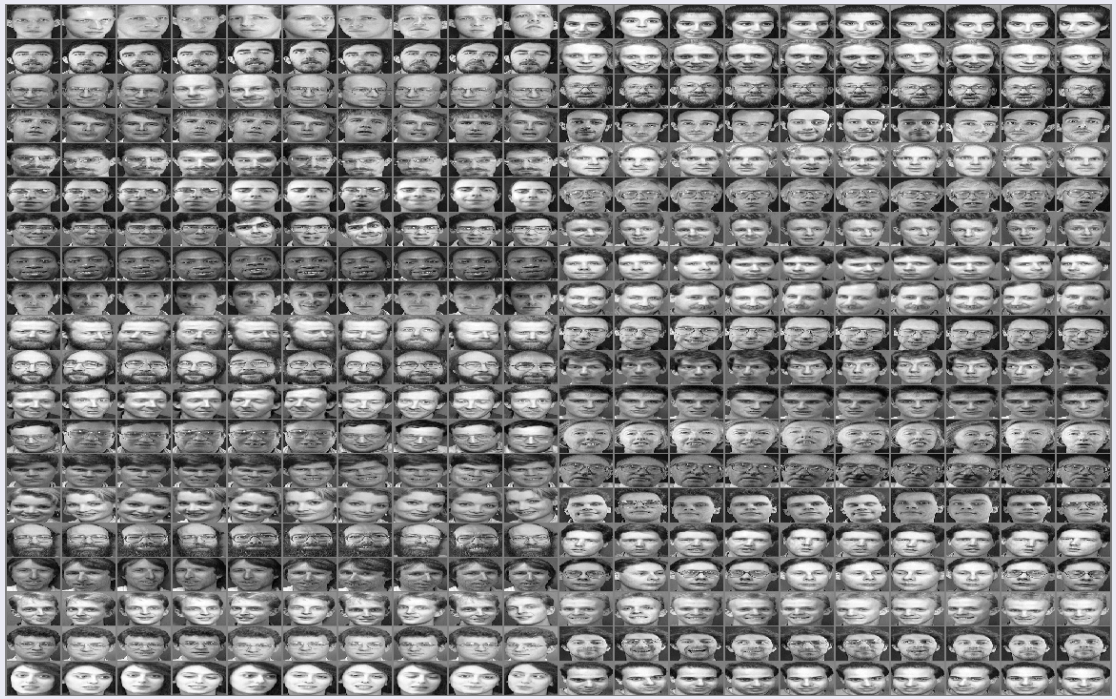


To obtain the correct sought output

{Andres,Fabiola} = People that can drive a certain car

Therefore

We want to use the Quantitative or Qualitative variables



To obtain the correct sought output

$$\{\text{Andres,Fabiola}\} = \text{People that can drive a certain car}$$

Outline

The Basic Problem

Suppose

- We observe a real-valued input variable $x \in \mathbb{R}$

The Basic Problem

Suppose

- We observe a real-valued input variable $x \in \mathbb{R}$

We are looking to predict

- The value of a real valued variable $y \in \mathbb{R}$

The Basic Problem

Suppose

- We observe a real-valued input variable $x \in \mathbb{R}$

We are looking to predict

- The value of a real valued variable $y \in \mathbb{R}$

Thus, we have the following training data set of size N

$$\boldsymbol{x} \equiv (x_1, x_2, \cdots, x_N)^T$$

$$y \equiv (y_1, y_2, \cdots, y_N)^T$$

Note: *We need data to construct prediction rules, often a lot of it.*

The Basic Problem

Suppose

- We observe a real-valued input variable $x \in \mathbb{R}$

We are looking to predict

- The value of a real valued variable $y \in \mathbb{R}$

Thus, we have the following training data set of size N

$$\boldsymbol{x} \equiv (x_1, x_2, \cdots, x_N)^T$$
$$\boldsymbol{y} \equiv (y_1, y_2, \cdots, y_N)^T$$

Note: We need data to construct prediction rules, often a lot of it.

The Basic Problem

Suppose

- We observe a real-valued input variable $x \in \mathbb{R}$

We are looking to predict

- The value of a real valued variable $y \in \mathbb{R}$

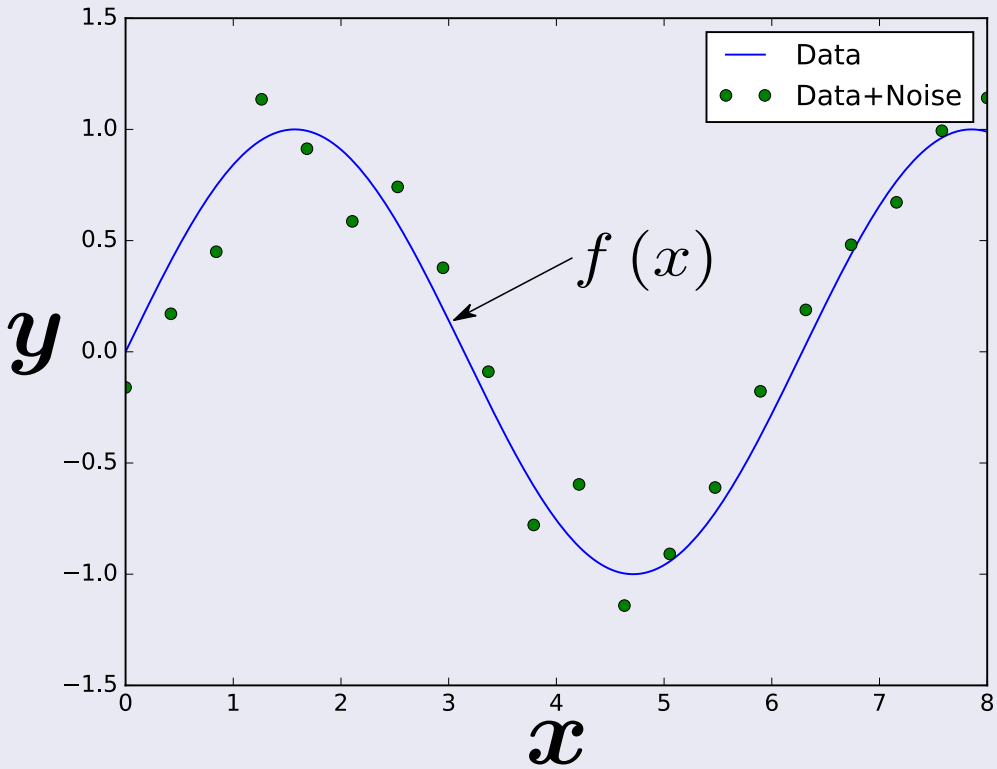
Thus, we have the following training data set of size N

$$\boldsymbol{x} \equiv (x_1, x_2, \cdots, x_N)^T$$
$$\boldsymbol{y} \equiv (y_1, y_2, \cdots, y_N)^T$$

Note: *We need data to construct prediction rules, often a lot of it.*

For Example

We have the function $g(x) = f(x) + \alpha U(0,1)$ with the real function $f(x) = \sin\{x\}$



What is our Goal?

Our goal is to exploit this training set

- We want to make predictions of the value $\widehat{y}$ (pronounced y-hat) given a new value $\widehat{x}$ (y-hat).

What can we use first?

$$y = g(x, \boldsymbol{w}) = w_0 + w_1 x + w_2 x^2 + \cdots + w_M x^d = \sum_{i=0}^{d} w_i x^i$$

Where

- d is the order of the polynomial.
- x^i denotes x raised to the power i.

What is our Goal?

Our goal is to exploit this training set

- We want to make predictions of the value $\widehat{y}$ (pronounced y-hat) given a new value $\widehat{x}$ (y-hat).

What can we use first?

$$y = g(x, \boldsymbol{w}) = w_0 + w_1 x + w_2 x^2 + \cdots + w_M x^d = \sum_{i=0}^{d} w_i x^i$$

Where

- d is the order of the polynomial.
- x^i denotes x raised to the power i.

What is our Goal?

Our goal is to exploit this training set

- We want to make predictions of the value $\widehat{y}$ (pronounced y-hat) given a new value $\widehat{x}$ (y-hat).

What can we use first?

$$y = g(x, \boldsymbol{w}) = w_0 + w_1 x + w_2 x^2 + \cdots + w_M x^d = \sum_{i=0}^{d} w_i x^i$$

Where

- d is the order of the polynomial.
- x^i denotes x raised to the power i.

Further

These functions are linear at the parameter w

- They are quite important and are called ***linear models!!!***

How do we guess these values?

- By fitting the polynomial to the training data.

How do we do this?

- This can be done by minimizing an error function or loss function measuring, ϵ:
  - The difference between the function $g(x, w)$, for any given value of w, and the training set data points.

Further

These functions are linear at the parameter w

- They are quite important and are called ***linear models!!!***

How do we guess these values?

- By fitting the polynomial to the training data.

How do we do this?

- This can be done by minimizing an error function or loss function measuring, ϵ:
  - The difference between the function $g(x, w)$, for any given value of w, and the training set data points.

Further

These functions are linear at the parameter w

- They are quite important and are called ***linear models!!!***

How do we guess these values?

- By fitting the polynomial to the training data.

How do we do this?

- This can be done by minimizing an error function or loss function measuring, ϵ:
  - The difference between the function $g(x, \boldsymbol{w})$, for any given value of $\boldsymbol{w}$, and the training set data points.

Outline

One simple choice of error function

The Sum of the Squares of the Errors

$$E(\boldsymbol{w}) = \frac{1}{2}\sum_{i=1}^{N}\left[g(x_i, \boldsymbol{w}) - y_i\right]^2$$

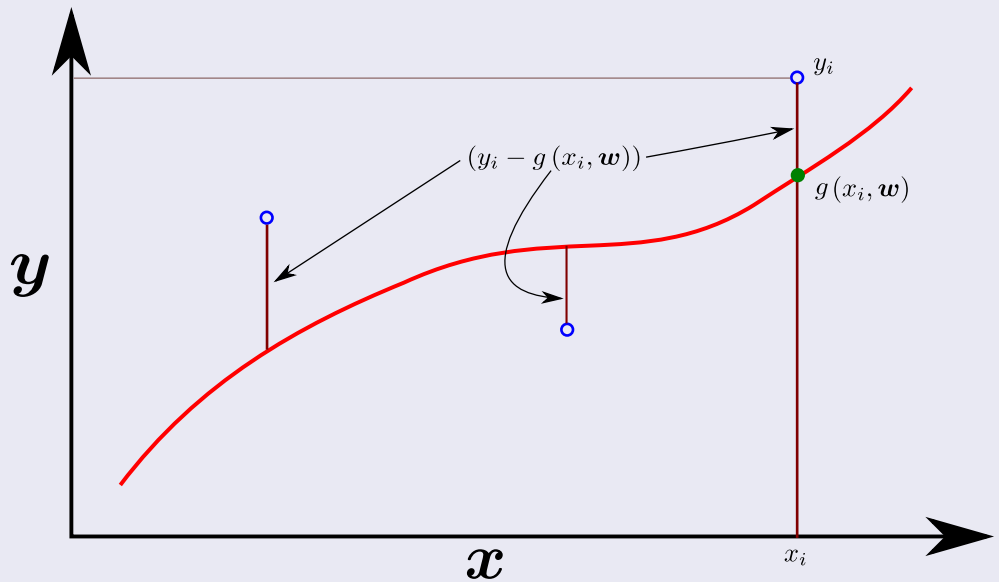
Something Notable

One simple choice of error function

The Sum of the Squares of the Errors

$$E(\boldsymbol{w}) = \frac{1}{2}\sum_{i=1}^{N}\left[g(x_i, \boldsymbol{w}) - y_i\right]^2$$

Something Notable



Outline

Case 1

Choose the estimate of $f(x)$, $g(x, \boldsymbol{w})$, to be independent of $\mathcal{D}$

