Linear Regression

Karl Stratos

June 18, 2018

The Regression Problem

- **Problem.** Find a desired input-output mapping $f : \mathcal{X} \to \mathbb{R}$ where the output is a real value.

$x =$  $\Longrightarrow$ $y = 0.1^\circ$

"How much should I turn my handle, given the environment?"

The Regression Problem

- **Problem.** Find a desired input-output mapping $f : \mathcal{X} \to \mathbb{R}$ where the output is a real value.

$x =$  $\Longrightarrow$ $y = 0.1^\circ$

"How much should I turn my handle, given the environment?"

- Today's focus: *data-driven* approach to regression

Overview

- Approaches to the Regression Problem
  - Not Data-Driven
  - Data-Driven: Nonparameteric
  - Data-Driven: Parameteric
- Linear Regression (a Parameteric Approach)
  - Model and Objective
  - Parameter Estimation
  - Generalization to Multi-Dimensional Input
- Polynomial Regression

Running Example: Predict Weight from Height

- Suppose we want a regression model $f : \mathcal{X} \rightarrow \mathbb{R}$ that predicts weight (in pounds) from height (in inches).
  - What is the input space?

Running Example: Predict Weight from Height

- Suppose we want a regression model $f : \mathcal{X} \to \mathbb{R}$ that predicts weight (in pounds) from height (in inches).
  - What is the input space? $\mathcal{X} = \mathbb{R}$

Running Example: Predict Weight from Height

- Suppose we want a regression model $f : \mathcal{X} \to \mathbb{R}$ that predicts weight (in pounds) from height (in inches).
  - What is the input space? $\mathcal{X} = \mathbb{R}$

- Naive approach: stipulate **rules**.
  - If $x \in [0, 30)$, then predict $y = 50$.
  - If $x \in [30, 60)$, then predict $y = 80$.
  - If $x \in [60, 70)$, then predict $y = 150$.
  - If $x \geq 70$, then predict $y = 200$.

Running Example: Predict Weight from Height

- Suppose we want a regression model $f : \mathcal{X} \to \mathbb{R}$ that predicts weight (in pounds) from height (in inches).
  - What is the input space? $\mathcal{X} = \mathbb{R}$

- Naive approach: stipulate **rules**.
  - If $x \in [0, 30)$, then predict $y = 50$.
  - If $x \in [30, 60)$, then predict $y = 80$.
  - If $x \in [60, 70)$, then predict $y = 150$.
  - If $x \geq 70$, then predict $y = 200$.

- Pro: Immediately programmable

Running Example: Predict Weight from Height

- Suppose we want a regression model $f : \mathcal{X} \to \mathbb{R}$ that predicts weight (in pounds) from height (in inches).
  - What is the input space? $\mathcal{X} = \mathbb{R}$

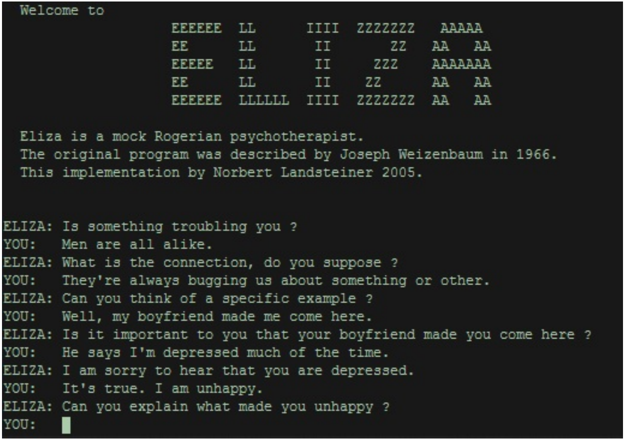
- Naive approach: stipulate **rules**.
  - If $x \in [0, 30)$, then predict $y = 50$.
  - If $x \in [30, 60)$, then predict $y = 80$.
  - If $x \in [60, 70)$, then predict $y = 150$.
  - If $x \geq 70$, then predict $y = 200$.