For example, $g(x, \boldsymbol{w}) = w_1 x + w_0$

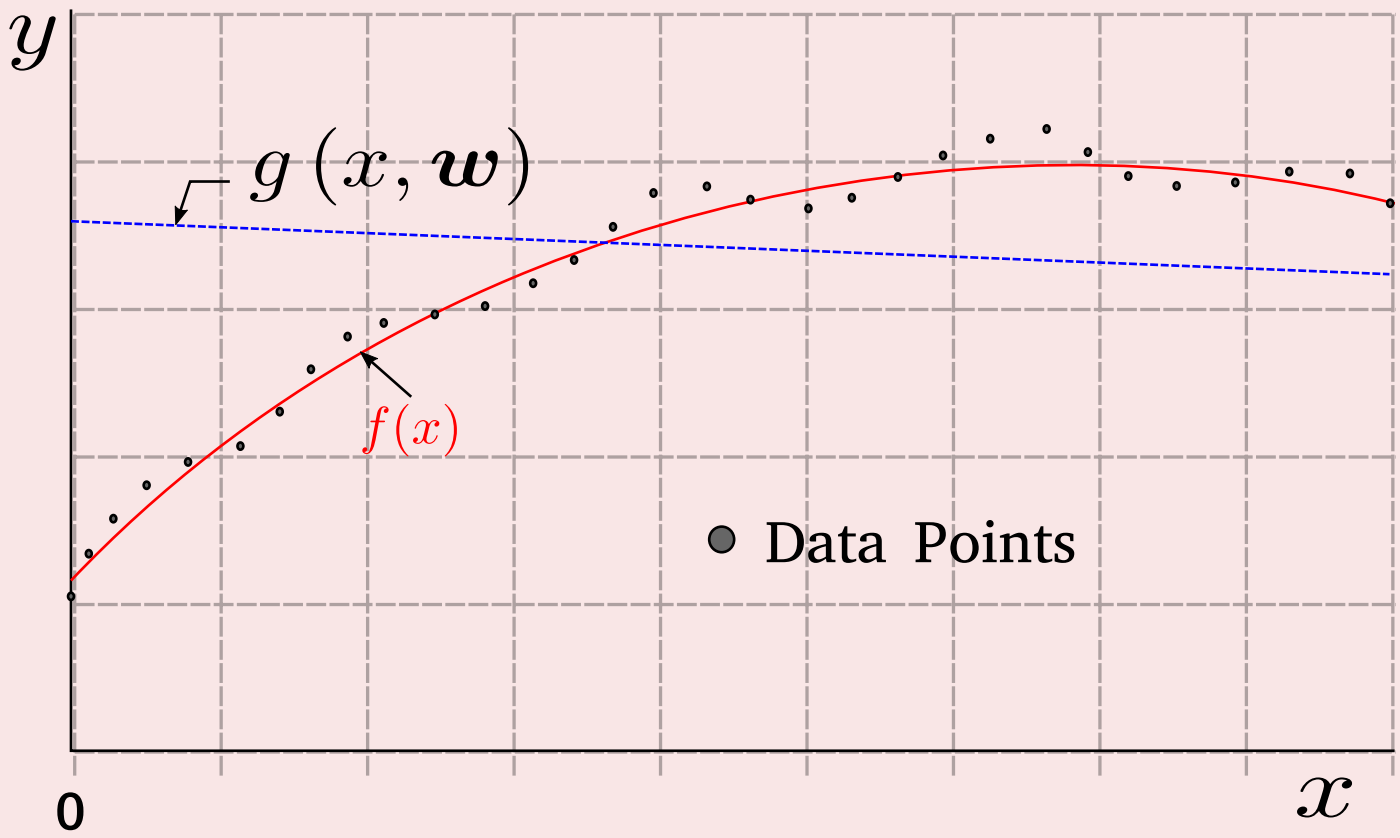
We call this **HIGH BIAS**

Case 1

Choose the estimate of $f(x)$, $g(x, \boldsymbol{w})$, to be independent of $\mathcal{D}$

For example, $g(x, \boldsymbol{w}) = w_1 x + w_0$

We call this **HIGH BIAS**



Case 2

In the other hand

Now, $g(x, \boldsymbol{w})$ corresponds to a polynomial of high degree so it can pass through each training point.

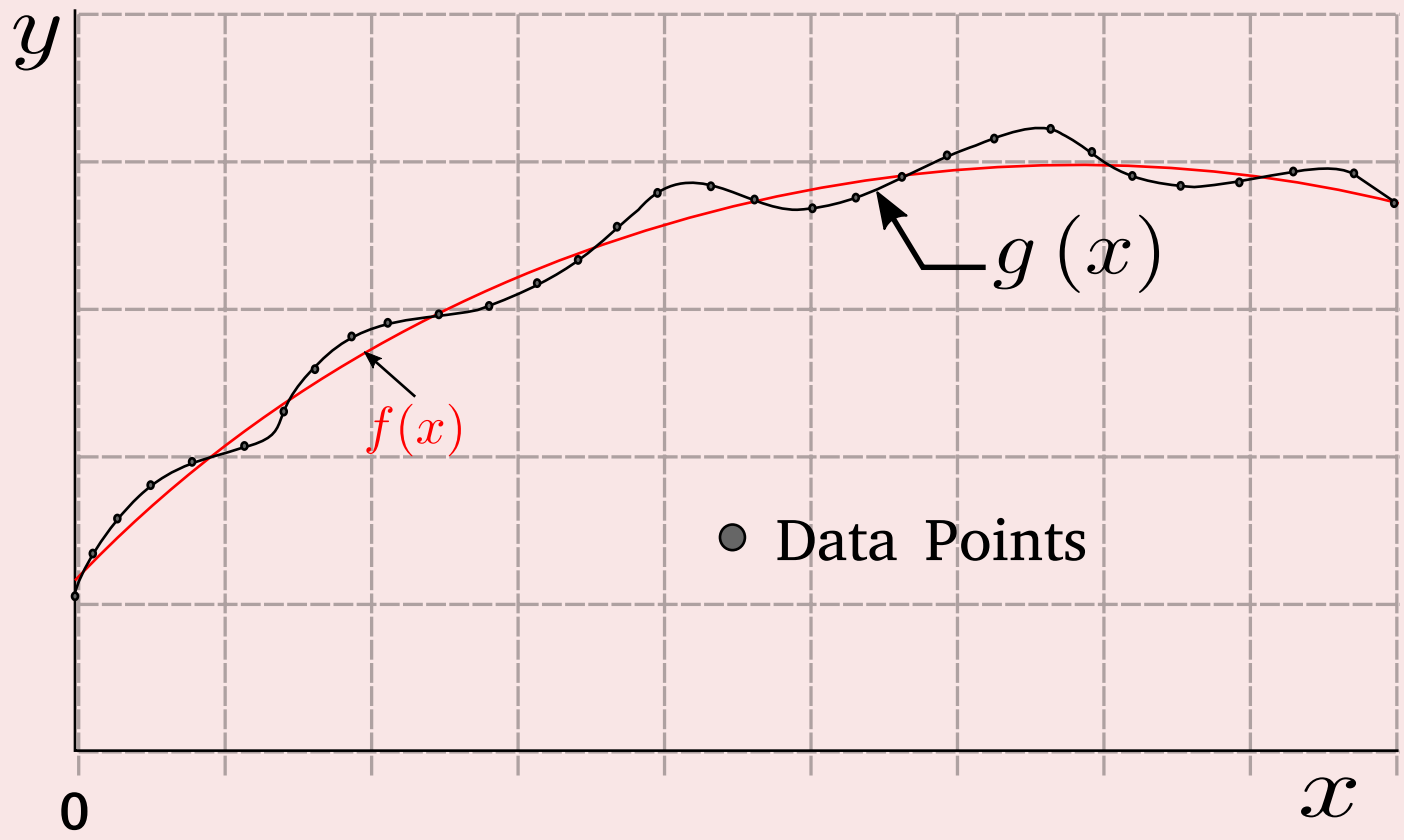
We call this **HIGH VARIANCE**

Case 2

In the other hand

Now, $g(x, \boldsymbol{w})$ corresponds to a polynomial of high degree so it can pass through each training point.

We call this **HIGH VARIANCE**



Outline

Outline

Our General Case

Our Data Set

1. A Series of $X \in \mathbb{R}^d$ of real valued random input vector.

$$x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{pmatrix}$$

   - Here, each variable X_i is Quantitative or Qualitative variables in the correct numeric representation.

2. A Series of $Y \in \mathbb{R}$ a real valued random output variables.

Our General Case

Our Data Set

1. A Series of $X \in \mathbb{R}^d$ of real valued random input vector.

$$x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{pmatrix}$$

   - Here, each variable X_i is Quantitative or Qualitative variables in the correct numeric representation.

2. A Series of $Y \in \mathbb{R}$ a real valued random output variables.

Our General Case

Our Data Set

1. A Series of $X \in \mathbb{R}^d$ of real valued random input vector.

$$x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{pmatrix}$$

   - Here, each variable X_i is Quantitative or Qualitative variables in the correct numeric representation.

2. A Series of $Y \in \mathbb{R}$ a real valued random output variables.

Linear Models

We have the following model

- The linear model has been a mainstay of statistics for the past 30 years.

The Model looks like on an input $X^T = (X_1, X_2, \ldots, X_d)$

$$\hat{Y} = \hat{w}_0 + \sum_{i=1}^{d} X_i \hat{w}_i$$

Linear Models

We have the following model

- The linear model has been a mainstay of statistics for the past 30 years.

The Model looks like on an input $X^T = (X_1, X_2, \ldots, X_d)$

$$\widehat{Y} = \widehat{w}_0 + \sum_{i=1}^{d} X_i \widehat{w}_i$$

It is many times convenient

To use the dot product in Linear Algebra

$$\widehat{Y} = (1, X_1, X_2, \ldots, X_d) \begin{pmatrix} \widehat{w}_0 \\ \widehat{w}_1 \\ \vdots \\ \widehat{w}_d \end{pmatrix} = X^T \widehat{\boldsymbol{w}}$$

Furthermore, $\widehat{Y}$ could be a constant or a N vector

$$\widehat{Y} = \begin{pmatrix} \widehat{Y}_1 \\ \widehat{Y}_2 \\ \vdots \\ \widehat{Y}_N \end{pmatrix} = \begin{pmatrix} 1 & X_1^{(1)} & X_2^{(1)} & \cdots & X_d^{(1)} \\ 1 & X_1^{(2)} & X_2^{(2)} & \cdots & X_d^{(2)} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 1 & X_1^{(N)} & X_2^{(N)} & \cdots & X_d^{(N)} \end{pmatrix} \begin{pmatrix} \widehat{w}_0 \\ \widehat{w}_1 \\ \vdots \\ \widehat{w}_d \end{pmatrix} = X w$$

It is many times convenient

To use the dot product in Linear Algebra

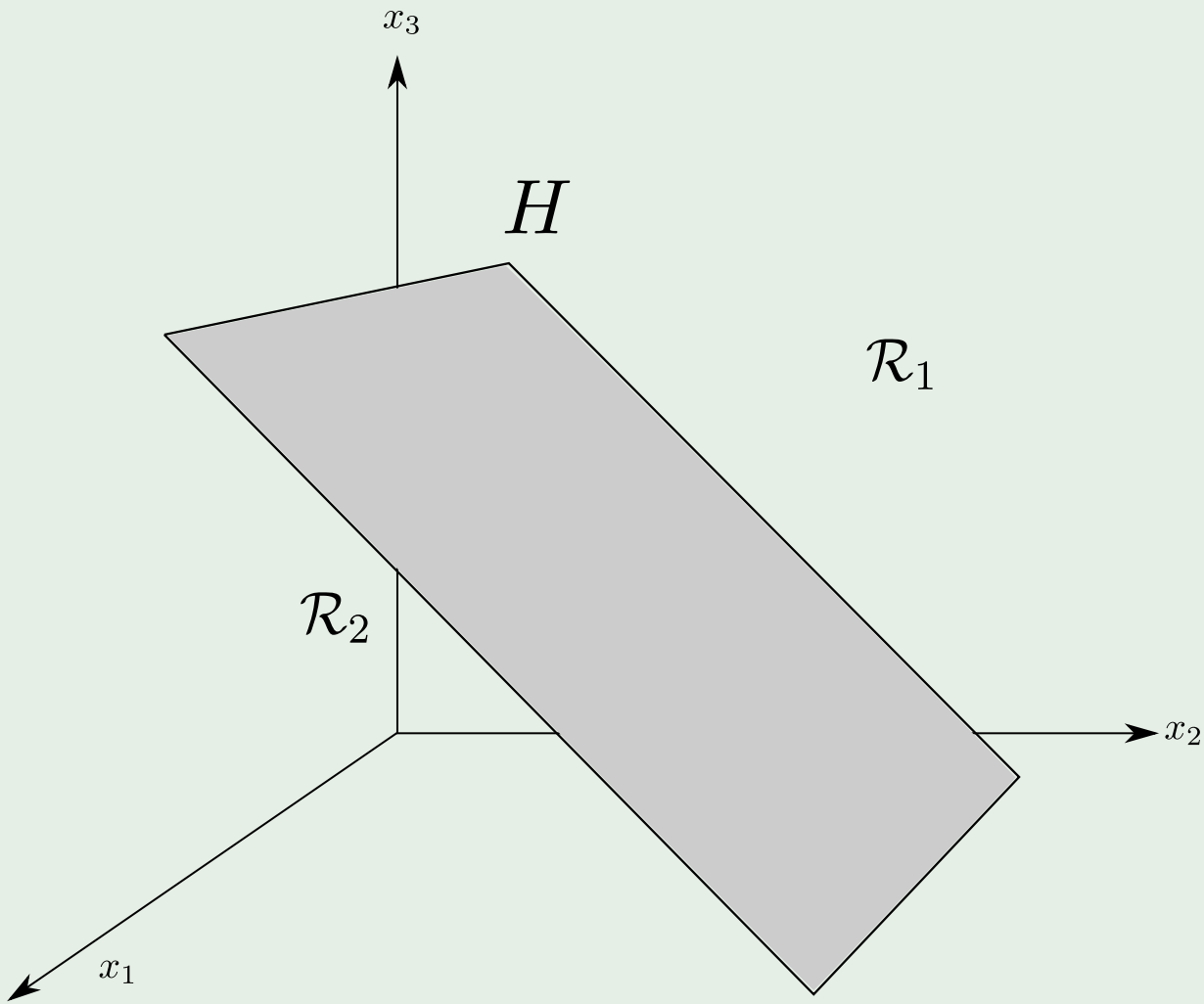
$$\widehat{Y} = (1, X_1, X_2, \ldots, X_d) \begin{pmatrix} \widehat{w}_0 \\ \widehat{w}_1 \\ \vdots \\ \widehat{w}_d \end{pmatrix} = X^T \widehat{\boldsymbol{w}}$$

Furthermore, $\widehat{Y}$ could be a constant or a N vector

$$\widehat{Y} = \begin{pmatrix} \widehat{Y}_1 \\ \widehat{Y}_2 \\ \vdots \\ \widehat{Y}_N \end{pmatrix} = \begin{pmatrix} 1 & X_1^{(1)} & X_2^{(1)} & \cdots & X_d^{(1)} \\ 1 & X_1^{(2)} & X_2^{(2)} & \cdots & X_d^{(2)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & X_1^{(N)} & X_2^{(N)} & \cdots & X_d^{(N)} \end{pmatrix} \begin{pmatrix} \widehat{w}_0 \\ \widehat{w}_1 \\ \vdots \\ \widehat{w}_d \end{pmatrix} = \boldsymbol{X}\boldsymbol{w}$$

This basically define an hyperplane

The space is split in two regions (Example in $\mathbb{R}^3$) by the hyperplane H



A Convenient Loss Functions

Thus, we look for a Loss function (A convenient one the LSE)

$$L(\boldsymbol{w}) = \sum_{i=1}^{N} \left(\boldsymbol{y}_i - \boldsymbol{x}_i^T \boldsymbol{w} \right)^2$$

Then

It is possible to get a unique solution

$$\boldsymbol{w} = \left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\boldsymbol{y}$$

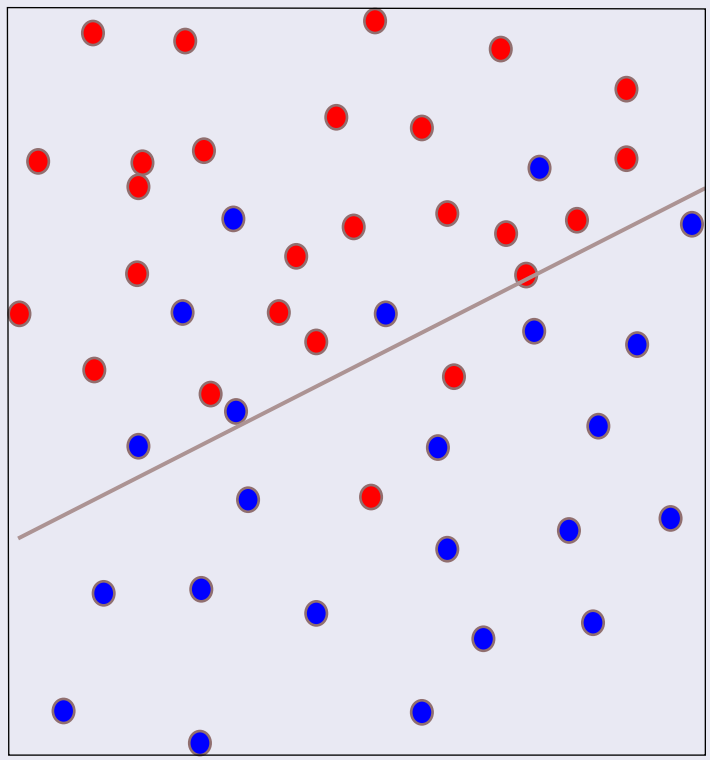
Then, it is possible to fit the linear model to the following data

Then

It is possible to get a unique solution

$$\boldsymbol{w} = \left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\boldsymbol{y}$$

Then, it is possible to fit the linear model to the following data



How do we do classification here?

Given

1. $Y = 0$ for the **blue** data set.
2. $Y = 1$ for the **red** data set.

Then, the fitted values $\hat{Y}$ are converted to a fitted class variable $\hat{G}$ according

$$\hat{G} = \begin{cases} \text{red} & \text{if } \hat{Y} > 0.5 \\ \text{blue} & \text{if } \hat{Y} \leq 0.5 \end{cases}$$

How do we do classification here?

Given

1. $Y = 0$ for the **blue** data set.
2. $Y = 1$ for the **red** data set.

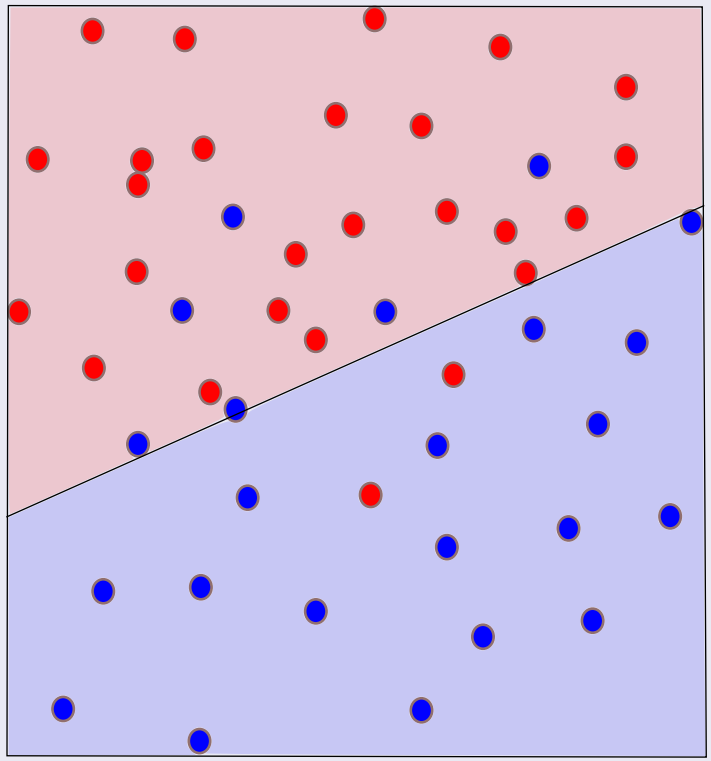
Then, the fitted values $\widehat{Y}$ are converted to a fitted class variable $\widehat{G}$ according

$$\widehat{G} = \begin{cases} \text{red} & \text{if } \widehat{Y} > 0.5 \\ \text{blue} & \text{if } \widehat{Y} \leq 0.5 \end{cases}$$

Decision Boundary

The two predicted classes are separated

Decision Boundary $\left\{\boldsymbol{x}|\boldsymbol{x}^T\widehat{\boldsymbol{w}} = 0.5\right\}$



We have a Problem

We have and issue

We do not know the underlaying models that generates the data.

Scenario 1

- The training data in each class were generated from bivariate Gaussian distributions with uncorrelated components and different means.

Thus!!!

- Look at the Blackboard

We have a Problem

We have and issue

We do not know the underlaying models that generates the data.

Scenario 1

- The training data in each class were generated from bivariate Gaussian distributions with uncorrelated components and different means.

Thus!!!

- Look at the Blackboard

We have a Problem

We have and issue

We do not know the underlaying models that generates the data.

Scenario 1

- The training data in each class were generated from bivariate Gaussian distributions with uncorrelated components and different means.

Thus!!!

- Look at the Blackboard

What is happening?

Scenario 2

- The training data in each class came from a mixture of 10 low-variance Gaussian distributions, with individual means themselves distributed as Gaussian.

Then

- Again to the Blackboard!!!

What is happening?

Scenario 2

- The training data in each class came from a mixture of 10 low-variance Gaussian distributions, with individual means themselves distributed as Gaussian.

Then

- Again to the Blackboard!!!

Outline

Nearest-Neighbor Methods

Nearest-neighbor methods use those observations in the training set

- Which are closets in the input space to a sample $\boldsymbol{x}$ to from $\widehat{Y}$.

K-Nearest Formulation

$$\widehat{Y}(\boldsymbol{x}) = \frac{1}{k} \sum_{\boldsymbol{x}_i \in N_k(\boldsymbol{x})} y_i$$

Where $N_k(\boldsymbol{x})$ is the neighborhood of $\boldsymbol{x}$ defined by the k closest points $\boldsymbol{x}_i$ in the training sample.

Nearest-Neighbor Methods

Nearest-neighbor methods use those observations in the training set

- Which are closets in the input space to a sample $\boldsymbol{x}$ to from $\widehat{Y}$.

K-Nearest Formulation

$$\widehat{Y}(\boldsymbol{x}) = \frac{1}{k} \sum_{\boldsymbol{x}_i \in N_k(\boldsymbol{x})} y_i$$

Where $N_k(\boldsymbol{x})$ is the neighborhood of $\boldsymbol{x}$ defined by the k closest points $\boldsymbol{x}_i$ in the training sample.

Clearly $N_k(\boldsymbol{x})$ requires a distance

Implies a Distance!!! Which one?

$$d_2(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{\boldsymbol{x}^T \boldsymbol{y}} \leftarrow\text{-}\text{-} \text{ Euclidean Distance}$$

$$d_1(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=}^{d} |x_i - y_i| \leftarrow\text{-}\text{-} \text{ Manhattan Distance}$$

$$d_p(\boldsymbol{x}, \boldsymbol{y}) = \left(\sum_{i=}^{d} |x_i - y_i|^p \right)^{\frac{1}{p}} \leftarrow\text{-}\text{-} \text{ Minkowski distance of order } p$$

Clearly $N_k(\boldsymbol{x})$ requires a distance

Implies a Distance!!! Which one?

$$d_2(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{\boldsymbol{x}^T \boldsymbol{y}} \text{ <-- Euclidean Distance}$$

$$d_1(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=}^{d} |x_i - y_i| \text{ <-- Manhattan Distance}$$

$$d_p(\boldsymbol{x}, \boldsymbol{y}) = \left(\sum_{i=}^{d} |x_i - y_i|^p \right)^{\frac{1}{p}} \text{ <-- Minkowski distance of order } p$$

Clearly $N_k(\boldsymbol{x})$ requires a distance

Implies a Distance!!! Which one?

$$d_2(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{\boldsymbol{x}^T \boldsymbol{y}} \text{ <-- Euclidean Distance}$$

$$d_1(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=}^{d} |x_i - y_i| \text{ <-- Manhattan Distance}$$

$$d_p(\boldsymbol{x}, \boldsymbol{y}) = \left(\sum_{i=}^{d} |x_i - y_i|^p \right)^{\frac{1}{p}} \text{ <-- Minkowski distance of order } p$$

Furthermore

Given a Data Matrix $\boldsymbol{X}$ and the Mean Data Matrix $\overline{\boldsymbol{X}}$

$$\boldsymbol{X} = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N1} & x_{N2} & \cdots & x_{Np} \end{pmatrix}, \quad \overline{\boldsymbol{X}} = \begin{pmatrix} \overline{\boldsymbol{x}} \\ \overline{\boldsymbol{x}} \\ \vdots \\ \overline{\boldsymbol{x}} \end{pmatrix} \text{ with}$$

$$\overline{\boldsymbol{X}} = \frac{1}{N}\sum_{i=1}^{N} \begin{pmatrix} x_{i1} & x_{i2} & \cdots & x_{ip} \end{pmatrix}^T$$

We generate the variance-covariance matrix

$$C_{\boldsymbol{X}} = \frac{1}{N-1}\left[\boldsymbol{X} - \overline{\boldsymbol{X}}\right]^T \left[\boldsymbol{X} - \overline{\boldsymbol{X}}\right]$$