- Pro: Immediately programmable

- **Cons**: "Uninformed", requires labor-intensive domain-specific rule engineering
  - There is no learning from data (on the machine's part).

Before We Move on to Data-Driven Approaches

Rule-based solutions can go surpringly far.



```
  Welcome to
                    EEEEEE  LL      IIII  ZZZZZZZ   AAAAA
                    EE      LL       II       ZZ   AA   AA
                    EEEEE   LL       II     ZZZ    AAAAAAA
                    EE      LL       II    ZZ      AA   AA
                    EEEEEE  LLLLLL  IIII  ZZZZZZZ  AA   AA

  Eliza is a mock Rogerian psychotherapist.
  The original program was described by Joseph Weizenbaum in 1966.
  This implementation by Norbert Landsteiner 2005.


ELIZA: Is something troubling you ?
YOU:   Men are all alike.
ELIZA: What is the connection, do you suppose ?
YOU:   They're always bugging us about something or other.
ELIZA: Can you think of a specific example ?
YOU:   Well, my boyfriend made me come here.
ELIZA: Is it important to you that your boyfriend made you come here ?
YOU:   He says I'm depressed much of the time.
ELIZA: I am sorry to hear that you are depressed.
YOU:   It's true. I am unhappy.
ELIZA: Can you explain what made you unhappy ?
YOU:   █
```

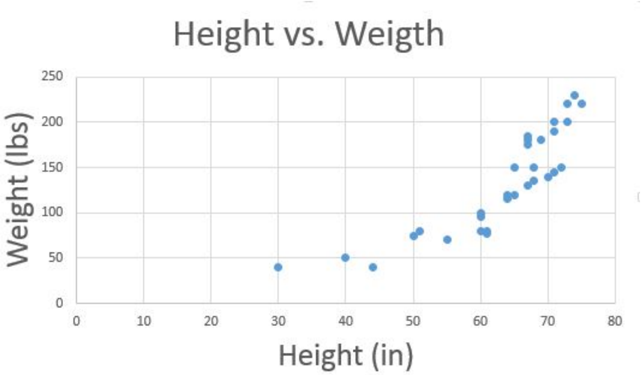
ELIZA: a conversation program from the 60s

Overview

- Approaches to the Regression Problem
  - Not Data-Driven
  - Data-Driven: Nonparameteric
  - Data-Driven: Parameteric
- Linear Regression (a Parameteric Approach)
  - Model and Objective
  - Parameter Estimation
  - Generalization to Multi-Dimensional Input
- Polynomial Regression

Data

- A set of n height-weight pairs $(x^{(1)}, y^{(1)}) \dots (x^{(n)}, y^{(n)}) \in \mathbb{R} \times \mathbb{R}$



Q. How can we use this data to obtain a weight predictor?

Simple Data-Specific Rules

Store all n data points in a dictionary $D(x^{(i)}) = y^{(i)}$.

1. Predict by memorization ("rote learning"):

$$f(x) = \begin{cases} D(x) & \text{if } x \in D \\ ? & \text{otherwise} \end{cases}$$

2. Or slightly better, predict by nearest neighbor search:

$$f(x) = D\left(\arg\min_{i=1}^{n} \left|\left|x - x^{(i)}\right|\right|\right)$$

Nonparametric Models

- These are simplest instances of **nonparameteric** models.
  - It just means that the model doesn't have any associated parameters before seeing the data.

- Pro: Adapts to data without assuming anything about a given problem, achieving better "coverage" with more data

- **Cons**
  - Not scalable: need to store the entire data
  - Issues with "overfitting": model excessively dependent on data, generalizing to new instances can be difficult.

Overview

Parameteric Models

- Dominant approach in machine learning.
- Assumes a fixed form of model f_θ defined by a set of **parameters** θ.