Furthermore

Given a Data Matrix $\boldsymbol{X}$ and the Mean Data Matrix $\overline{\boldsymbol{X}}$

$$\boldsymbol{X} = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N1} & x_{N2} & \cdots & x_{Np} \end{pmatrix}, \quad \overline{\boldsymbol{X}} = \begin{pmatrix} \overline{\boldsymbol{x}} \\ \overline{\boldsymbol{x}} \\ \vdots \\ \overline{\boldsymbol{x}} \end{pmatrix} \text{ with}$$

$$\overline{\boldsymbol{X}} = \frac{1}{N}\sum_{i=1}^{N} \begin{pmatrix} x_{i1} & x_{i2} & \cdots & x_{ip} \end{pmatrix}^T$$

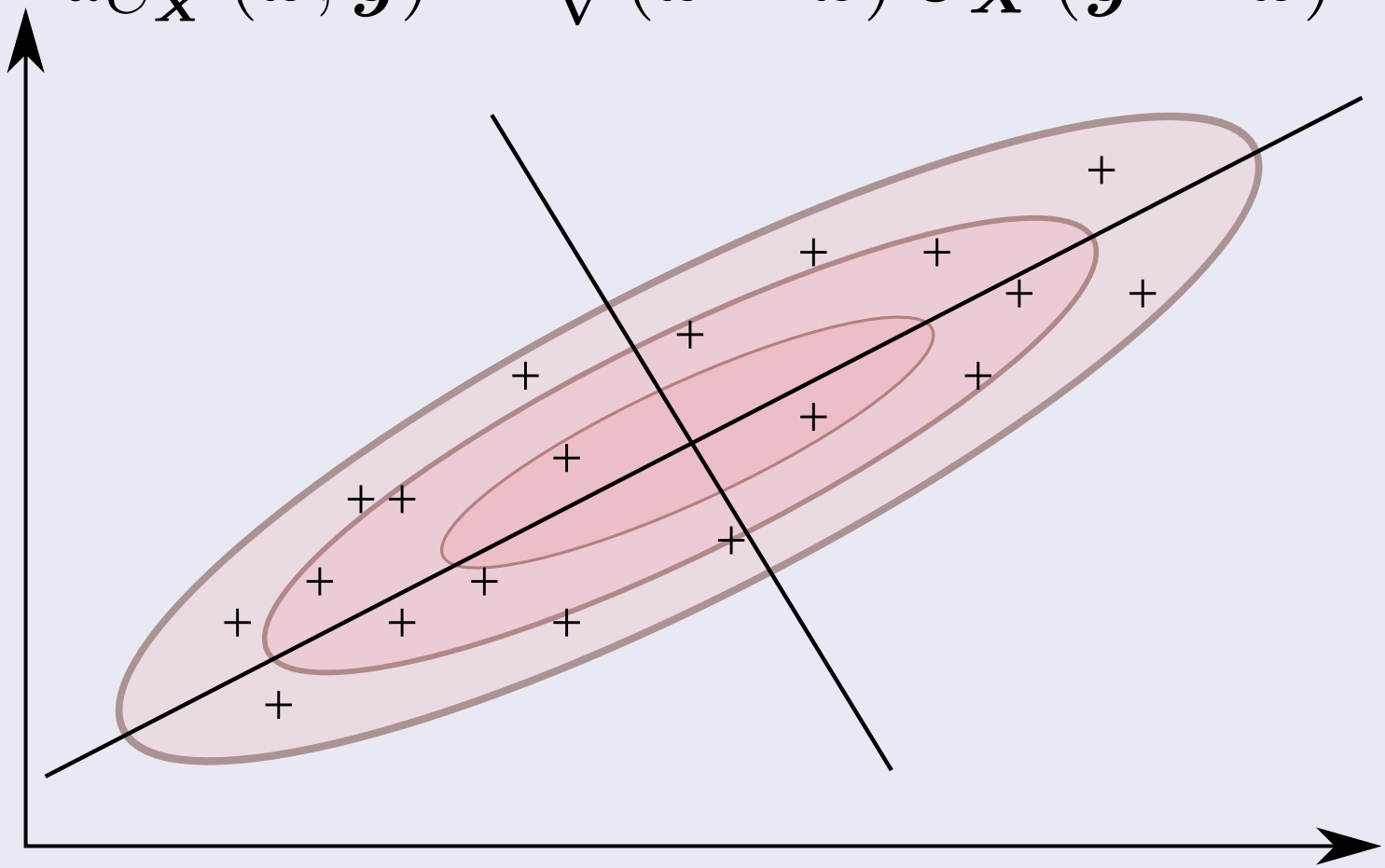
We generate the variance-covariance matrix

$$C_{\boldsymbol{X}} = \frac{1}{N-1}\left[\boldsymbol{X} - \overline{\boldsymbol{X}}\right]^T \left[X - \overline{\boldsymbol{X}}\right]$$

Then, we have

The Mahalanobis Distance

$$d_{C_{\boldsymbol{X}}}(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{(\boldsymbol{x} - \overline{\boldsymbol{x}})\, C_{\boldsymbol{X}}\, (\boldsymbol{y} - \overline{\boldsymbol{x}})}$$



Therefore

we find the k observations

With x_i closest to x in input space, and average their responses.

And Again

$$\hat{G} = \begin{cases} \text{red} & \text{if } \hat{Y} > 0.5 \\ \text{blue} & \text{if } \hat{Y} \leq 0.5 \end{cases}$$

Therefore

we find the k observations

With x_i closest to x in input space, and average their responses.

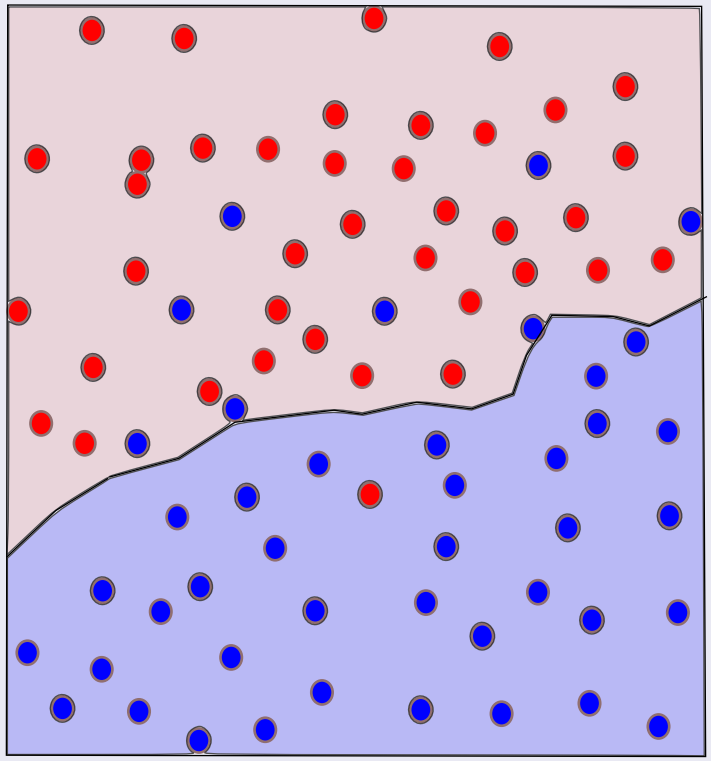
And Again

$$\widehat{G} = \begin{cases} \text{red} & \text{if } \widehat{Y} > 0.5 \\ \text{blue} & \text{if } \widehat{Y} \leq 0.5 \end{cases}$$

Example

We have only five neighbor, $K = 5$

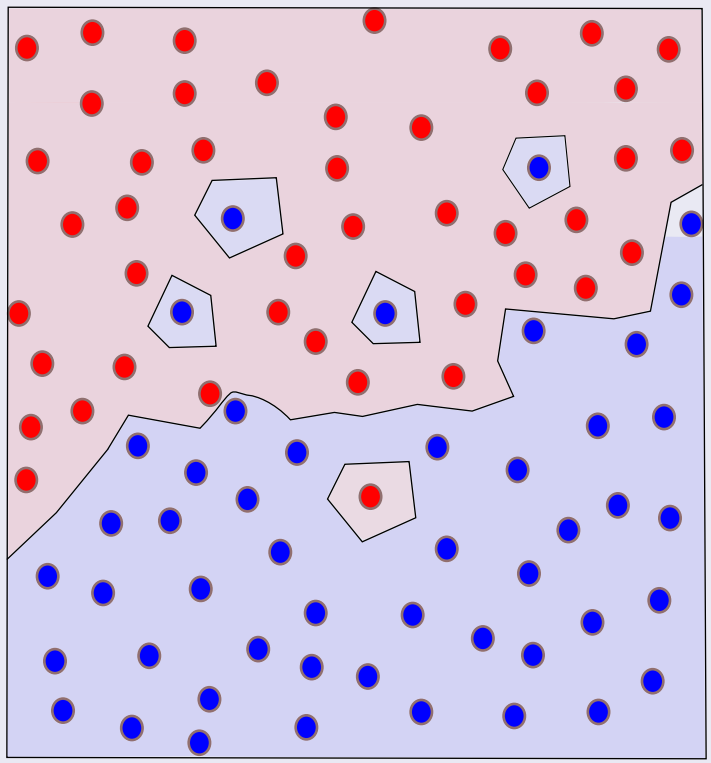
$k = 5$



Example - Actually The Voronoi Tessellation of the Training Data

We have only one neighbor, $K = 1$

$k = 1$



Note: Each point x_i has an associated tile bounding the region for which it is the closest input point.

Therefore

$K = 1$ Vs. $K = 5$

For $K = 5$, we see that far fewer training observations are misclassified when compared with the Linear Model

With $K = 1$

None of the training data are misclassified!!!

Therefore

$K = 1$ Vs. $K = 5$

For $K = 5$, we see that far fewer training observations are misclassified when compared with the Linear Model

With $K = 1$

None of the training data are misclassified!!!

Outline

For example

Kernel methods

- They use weights that decrease smoothly to zero with distance from the target point,
  - Quite different rather from using 0/1 weights used by k-nearest neighbors.

For example

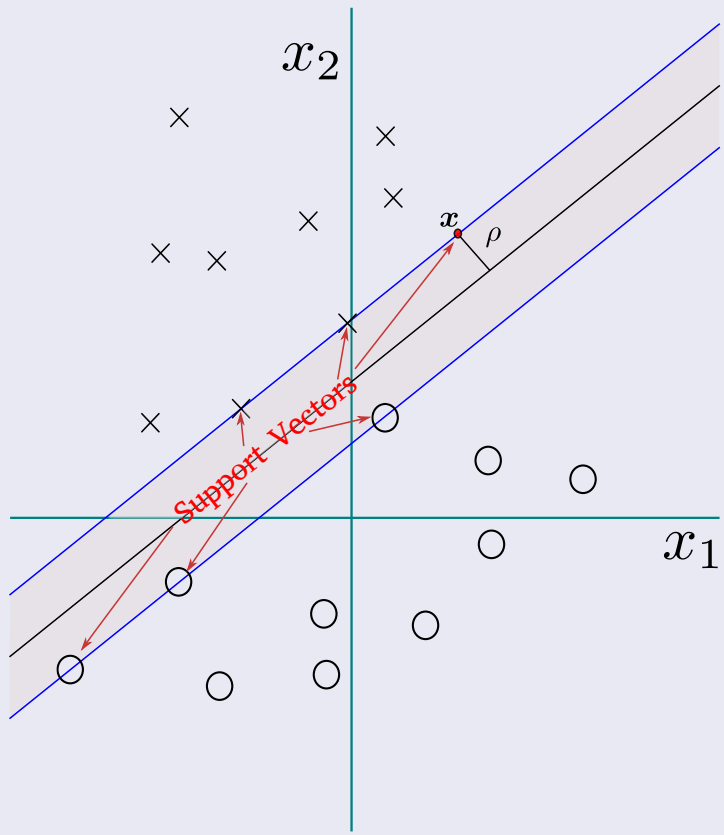
Kernel methods

- They use weights that decrease smoothly to zero with distance from the target point,
  - Quite different rather from using 0/1 weights used by k-nearest neighbors.

For example

Kernel methods

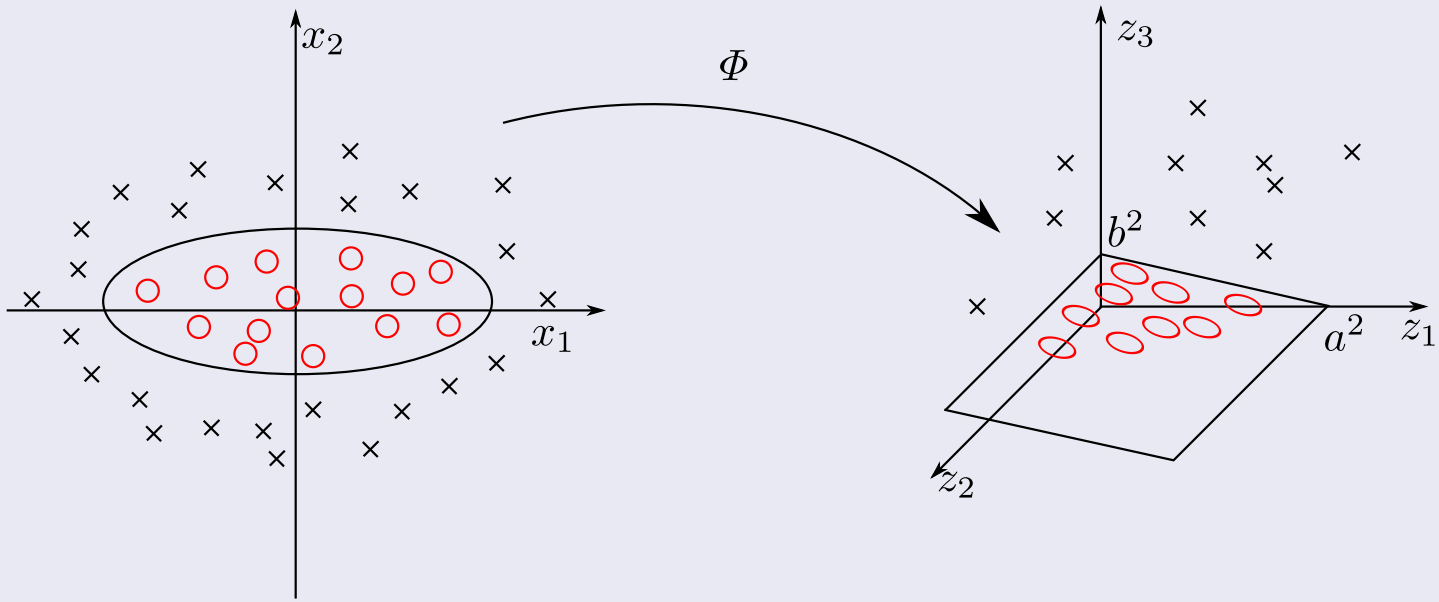
- They use weights that decrease smoothly to zero with distance from the target point,
  - Quite different rather from using 0/1 weights used by k-nearest neighbors.



Furthermore

Something Notable

- In High-Dimensional spaces the distance kernels are modified to obtain better classifications.



$$\Phi : (x_1, x_2) \rightarrow \left(x_1^2, \sqrt{2}x_1x_2, x_2^2\right)$$

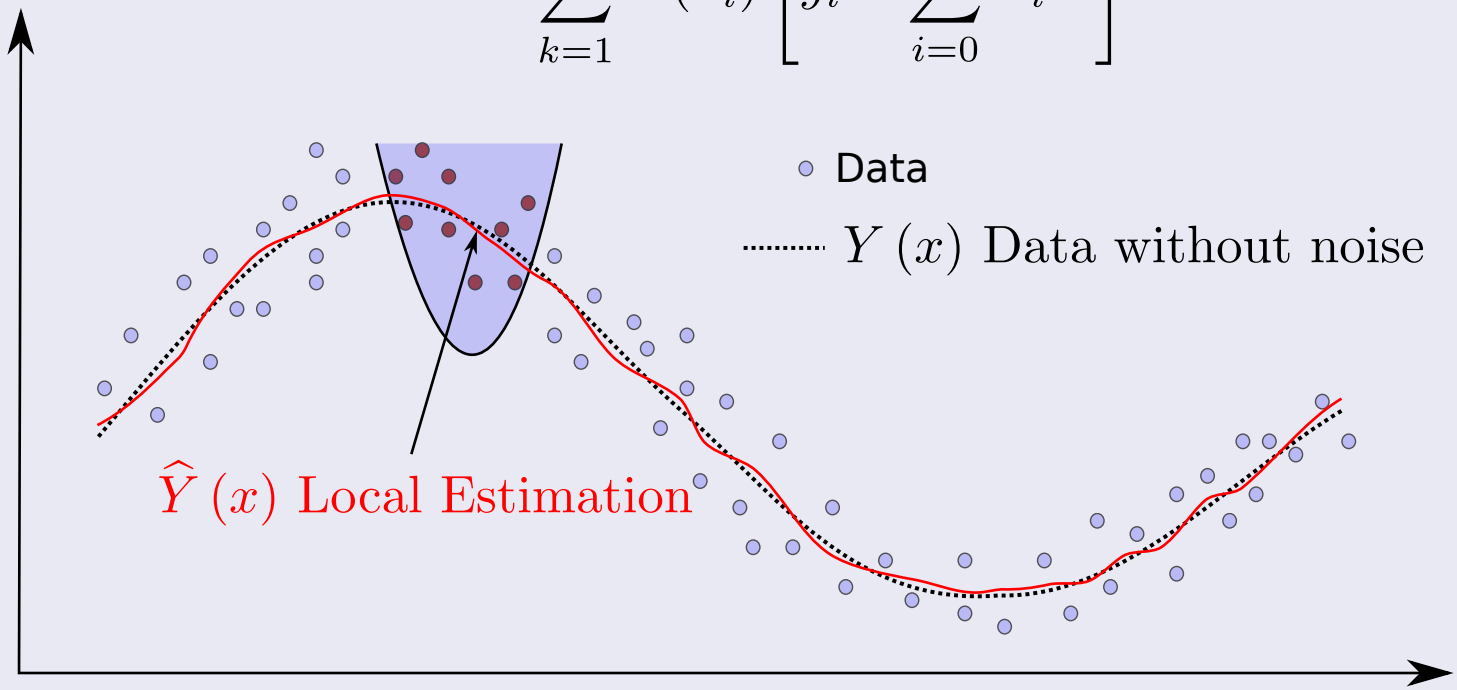
$$\left(\frac{x_1}{a}\right)^2 + \left(\frac{x_2}{b}\right)^2 = 1 \rightarrow \frac{z_1}{a^2} + \frac{z_3}{b^2} = 1$$

Example

Local Regression

Local regression fits linear models by locally weighted least squares.

$$\sum_{k=1}^{N} w(x_i) \left[y_i - \sum_{i=0}^{d} w_i x^i \right]^2$$



Outline

The Samples as Random Variables

As Always Probability

We first consider:

- $X \in \mathbb{R}^d$ denote a real valued input vector
- $Y \in \mathbb{R}$ a real valued random output

Therefore, we have a Joint Distribution $P(X, Y)$ and we seek

$$f(X) \text{ predicting } Y$$

The Samples as Random Variables

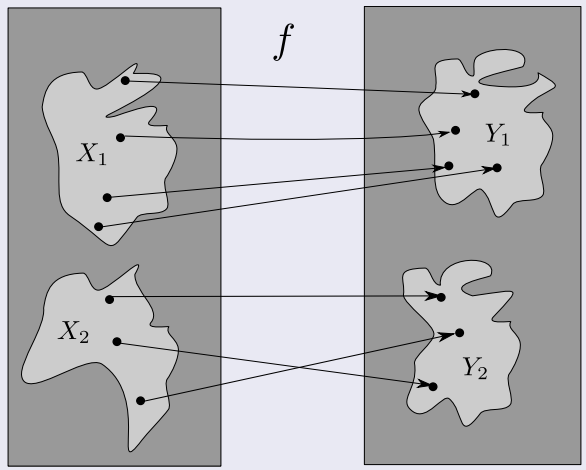
As Always Probability

We first consider:

- $X \in \mathbb{R}^d$ denote a real valued input vector
- $Y \in \mathbb{R}$ a real valued random output

Therefore, we have a Joint Distribution $P(X, Y)$ and we seek

$$f(X) \text{ predicting } Y$$



Outline

We require a Loss Function

A convenient one is the Squared Error Loss

$$L(Y, f(X)) = (Y - f(X))^2$$

There is a relation to noise $\epsilon \sim N(0, 1)$

$$Y_{noise}(X) = f(X) + \epsilon$$

The Squared Error Loss

- It tries to minimize the error ϵ!!!

We require a Loss Function

A convenient one is the Squared Error Loss

$$L\left(Y, f\left(X\right)\right) = \left(Y - f\left(X\right)\right)^2$$

There is a relation to noise $\epsilon \sim N\left(0, 1\right)$

$$Y_{noise}\left(X\right) = f\left(X\right) + \epsilon$$

The Squared Error Loss

- It tries to minimize the error ϵ!!!

We require a Loss Function

A convenient one is the Squared Error Loss

$$L\left(Y, f\left(X\right)\right) = \left(Y - f\left(X\right)\right)^2$$

There is a relation to noise $\epsilon \sim N\left(0, 1\right)$

$$Y_{noise}\left(X\right) = f\left(X\right) + \epsilon$$

The Squared Error Loss

- It tries to minimize the error ϵ!!!

This leads us to a criterion for choosing f

The Expected Prediction Error (EPE)

$$\begin{aligned} EPE &= E\left(Y - f\left(X\right)\right)^2 \\ &= \int \left[y - f\left(x\right)\right]^2 p_{xy}\left(x, y\right) dxdy \end{aligned}$$

Now, we can condition the probability density function with respect to X

$$p\left(X, Y\right) = p\left(Y|X\right) p\left(X\right)$$

This leads us to a criterion for choosing f

The Expected Prediction Error (EPE)

$$\begin{aligned} EPE &= E\left(Y - f\left(X\right)\right)^2 \\ &= \int \left[y - f\left(x\right)\right]^2 p_{xy}\left(x, y\right) dxdy \end{aligned}$$

Now, we can condition the probability density function with respect to X

$$p\left(X, Y\right) = p\left(Y|X\right) p\left(X\right)$$

Thus

We have

$$\int [y - f(x)]^2 p_{xy}(x,y)\, dxdy = \int_X \int_Y [y - f(x)]^2 p_{y|x}(y|x)\, p_x(x)\, dxdy$$

$$= \int_X \left[\int_Y [y - f(x)]^2 p_{y|x}(y|x)\, dy \right] dx$$

$$= E_X \left[\int_Y [y - f(x)]^2 p_{y|x}(y|x)\, dy \right]$$

$$= E_X E_{Y|X} \left[(Y - f(X))^2 | X \right]$$

Thus

We have

$$\begin{aligned}\int [y - f(x)]^2 p_{xy}(x,y)\,dxdy &= \int_X \int_Y [y - f(x)]^2 p_{y|x}(y|x)\, p_x(x)\,dxdy \\ &= \int_X \left[\int_Y [y - f(x)]^2 p_{y|x}(y|x)\,dy\right] dx \\ &= E_X\left[\int_Y [y - f(x)]^2 p_{y|x}(y|x)\,dy\right] \\ &= E_X E_{Y|X}\left[(Y - f(X))^2 | X\right]\end{aligned}$$

What happens if we fix X?

$$EPE(f)_{X=x} = E_{Y|X=x}\left[(Y - f(x))^2 | X = x\right]$$

Thus

We have

$$\begin{aligned}\int [y - f(x)]^2 p_{xy}(x,y)\,dxdy &= \int_X \int_Y [y - f(x)]^2 p_{y|x}(y|x)\,p_x(x)\,dxdy \\ &= \int_X \left[\int_Y [y - f(x)]^2 p_{y|x}(y|x)\,dy\right] dx \\ &= E_X\left[\int_Y [y - f(x)]^2 p_{y|x}(y|x)\,dy\right] \\ &= E_X E_{Y|X}\left[(Y - f(X))^2 | X\right]\end{aligned}$$

What happens if we fix X?

$$EPE(f)_{X=x} = E_{Y|X=x}\left[(Y - f(x))^2 | X = x\right]$$

Thus

We have

$$\begin{aligned}\int [y-f(x)]^2 p_{xy}(x,y)\,dxdy &= \int_X \int_Y [y-f(x)]^2 p_{y|x}(y|x)\,p_x(x)\,dxdy \\ &= \int_X \left[\int_Y [y-f(x)]^2 p_{y|x}(y|x)\,dy\right] dx \\ &= E_X\left[\int_Y [y-f(x)]^2 p_{y|x}(y|x)\,dy\right] \\ &= E_X E_{Y|X}\left[(Y-f(X))^2|X\right]\end{aligned}$$

What happens if we fix X?

$$EPE(f)_{X=x} = E_{Y|X=x}\left[(Y-f(x))^2|X=x\right]$$

Thus

We have

$$\begin{aligned}\int [y - f(x)]^2 p_{xy}(x,y)\, dxdy &= \int_X \int_Y [y - f(x)]^2 p_{y|x}(y|x)\, p_x(x)\, dxdy \\ &= \int_X \left[\int_Y [y - f(x)]^2 p_{y|x}(y|x)\, dy \right] dx \\ &= E_X \left[\int_Y [y - f(x)]^2 p_{y|x}(y|x)\, dy \right] \\ &= E_X E_{Y|X} \left[(Y - f(X))^2 | X \right]\end{aligned}$$