Parameteric Models

- Dominant approach in machine learning.
- Assumes a fixed form of model f_θ defined by a set of **parameters** θ.
- The parameters θ are learned from data $S = \left\{ (x^{(i)}, y^{(i)}) \right\}_{i=1}^{n}$ by optimizing a data-dependent "**objective**" or "**loss**" function $J_S(\theta) \in \mathbb{R}$.

Parameteric Models

- Dominant approach in machine learning.

- Assumes a fixed form of model f_θ defined by a set of **parameters** θ.

- The parameters θ are learned from data $S = \left\{ (x^{(i)}, y^{(i)}) \right\}_{i=1}^{n}$ by optimizing a data-dependent "**objective**" or "**loss**" function $J_S(\theta) \in \mathbb{R}$.

- Optimizing J_S wrt. parameter θ is the learning problem!

$$\theta^* = \arg\min_\theta J_S(\theta)$$

Parameteric Models

- Dominant approach in machine learning.
- Assumes a fixed form of model f_θ defined by a set of **parameters** θ.
- The parameters θ are learned from data $S = \left\{\left(x^{(i)}, y^{(i)}\right)\right\}_{i=1}^{n}$ by optimizing a data-dependent "**objective**" or "**loss**" function $J_S(\theta) \in \mathbb{R}$.
- Optimizing J_S wrt. parameter θ is the learning problem!

$$\theta^* = \arg\min_\theta J_S(\theta)$$

- Today: Focus on a simplest parametric model called **linear regression**.

Overview

- Approaches to the Regression Problem
  - Not Data-Driven
  - Data-Driven: Nonparameteric
  - Data-Driven: Parameteric
- Linear Regression (a Parameteric Approach)
  - Model and Objective
  - Parameter Estimation
  - Generalization to Multi-Dimensional Input
- Polynomial Regression

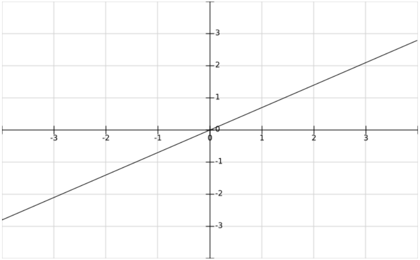
Linear Regression Model

- Model parameter: $w \in \mathbb{R}$

- Model definition:

$$f_w(x) := wx$$

- Defines a line with slope w



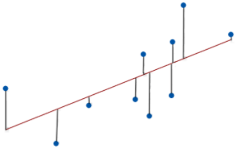
- Goal: learn w from data $S = \left\{\left(x^{(i)}, y^{(i)}\right)\right\}_{i=1}^{n}$
  - Need a data-dependent objective function $J_S(w)$

Least Squares Objective

- **Least squares objective**: minimize

$$J_S^{\mathrm{LS}}(w) := \sum_{i=1}^{n} \left(y^{(i)} - w x^{(i)} \right)^2$$

- Idea: fit a line on the training data by reducing the sum of squared residuals



Overview

- Approaches to the Regression Problem
  - Not Data-Driven
  - Data-Driven: Nonparameteric
  - Data-Driven: Parameteric
- Linear Regression (a Parameteric Approach)
  - Model and Objective
  - Parameter Estimation
  - Generalization to Multi-Dimensional Input
- Polynomial Regression

The Learning Problem

- Solve for the scalar

$$w_S^{\mathrm{LS}} = \arg\min_{w \in \mathbb{R}} \underbrace{\sum_{i=1}^{n} \left(y^{(i)} - w x^{(i)} \right)^2}_{J_S^{\mathrm{LS}}(w)}$$

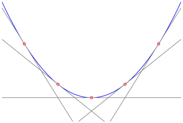
The Learning Problem

- Solve for the scalar

$$w_S^{\mathrm{LS}} = \arg\min_{w \in \mathbb{R}} \underbrace{\sum_{i=1}^{n} \left(y^{(i)} - w x^{(i)} \right)^2}_{J_S^{\mathrm{LS}}(w)}$$