What happens if we fix X?

$$EPE(f)_{X=\boldsymbol{x}} = E_{Y|X=\boldsymbol{x}} \left[(Y - f(\boldsymbol{x}))^2 | X = \boldsymbol{x} \right]$$

We can optimize the function

By a Simple Analysis

$$E_{Y|X=\boldsymbol{x}}\left[\left(Y-f\left(\boldsymbol{x}\right)\right)^{2}|X=\boldsymbol{x}\right]=E_{Y|X=\boldsymbol{x}}\left[\left(Y+\overline{Y}-\overline{Y}-f\left(\boldsymbol{x}\right)\right)^{2}|X=\boldsymbol{x}\right]$$

$$=E_{Y|X=\boldsymbol{x}}\left[\left(Y-\overline{Y}\right)^{2}|X=\boldsymbol{x}\right]+...$$

$$E_{Y|X=\boldsymbol{x}}\left[\left(\overline{Y}-f\left(\boldsymbol{x}\right)\right)^{2}|X=\boldsymbol{x}\right]+...$$

$$2E_{Y|X=\boldsymbol{x}}\left[\left(\overline{Y}-f\left(\boldsymbol{x}\right)\right)\left(Y-\overline{Y}\right)|X=\boldsymbol{x}\right]$$

$$=E_{Y|X=\boldsymbol{x}}\left[\left(Y-\overline{Y}\right)^{2}|X=\boldsymbol{x}\right]+...$$

$$E_{Y|X=\boldsymbol{x}}\left[\left(\overline{Y}-f\left(\boldsymbol{x}\right)\right)^{2}|X=\boldsymbol{x}\right]+...$$

$$2\left(\overline{Y}-f\left(\boldsymbol{x}\right)\right)E_{Y|X=\boldsymbol{x}}\left[\left(Y-\overline{Y}\right)|X=\boldsymbol{x}\right]$$

We can optimize the function

By a Simple Analysis

$$
\begin{aligned}
E_{Y|X=\boldsymbol{x}}\left[\left(Y-f\left(\boldsymbol{x}\right)\right)^{2}|X=\boldsymbol{x}\right] &= E_{Y|X=\boldsymbol{x}}\left[\left(Y+\overline{Y}-\overline{Y}-f\left(\boldsymbol{x}\right)\right)^{2}|X=\boldsymbol{x}\right] \\
&= E_{Y|X=\boldsymbol{x}}\left[\left(Y-\overline{Y}\right)^{2}|X=\boldsymbol{x}\right]+... \\
&\quad E_{Y|X=\boldsymbol{x}}\left[\left(\overline{Y}-f\left(\boldsymbol{x}\right)\right)^{2}|X=\boldsymbol{x}\right]+... \\
&\quad 2E_{Y|X=\boldsymbol{x}}\left[\left(\overline{Y}-f\left(\boldsymbol{x}\right)\right)\left(Y-\overline{Y}\right)|X=\boldsymbol{x}\right] \\
&= E_{Y|X=\boldsymbol{x}}\left[\left(Y-\overline{Y}\right)^{2}|X=\boldsymbol{x}\right]+... \\
&\quad E_{Y|X=\boldsymbol{x}}\left[\left(\overline{Y}-f\left(\boldsymbol{x}\right)\right)^{2}|X=\boldsymbol{x}\right]+... \\
&\quad 2\left(\overline{Y}-f\left(\boldsymbol{x}\right)\right)E_{Y|X=\boldsymbol{x}}\left[\left(Y-\overline{Y}\right)|X=\boldsymbol{x}\right]
\end{aligned}
$$

We can optimize the function

By a Simple Analysis

$$
\begin{aligned}
E_{Y|X=\boldsymbol{x}}\left[\left(Y-f\left(\boldsymbol{x}\right)\right)^{2}|X=\boldsymbol{x}\right] &= E_{Y|X=\boldsymbol{x}}\left[\left(Y+\overline{Y}-\overline{Y}-f\left(\boldsymbol{x}\right)\right)^{2}|X=\boldsymbol{x}\right] \\
&= E_{Y|X=\boldsymbol{x}}\left[\left(Y-\overline{Y}\right)^{2}|X=\boldsymbol{x}\right]+... \\
&\quad E_{Y|X=\boldsymbol{x}}\left[\left(\overline{Y}-f\left(\boldsymbol{x}\right)\right)^{2}|X=\boldsymbol{x}\right]+... \\
&\quad 2E_{Y|X=\boldsymbol{x}}\left[\left(\overline{Y}-f\left(\boldsymbol{x}\right)\right)\left(Y-\overline{Y}\right)|X=\boldsymbol{x}\right] \\
&= E_{Y|X=\boldsymbol{x}}\left[\left(Y-\overline{Y}\right)^{2}|X=\boldsymbol{x}\right]+... \\
&\quad E_{Y|X=\boldsymbol{x}}\left[\left(\overline{Y}-f\left(\boldsymbol{x}\right)\right)^{2}|X=\boldsymbol{x}\right]+... \\
&\quad 2\left(\overline{Y}-f\left(\boldsymbol{x}\right)\right)E_{Y|X=\boldsymbol{x}}\left[\left(Y-\overline{Y}\right)|X=\boldsymbol{x}\right]
\end{aligned}
$$

Further, we have

We have

$$E_{Y|X=\boldsymbol{x}}\left[\left(Y-\overline{Y}\right)|X=\boldsymbol{x}\right]=E_{Y|X=\boldsymbol{x}}\left[Y\right]-E_{Y|X=\boldsymbol{x}}\left[\frac{1}{N}\sum_{i=1}^{N}Y_i\right]$$

$$=\mu_Y-\frac{1}{N}\sum_{i=1}^{N}E_{Y|X=x}\left[Y_i\right]$$

$$=\mu_Y-\frac{N\mu_Y}{N}$$

$$=0$$

Further, we have

We have

$$E_{Y|X=\boldsymbol{x}}\left[\left(Y-\overline{Y}\right)|X=\boldsymbol{x}\right]=E_{Y|X=\boldsymbol{x}}\left[Y\right]-E_{Y|X=\boldsymbol{x}}\left[\frac{1}{N}\sum_{i=1}^{N}Y_i\right]$$

$$=\mu_Y-\frac{1}{N}\sum_{i=1}^{N}E_{Y|X=\boldsymbol{x}}\left[Y_i\right]$$

$$=\mu_Y-\frac{N\mu_Y}{N}$$

$$=0$$

Further, we have

We have

$$E_{Y|X=\boldsymbol{x}}\left[\left(Y-\overline{Y}\right)|X=\boldsymbol{x}\right]=E_{Y|X=\boldsymbol{x}}\left[Y\right]-E_{Y|X=\boldsymbol{x}}\left[\frac{1}{N}\sum_{i=1}^{N}Y_i\right]$$

$$=\mu_Y-\frac{1}{N}\sum_{i=1}^{N}E_{Y|X=\boldsymbol{x}}\left[Y_i\right]$$

$$=\mu_Y-\frac{N\mu_Y}{N}$$

$$=0$$

Further, we have

We have

$$E_{Y|X=\boldsymbol{x}}\left[\left(Y-\overline{Y}\right)|X=\boldsymbol{x}\right]=E_{Y|X=\boldsymbol{x}}\left[Y\right]-E_{Y|X=\boldsymbol{x}}\left[\frac{1}{N}\sum_{i=1}^{N}Y_i\right]$$

$$=\mu_Y-\frac{1}{N}\sum_{i=1}^{N}E_{Y|X=\boldsymbol{x}}\left[Y_i\right]$$

$$=\mu_Y-\frac{N\mu_Y}{N}$$

$$=0$$

Finally

We have

$$E_{Y|X=\boldsymbol{x}}\left[\left(Y-f\left(\boldsymbol{x}\right)\right)^{2}|X=\boldsymbol{x}\right]=E_{Y|X=\boldsymbol{x}}\left[\left(Y-\overline{Y}\right)^{2}|X=\boldsymbol{x}\right]+...$$
$$E_{Y|X=\boldsymbol{x}}\left[\left(\overline{Y}-f\left(\boldsymbol{x}\right)\right)^{2}|X=\boldsymbol{x}\right]$$

Then

We have that we can optimize point-wise

Then, if we choose

$$f(X) = \overline{Y} \approx E_Y[Y|X = \boldsymbol{x}]$$

- The conditional expectation, also known as the **regression function**!!!

The variance for $\overline{Y}$ that can be approximated by

$$\frac{\sigma^2}{n}$$

Then

We have that we can optimize point-wise

Then, if we choose

$$f(X) = \overline{Y} \approx E_Y[Y|X = \boldsymbol{x}]$$

- The conditional expectation, also known as the **regression function**!!!

Additionally, we have

$$E_{Y|X=\boldsymbol{x}}\left[(Y - f(\boldsymbol{x}))^2 | X = \boldsymbol{x}\right] = E_{Y|X=\boldsymbol{x}}\left[\left(Y - \overline{Y}\right)^2 | X = \boldsymbol{x}\right]$$

The variance for Y that can be approximated by

$$\hat{\sigma}_Y^2 = \frac{1}{N-1}\sum_{i=1}^{N}\left(Y_i - \overline{Y}\right)^2$$

Then

We have that we can optimize point-wise

Then, if we choose

$$f(X) = \overline{Y} \approx E_Y[Y|X = \boldsymbol{x}]$$

- The conditional expectation, also known as the **regression function**!!!

Additionally, we have

$$E_{Y|X=\boldsymbol{x}}\left[(Y - f(\boldsymbol{x}))^2 |X = \boldsymbol{x}\right] = E_{Y|X=\boldsymbol{x}}\left[\left(Y - \overline{Y}\right)^2 |X = \boldsymbol{x}\right]$$

The variance for Y that can be approximated by

$$\hat{\sigma}_Y^2 = \frac{1}{N-1}\sum_{i=1}^{N}\left(Y_i - \overline{Y}\right)^2$$

Then

We have that we can optimize point-wise

Then, if we choose

$$f(X) = \overline{Y} \approx E_Y[Y|X = \boldsymbol{x}]$$

- The conditional expectation, also known as the **regression function**!!!

Additionally, we have

$$E_{Y|X=\boldsymbol{x}}\left[(Y - f(\boldsymbol{x}))^2 |X = \boldsymbol{x}\right] = E_{Y|X=\boldsymbol{x}}\left[\left(Y - \overline{Y}\right)^2 |X = \boldsymbol{x}\right]$$

The variance for Y that can be approximated by

$$\hat{\sigma}_Y^2 = \frac{1}{N-1}\sum_{i=1}^{N}\left(Y_i - \overline{Y}\right)^2$$

Then

We have that we can optimize point-wise

Then, if we choose

$$f(X) = \overline{Y} \approx E_Y[Y|X = \boldsymbol{x}]$$

- The conditional expectation, also known as the **regression function**!!!

Additionally, we have

$$E_{Y|X=\boldsymbol{x}}\left[(Y - f(\boldsymbol{x}))^2 |X = \boldsymbol{x}\right] = E_{Y|X=\boldsymbol{x}}\left[\left(Y - \overline{Y}\right)^2 |X = \boldsymbol{x}\right]$$

The variance for Y that can be approximated by

$$\widehat{\sigma}_Y^2 = \frac{1}{N-1}\sum_{i=1}^{N}\left(Y_i - \overline{Y}\right)^2$$

Finally

Thus, the best prediction of Y at any point $X = \boldsymbol{x}$ the regression function for LSE

- It is the conditional mean.

$$E_Y [Y|X = \boldsymbol{x}]$$

- When best is measured by average squared error.

Finally

Thus, the best prediction of Y at any point $X = \boldsymbol{x}$ the regression function for LSE

- It is the conditional mean.

$$E_Y [Y|X = \boldsymbol{x}]$$

  - When **best is measured by average squared error.**

Outline

Now Nearest Neighborhood

At each point x

The method calculates the average of all those $y_i's$ with input $\boldsymbol{x}_i = \boldsymbol{x}$

$$\frac{1}{n_{\boldsymbol{x}_i=\boldsymbol{x}}} \sum_{\boldsymbol{x}_i=\boldsymbol{x}} y_i$$

Or in other way, an estimation based in the average

$$\widehat{f}(\boldsymbol{x}) = Ave\left(y_i | \boldsymbol{x}_i \in N_k(x)\right)$$

Now Nearest Neighborhood

At each point x

The method calculates the average of all those $y_i's$ with input $\boldsymbol{x}_i = \boldsymbol{x}$

$$\frac{1}{n_{\boldsymbol{x}_i=\boldsymbol{x}}} \sum_{\boldsymbol{x}_i=\boldsymbol{x}} y_i$$

Or in other way, an estimation based in the average

$$\widehat{f}(\boldsymbol{x}) = Ave\left(y_i | \boldsymbol{x}_i \in N_k(x)\right)$$

Therefore

Two things happen here

- Expectation is approximated by averaging over sample data

$$\frac{1}{k} \sum_{\boldsymbol{x}_i \in N_k(x)} y$$

Thus, conditioning

- It is relaxing to some region "close" to the target point

Therefore

Two things happen here

- Expectation is approximated by averaging over sample data

$$\frac{1}{k} \sum_{\boldsymbol{x}_i \in N_k(x)} y$$

Thus, conditioning

- It is relaxing to some region **"close"** to the target point

Therefore

For large training sample size N

- The points in the neighborhood are likely to be close to $\boldsymbol{x}$.
  - Then as k gets large the average will get more stable.

It is more under regularity conditions on $P(X, Y)$

- One can for that as $N \to \infty$ and $k \to \infty$ such that $k/N \to 0$

$$\widehat{f}(\boldsymbol{x}) \to E(Y|X = \boldsymbol{x})$$

Problem

We often do not have very large number of samples!!!

Therefore

For large training sample size N

- The points in the neighborhood are likely to be close to $\boldsymbol{x}$.
  - Then as k gets large the average will get more stable.

It is more under regularity conditions on $P(X, Y)$

- One can for that as $N \to \infty$ and $k \to \infty$ such that $k/N \to 0$

$$\widehat{f}(\boldsymbol{x}) \to E(Y|X = \boldsymbol{x})$$

Problem

We often do not have very large number of samples!!!

Therefore

For large training sample size N

- The points in the neighborhood are likely to be close to $\boldsymbol{x}$.
  - Then as k gets large the average will get more stable.

It is more under regularity conditions on $P(X, Y)$

- One can for that as $N \to \infty$ and $k \to \infty$ such that $k/N \to 0$

$$\widehat{f}(\boldsymbol{x}) \to E(Y|X = \boldsymbol{x})$$

Problem

We often do not have very large number of samples!!!

However

As the dimension d gets large

Thus, the metric size of the k-nearest neighborhood also gets larger.

Making

$$\widehat{f}(\boldsymbol{x}) \to E(Y|X=\boldsymbol{x})$$

It fails miserably.

However

As the dimension d gets large

Thus, the metric size of the k-nearest neighborhood also gets larger.

Making

$$\widehat{f}(\boldsymbol{x}) \to E(Y|X = \boldsymbol{x})$$

It fails miserably.

Outline

How does Linear Regression fit into this framework?

The regression function $f(\boldsymbol{x})$ is approximately linear in its arguments

$$f(\boldsymbol{x}) = \boldsymbol{x}^T \boldsymbol{w}$$

How does Linear Regression fit into this framework?

The regression function $f(\boldsymbol{x})$ is approximately linear in its arguments

$$f(\boldsymbol{x}) = \boldsymbol{x}^T \boldsymbol{w}$$

Plugging this linear model for $f(\boldsymbol{x})$ into EPE and differentiating

$$\boldsymbol{w} = \left[E\left(XX^T\right)\right]^{-1} E(XY)$$

How does Linear Regression fit into this framework?

The regression function $f(x)$ is approximately linear in its arguments

$$f(\boldsymbol{x}) = \boldsymbol{x}^T \boldsymbol{w}$$

Plugging this linear model for $f(x)$ into EPE and differentiating

$$\boldsymbol{w} = \left[E\left(XX^T\right)\right]^{-1} E(XY)$$

Note

- Note we have not conditioned on X.
- We have used our knowledge of the functional relationship.
  - for pooling over values of X.

How does Linear Regression fit into this framework?

The regression function $f(x)$ is approximately linear in its arguments

$$f(\boldsymbol{x}) = \boldsymbol{x}^T \boldsymbol{w}$$

Plugging this linear model for $f(x)$ into EPE and differentiating

$$\boldsymbol{w} = \left[E\left(XX^T\right)\right]^{-1} E(XY)$$

Note

- Note we have not conditioned on X.
- We have used our knowledge of the functional relationship.
  - for pooling over values of X.

How does Linear Regression fit into this framework?

The regression function $f(\boldsymbol{x})$ is approximately linear in its arguments

$$f(\boldsymbol{x}) = \boldsymbol{x}^T \boldsymbol{w}$$

Plugging this linear model for $f(\boldsymbol{x})$ into EPE and differentiating

$$\boldsymbol{w} = \left[E\left(XX^T\right)\right]^{-1} E(XY)$$

Note

- Note we have not conditioned on X.
- We have used our knowledge of the functional relationship.
  - for pooling over values of X.

Therefore

The least squares solution

- It amounts to replacing the expectation in

$$\boldsymbol{w} = \left[E\left(XX^T\right)\right]^{-1} E\left(XY\right)$$

by averages over the training data.

Then, we have that

k-nearest neighbors and least squares end up approximating conditional expectations by averages.

Therefore

The least squares solution

- It amounts to replacing the expectation in

$$\boldsymbol{w} = \left[E\left(XX^T\right)\right]^{-1} E\left(XY\right)$$

by averages over the training data.

Then, we have that

k-nearest neighbors and least squares end up approximating conditional expectations by averages.

Therefore

We have the following differences

- Least squares assumes $f(x)$ is well approximated by a globally linear function.
- k-nearest neighbors assumes $f(x)$ is well approximated by a locally constant function.

Outline

Some Times

We take the following assumption about the data

$$Y = f(X) + \epsilon$$

Where

- The Random Error has $E[\epsilon] = 0$
- And the error is independent of X

Under this model, we have already a solution

$$f(\boldsymbol{x}) = E[Y|X = \boldsymbol{x}]$$

The conditional distribution $P(Y|X)$ depends on X

- Only through the conditional mean $f(\boldsymbol{x})$

Some Times

We take the following assumption about the data

$$Y = f(X) + \epsilon$$

Where

- The Random Error has $E[\epsilon] = 0$
- And the error is independent of X

Under this model, we have already a solution

$$f(\boldsymbol{x}) = E[Y|X = \boldsymbol{x}]$$

The conditional distribution $P(Y|X)$ depends on X

- Only through the conditional mean $f(\boldsymbol{x})$

Some Times

We take the following assumption about the data

$$Y = f(X) + \epsilon$$

Where

- The Random Error has $E[\epsilon] = 0$
- And the error is independent of X

Under this model, we have already a solution

$$f(\boldsymbol{x}) = E[Y|X = \boldsymbol{x}]$$

The conditional distribution $P(Y|X)$ depends on X

- Only through the conditional mean $f(\boldsymbol{x})$

This is quite useful

Given that in most systems, the input-output pairs (X, Y)

- It will not have a deterministic relationship $Y = f(X)$

Nevertheless

- There will be other non measured variables that also contribute to Y

For example

- Error in the measurement of the system error!!!

This is quite useful

Given that in most systems, the input-output pairs (X, Y)

- It will not have a deterministic relationship $Y = f(X)$

Nevertheless

- There will be other non measured variables that also contribute to Y

For example

- Error in the measurement of the system error!!!

This is quite useful

Given that in most systems, the input-output pairs (X, Y)

- It will not have a deterministic relationship $Y = f(X)$

Nevertheless

- There will be other non measured variables that also contribute to Y

For example

- Error in the measurement of the system error!!!

Therefore

It is natural to use

- Least Squares as a data criterion for model estimation!!!

Additionally, we can modify the independence assuming

$$Var\left(Y|X=x\right)=\sigma\left(x\right)$$

Then

- Both the mean and variance depend on X

Therefore

It is natural to use

- Least Squares as a data criterion for model estimation!!!

Additionally, we can modify the independence assuming

$$Var\left(Y|X=\boldsymbol{x}\right)=\sigma\left(\boldsymbol{x}\right)$$

Then

- Both the mean and variance depend on X

Therefore

It is natural to use

- Least Squares as a data criterion for model estimation!!!

Additionally, we can modify the independence assuming

$$Var\left(Y|X=\boldsymbol{x}\right)=\sigma\left(\boldsymbol{x}\right)$$

Then

- Both the mean and variance depend on X

However

In general the conditional distribution $P(Y|X)$

- It can depend on X in complicated ways... and thus, the simplification models!!!

Outline

Now

Given the model $Y = f(X) + \epsilon$

- Supervised Learning tries to learn f by data from a teacher.

Now

Given the model $Y = f(X) + \epsilon$

- Supervised Learning tries to learn f by data from a teacher.

Thus

- It is necessary to observe the system
- Collect data from it!!!
- Assemble a training set of observations

$$\mathcal{D} = \{(x_i, y_i) | i = 1, 2, \ldots, N\}$$

Now

Given the model $Y = f(X) + \epsilon$

- Supervised Learning tries to learn f by data from a teacher.

Thus

- It is necessary to observe the system
- Collect data from it!!!
- Assemble a training set of observations

$$\mathcal{D} = \{(x_i, y_i) | i = 1, 2, \ldots, N\}$$

Now

Given the model $Y = f(X) + \epsilon$

- Supervised Learning tries to learn f by data from a teacher.

Thus

- It is necessary to observe the system
- Collect data from it!!!
- Assemble a training set of observations

$$\mathcal{D} = \{(x_i, y_i) \,|i = 1, 2, \ldots, N\}$$

Then

This training set is feed into a learning algorithm

This system produces an output

$$\widehat{f}(x_i)$$

Something Notable

The Learning algorithm has the ability to modify its input/output relationship $\widehat{f}$ based on the difference $y_i - f(x_i)$.

This is similar to function Approximation

- At Applied Mathematics and Statistics the input $\mathcal{D}$ are viewed as points in $(d+1)-$dimensional space.

Then

This training set is feed into a learning algorithm

This system produces an output

$$\widehat{f}(x_i)$$

Something Notable

The Learning algorithm has the ability to modify its input/output relationship $\widehat{f}$ based on the difference $y_i - f(x_i)$.

This is similar to function Approximation

- At Applied Mathematics and Statistics the input $\mathcal{D}$ are viewed as points in $(d+1)-$dimensional space.

Then

This training set is feed into a learning algorithm

This system produces an output

$$\widehat{f}(x_i)$$

Something Notable

The Learning algorithm has the ability to modify its input/output relationship $\widehat{f}$ based on the difference $y_i - f(x_i)$.

This is similar to function Approximation

- At Applied Mathematics and Statistics the input $\mathcal{D}$ are viewed as points in $(d+1)-$dimensional space

Outline

The function $f(x)$

Domain

- The domain of a function is the complete set of possible values of the independent variable.
- In our case, the $d-$dimensional subspace.

The function $f(x)$

Domain

- The domain of a function is the complete set of possible values of the independent variable.
- In our case, the $d-$dimensional subspace.

Range

- The range of a function is the complete set of all possible resulting values of the dependent variable.
- In our case, the output of $y_i's$ of our training data set.

The function $f(x)$

Domain

- The domain of a function is the complete set of possible values of the independent variable.
- In our case, the $d-$dimensional subspace.

Range

- The range of a function is the complete set of all possible resulting values of the dependent variable.
- In our case, the output of $y_i's$ of our training data set.

That, we relate by the following function

$$y_i = f(x_i) + \epsilon_i$$

Assuming linear additivity structure between noise input and outputs.

The function $f(x)$

Domain

- The domain of a function is the complete set of possible values of the independent variable.
- In our case, the $d-$dimensional subspace.

Range

- The range of a function is the complete set of all possible resulting values of the dependent variable.
- In our case, the output of $y_i's$ of our training data set.

That, we relate by the following function

$$y_i = f(x_i) + \epsilon_i$$

Assuming linear additivity structure between noise input and outputs.

The function $f(x)$

Domain

- The domain of a function is the complete set of possible values of the independent variable.
- In our case, the $d-$dimensional subspace.

Range

- The range of a function is the complete set of all possible resulting values of the dependent variable.
- In our case, the output of $y_i's$ of our training data set.

That, we relate by the following function

$$y_i = f(x_i) + \epsilon_i$$

Assuming linear additivity structure between noise input and outputs.

The Final Goal

Something Notable

- It is to obtain a useful approximation (fitting) to $f(x)$ for all x in some region of $\mathbb{R}^d$, given the representations in $\mathcal{D}$.

You can think as no so glamorous than the learning paradigm

- But using this approach, we can use all the tools generated in the last 200 years for function approximation!!!

Basically

- We can see Supervised Learning as a controlled over-fitting!!!

The Final Goal

Something Notable

- It is to obtain a useful approximation (fitting) to $f(x)$ for all x in some region of $\mathbb{R}^d$, given the representations in $\mathcal{D}$.

You can think as no so glamorous than the learning paradigm

- But using this approach, we can use all the tools generated in the last 200 years for function approximation!!!