- The objective $J_S^{\mathrm{LS}}(w)$ is strongly convex in w (unless all $x^{(i)} = 0$), thus the global minimum is uniquely achieved by w_S^{LS} satisfying

$$\left. \frac{\partial J_S^{\mathrm{LS}}(w)}{\partial w} \right|_{w = w_S^{\mathrm{LS}}} = 0$$



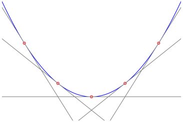
The Learning Problem

- Solve for the scalar

$$w_S^{\mathrm{LS}} = \arg\min_{w \in \mathbb{R}} \underbrace{\sum_{i=1}^{n} \left(y^{(i)} - w x^{(i)} \right)^2}_{J_S^{\mathrm{LS}}(w)}$$

- The objective $J_S^{\mathrm{LS}}(w)$ is strongly convex in w (unless all $x^{(i)} = 0$), thus the global minimum is uniquely achieved by w_S^{LS} satisfying

$$\left. \frac{\partial J_S^{\mathrm{LS}}(w)}{\partial w} \right|_{w = w_S^{\mathrm{LS}}} = 0$$



- Solving this system yields the close-form expression:

$$w_S^{\mathrm{LS}} = \frac{\sum_{i=1}^{n} x^{(i)} y^{(i)}}{\sum_{i=1}^{n} (x^{(i)})^2}$$

Overview

Approaches to the Regression Problem

- Not Data-Driven
- Data-Driven: Nonparameteric
- Data-Driven: Parameteric

Linear Regression (a Parameteric Approach)

- Model and Objective
- Parameter Estimation
- Generalization to Multi-Dimensional Input

Polynomial Regression

Linear Regression with Multi-Dimensional Input

- Input $\boldsymbol{x} \in \mathbb{R}^d$ is now a vector of d **features** $x_1 \ldots x_d \in \mathbb{R}$.

$x_1 = 65$ (height)
$x_2 = 29$ (age)
$x_3 = 1$ (male indicator)
$x_4 = 0$ (female indicator)

$\Longrightarrow$ $y = 140$ (pounds)

Linear Regression with Multi-Dimensional Input

- Input $\boldsymbol{x} \in \mathbb{R}^d$ is now a vector of d **features** $x_1 \ldots x_d \in \mathbb{R}$.

$$\begin{array}{l} x_1 = 65 \text{ (height)} \\ x_2 = 29 \text{ (age)} \\ x_3 = 1 \text{ (male indicator)} \\ x_4 = 0 \text{ (female indicator)} \end{array} \qquad \Longrightarrow \qquad y = 140 \text{ (pounds)}$$

- Model: $\boldsymbol{w} \in \mathbb{R}^d$ defining

$$\begin{aligned} f_{\boldsymbol{w}}(\boldsymbol{x}) &:= \boldsymbol{w} \cdot \boldsymbol{x} = \boldsymbol{w}^\top \boldsymbol{x} = \langle \boldsymbol{w}, \boldsymbol{x} \rangle \\ &= w_1 x_1 + \cdots + w_d x_d \end{aligned}$$

Linear Regression with Multi-Dimensional Input

- Input $\boldsymbol{x} \in \mathbb{R}^d$ is now a vector of d **features** $x_1 \ldots x_d \in \mathbb{R}$.

$$\begin{aligned} x_1 &= 65 \text{ (height)} \\ x_2 &= 29 \text{ (age)} \\ x_3 &= 1 \text{ (male indicator)} \\ x_4 &= 0 \text{ (female indicator)} \end{aligned} \qquad \Longrightarrow \qquad y = 140 \text{ (pounds)}$$

- Model: $\boldsymbol{w} \in \mathbb{R}^d$ defining

$$\begin{aligned} f_{\boldsymbol{w}}(\boldsymbol{x}) &:= \boldsymbol{w} \cdot \boldsymbol{x} = \boldsymbol{w}^\top \boldsymbol{x} = \langle \boldsymbol{w}, \boldsymbol{x} \rangle \\ &= w_1 x_1 + \cdots + w_d x_d \end{aligned}$$