Basically

- We can see Supervised Learning as a controlled over-fitting!!!

The Final Goal

Something Notable

- It is to obtain a useful approximation (fitting) to $f(x)$ for all x in some region of $\mathbb{R}^d$, given the representations in $\mathcal{D}$.

You can think as no so glamorous than the learning paradigm

- But using this approach, we can use all the tools generated in the last 200 years for function approximation!!!

Basically

- We can see Supervised Learning as a controlled over-fitting!!!

Outline

Parameters in the Approximations

For example, in the linear model $f(x) = \boldsymbol{x}^T w$

- There is a parameter for approximation $\theta = w$

In another example, using linear basis expansion

$$f_\theta(\boldsymbol{x}) = \sum_{k=1}^{K} h_k(\boldsymbol{x}) \theta_k$$

Traditional examples of these functions

- $x_1^2, x_1 x_2^2, \cos(x_1)$
- An also

$$h_k(\boldsymbol{x}) = \frac{1}{1 + \exp\{-\boldsymbol{x}^T \theta_k\}}$$

Parameters in the Approximations

For example, in the linear model $f(x) = \boldsymbol{x}^T w$

- There is a parameter for approximation $\theta = w$

In another example, using linear basis expansion

$$f_\theta(\boldsymbol{x}) = \sum_{k=1}^{K} h_k(\boldsymbol{x}) \theta_k$$

Traditional examples of these functions

- $x_1^2, x_1 x_2^2, \cos(x_1)$
- An also

$$h_k(\boldsymbol{x}) = \frac{1}{1 + \exp\{-\boldsymbol{x}^T \theta_k\}}$$

Parameters in the Approximations

For example, in the linear model $f(x) = \boldsymbol{x}^T w$

- There is a parameter for approximation $\theta = w$

In another example, using linear basis expansion

$$f_\theta(\boldsymbol{x}) = \sum_{k=1}^{K} h_k(\boldsymbol{x}) \theta_k$$

Traditional examples of these functions

- $x_1^2, x_1 x_2^2, \cos(x_1)$
- An also

$$h_k(\boldsymbol{x}) = \frac{1}{1 + \exp\{-\boldsymbol{x}^T \theta_k\}}$$

Outline

Residual Sum of Squares (RSS)

Here, the general structure for the RSS(f) under a Penalty/Regularization

$$PRSS\left(f,\lambda\right)=RSS\left(f\right)+\lambda J\left(f\right)$$

For Example, we have Ridge Regression

$$\sum_{i=1}^{N}\left(y_i-\boldsymbol{x}^T\right)^2+\lambda\sum_{i=1}^{d}w_i^2 \tag{1}$$

Residual Sum of Squares (RSS)

Here, the general structure for the RSS(f) under a Penalty/Regularization

$$PRSS\left(f,\lambda\right)=RSS\left(f\right)+\lambda J\left(f\right)$$

For Example, we have Ridge Regression

$$\sum_{i=1}^{N}\left(y_i-\boldsymbol{x}^T\right)^2+\lambda\sum_{i=1}^{d}w_i^2 \tag{1}$$

Outline

Kernel Methods

You can think on these methods as

- They try to estimate the regression function or conditional expectation by specifying:
  - The properties of the local Neighborhood,
  - The class of regular functions fitted locally.

Kernel Methods

You can think on these methods as

- They try to estimate the regression function or conditional expectation by specifying:
  - The properties of the local Neighborhood,
  - The class of regular functions fitted locally.

For this, they use kernels as

$$K_\lambda(x, x_0) = \frac{1}{\lambda} \exp\left\{ \frac{\|x - x_0\|^2}{2\lambda} \right\}$$

Kernel Methods

You can think on these methods as

- They try to estimate the regression function or conditional expectation by specifying:
  - The properties of the local Neighborhood,
  - The class of regular functions fitted locally.

For this, they use kernels as

$$K_\lambda(\boldsymbol{x}, \boldsymbol{x}_0) = \frac{1}{\lambda} \exp\left\{ -\frac{\|\boldsymbol{x} - \boldsymbol{x}_0\|^2}{2\lambda} \right\}$$

What happens here?

We have the following

$$K_\lambda(\boldsymbol{x}, \boldsymbol{x}_0) = \frac{1}{\lambda} \exp\left\{-\frac{(\boldsymbol{x}-\boldsymbol{x}_0)^2}{2\lambda}\right\} \qquad K_\lambda(\boldsymbol{x}, \boldsymbol{x}_0) = \frac{1}{\lambda} \exp\left\{-\frac{|\boldsymbol{x}-\boldsymbol{x}_0|}{2\lambda}\right\}$$

As in Regression

We can define a way of doing estimation

$$RSS\left(f_{\boldsymbol{w}}, \boldsymbol{x}_0\right) = \sum_{i=1}^{N} K_\lambda\left(\boldsymbol{x}_i, \boldsymbol{x}_0\right)\left(y_i - f_{\boldsymbol{w}}\left(\boldsymbol{x}_i\right)\right)^2$$

As in Regression

We can define a way of doing estimation

$$RSS\left(f_{\boldsymbol{w}}, \boldsymbol{x}_0\right) = \sum_{i=1}^{N} K_\lambda\left(\boldsymbol{x}_i, \boldsymbol{x}_0\right)\left(y_i - f_{\boldsymbol{w}}\left(\boldsymbol{x}_i\right)\right)^2$$

Where $f_{\boldsymbol{w}}$

1. $f_{\boldsymbol{w}}\left(\boldsymbol{x}\right) = w_0$ the constant function (Nadaraya–Watson Estimate).
2. $f_{\boldsymbol{w}}\left(\boldsymbol{x}\right) = \sum_{i=0}^{d} x_i w_i$ the classic local linear regression models.

As in Regression

We can define a way of doing estimation

$$RSS\left(f_{\boldsymbol{w}}, \boldsymbol{x}_0\right) = \sum_{i=1}^{N} K_\lambda\left(\boldsymbol{x}_i, \boldsymbol{x}_0\right)\left(y_i - f_{\boldsymbol{w}}\left(\boldsymbol{x}_i\right)\right)^2$$

Where $f_{\boldsymbol{w}}$

1. $f_{\boldsymbol{w}}\left(\boldsymbol{x}\right) = w_0$ the constant function (Nadaraya–Watson Estimate).
2. $f_{\boldsymbol{w}}\left(\boldsymbol{x}\right) = \sum_{i=0}^{d} x_i w_i$ the classic local linear regression models.

For Example

Nearest-Neighbor Methods

It can be thought as a kernel method with a data dependent metric:

$$K_k(\boldsymbol{x}, \boldsymbol{x}_0) = I\left[\|\boldsymbol{x} - \boldsymbol{x}_0\| \leq \left\|\boldsymbol{x}_{(i)} - \boldsymbol{x}_0\right\| | i = 1, 2, \ldots, k\right]$$

For Example

Nearest-Neighbor Methods

It can be thought as a kernel method with a data dependent metric:

$$K_k(\boldsymbol{x}, \boldsymbol{x}_0) = I\left[\|\boldsymbol{x} - \boldsymbol{x}_0\| \leq \left\|\boldsymbol{x}_{(i)} - \boldsymbol{x}_0\right\| | i = 1, 2, \ldots, k\right]$$

Where

- $\boldsymbol{x}_{(i)}$ is the training observation ranked i^{th} in distance from $\boldsymbol{x}_0$.
- $I(S)$ is the indicator of the set S.

For Example

Nearest-Neighbor Methods

It can be thought as a kernel method with a data dependent metric:

$$K_k(\boldsymbol{x}, \boldsymbol{x}_0) = I\left[\|\boldsymbol{x} - \boldsymbol{x}_0\| \leq \left\|\boldsymbol{x}_{(i)} - \boldsymbol{x}_0\right\| | i = 1, 2, \ldots, k\right]$$

Where

- $\boldsymbol{x}_{(i)}$ is the training observation ranked i^{th} in distance from $\boldsymbol{x}_0$.
- $I(S)$ is the indicator of the set S.

Outline

A more wide variety of flexible models

For Example, Linear and Polynomial Expansions

$$f_{\boldsymbol{w}}(\boldsymbol{x}) = \sum_{m=1}^{M} w_m h_m(\boldsymbol{x})$$

A more wide variety of flexible models

For Example, Linear and Polynomial Expansions

$$f_{\boldsymbol{w}}(\boldsymbol{x}) = \sum_{m=1}^{M} w_m h_m(\boldsymbol{x})$$

Where

- h_m is a function on $\boldsymbol{x}$.
- with the linear term w_m acting on the function h_m

A more wide variety of flexible models

For Example, Linear and Polynomial Expansions

$$f_{\boldsymbol{w}}(\boldsymbol{x}) = \sum_{m=1}^{M} w_m h_m(\boldsymbol{x})$$

Where

- h_m is a function on $\boldsymbol{x}$.
- with the linear term w_m acting on the function h_m

A more wide variety of flexible models

For Example, Linear and Polynomial Expansions

$$f_{\boldsymbol{w}}(\boldsymbol{x}) = \sum_{m=1}^{M} w_m h_m(\boldsymbol{x})$$

Where

- h_m is a function on $\boldsymbol{x}$.
- with the linear term w_m acting on the function h_m

Other Examples

Something Notable

- Tensor products of spline bases can be used for inputs with dimensions larger than one - CART and MARS models

Radial basis functions

$$f_{\boldsymbol{w}}(\boldsymbol{x}) = \sum_{m=1}^{M} w_m K_{\lambda_m}(\mu_m, \boldsymbol{x}) \text{ with } K_{\lambda}(\mu, \boldsymbol{x}) = \exp\left\{\frac{\|\boldsymbol{x} - \mu\|^2}{2\lambda}\right\}$$

A single-layer feed-forward neural network

$$f_{\boldsymbol{w}}(\boldsymbol{x}) = \sum_{m=1}^{M} w_m S\left(\alpha_m^T \boldsymbol{x} + b_m\right) \text{ with } S(y) = \frac{1}{1 + \exp\{-y\}}$$

Other Examples

Something Notable

- Tensor products of spline bases can be used for inputs with dimensions larger than one - CART and MARS models

Radial basis functions

$$f_{\boldsymbol{w}}(\boldsymbol{x}) = \sum_{m=1}^{M} w_m K_{\lambda_m}(\mu_m, \boldsymbol{x}) \text{ with } K_\lambda(\mu, \boldsymbol{x}) = \exp\left\{-\frac{\|\boldsymbol{x}-\mu\|^2}{2\lambda}\right\}$$

A single-layer feed-forward neural network

$$f_{\boldsymbol{w}}(\boldsymbol{x}) = \sum_{m=1}^{M} w_m S\left(\alpha_m^T \boldsymbol{x} + b_m\right) \text{ with } S(y) = \frac{1}{1+\exp\{-y\}}$$

Other Examples

Something Notable

- Tensor products of spline bases can be used for inputs with dimensions larger than one - CART and MARS models

Radial basis functions

$$f_{\boldsymbol{w}}(\boldsymbol{x}) = \sum_{m=1}^{M} w_m K_{\lambda_m}(\mu_m, \boldsymbol{x}) \text{ with } K_\lambda(\mu, \boldsymbol{x}) = \exp\left\{-\frac{\|\boldsymbol{x}-\mu\|^2}{2\lambda}\right\}$$

A single-layer feed-forward neural network

$$f_{\boldsymbol{w}}(\boldsymbol{x}) = \sum_{m=1}^{M} w_m S\left(\alpha_m^T \boldsymbol{x} + b_m\right) \text{ with } S(y) = \frac{1}{1+\exp\{-y\}}$$

Outline

Conclusions

Machine Learning is a quite wide and vast field

- It requires Time
- It requires Effort
- It can be sometimes hard!!!

Conclusions

Machine Learning is a quite wide and vast field

- It requires Time
- It requires Effort
- It can be sometimes hard!!!

This is the main reason of this class

- To take step by step into such interesting field as Machine Learning!!!

Conclusions

Machine Learning is a quite wide and vast field

- It requires Time
- It requires Effort
- It can be sometimes hard!!!

This is the main reason of this class

- To take step by step into such interesting field as Machine Learning!!!

Thank you for being passengers

- An Future Pilots in this class!!!

Conclusions

Machine Learning is a quite wide and vast field

- It requires Time
- It requires Effort
- It can be sometimes hard!!!

This is the main reason of this class

- To take step by step into such interesting field as Machine Learning!!!

Thank you for being passengers

- An Future Pilots in this class!!!

Conclusions

Machine Learning is a quite wide and vast field

- It requires Time
- It requires Effort
- It can be sometimes hard!!!

This is the main reason of this class

- To take step by step into such interesting field as Machine Learning!!!

Thank you for being passengers

- An Future Pilots in this class!!!