- Least squares objective: exactly the same. Assume $n \geq d$!

$$J_S^{\text{LS}}(\boldsymbol{w}) = \sum_{i=1}^{n} \left(\underbrace{y^{(i)}}_{\mathbb{R}} - \underbrace{\boldsymbol{w} \cdot \boldsymbol{x}^{(i)}}_{\mathbb{R}} \right)^2$$

The Learning Problem

- Solve for the vector

$$\boldsymbol{w}_S^{\mathrm{LS}} = \operatorname*{arg\,min}_{\boldsymbol{w}\in\mathbb{R}^d} \sum_{i=1}^{n} \left(y^{(i)} - \boldsymbol{w} \cdot \boldsymbol{x}^{(i)} \right)^2 = \operatorname*{arg\,min}_{\boldsymbol{w}\in\mathbb{R}^d} ||\boldsymbol{y} - X\boldsymbol{w}||_2^2$$

where $\boldsymbol{y}_i = y^{(i)} \in \mathbb{R}$ and $X \in \mathbb{R}^{n\times d}$ has rows $\boldsymbol{x}^{(i)}$.

The Learning Problem

- Solve for the vector

$$\boldsymbol{w}_S^{\mathrm{LS}} = \operatorname*{arg\,min}_{\boldsymbol{w} \in \mathbb{R}^d} \sum_{i=1}^{n} \left(y^{(i)} - \boldsymbol{w} \cdot \boldsymbol{x}^{(i)} \right)^2 = \operatorname*{arg\,min}_{\boldsymbol{w} \in \mathbb{R}^d} \; ||\boldsymbol{y} - X\boldsymbol{w}||_2^2$$

  where $\boldsymbol{y}_i = y^{(i)} \in \mathbb{R}$ and $X \in \mathbb{R}^{n \times d}$ has rows $\boldsymbol{x}^{(i)}$.

- $||\boldsymbol{y} - X\boldsymbol{w}||_2^2$ is strongly convex in $\boldsymbol{w}$ (unless rank$(X) < d$), thus the global minimum is uniquely achieved by $\boldsymbol{w}_S^{\mathrm{LS}}$ satisfying

$$\left. \frac{\partial \, ||\boldsymbol{y} - X\boldsymbol{w}||_2^2}{\partial \boldsymbol{w}} \right|_{\boldsymbol{w} = \boldsymbol{w}_S^{\mathrm{LS}}} = \mathbf{0}_{d \times 1}$$

The Learning Problem

- Solve for the vector

  $$\boldsymbol{w}_S^{\mathrm{LS}} = \operatorname*{arg\,min}_{\boldsymbol{w}\in\mathbb{R}^d} \sum_{i=1}^{n} \left(y^{(i)} - \boldsymbol{w} \cdot \boldsymbol{x}^{(i)} \right)^2 = \operatorname*{arg\,min}_{\boldsymbol{w}\in\mathbb{R}^d} ||\boldsymbol{y} - X\boldsymbol{w}||_2^2$$

  where $\boldsymbol{y}_i = y^{(i)} \in \mathbb{R}$ and $X \in \mathbb{R}^{n\times d}$ has rows $\boldsymbol{x}^{(i)}$.

- $||\boldsymbol{y} - X\boldsymbol{w}||_2^2$ is strongly convex in $\boldsymbol{w}$ (unless $\mathrm{rank}(X) < d$), thus the global minimum is uniquely achieved by $\boldsymbol{w}_S^{\mathrm{LS}}$ satisfying

  $$\left.\frac{\partial\, ||\boldsymbol{y} - X\boldsymbol{w}||_2^2}{\partial \boldsymbol{w}}\right|_{\boldsymbol{w}=\boldsymbol{w}_S^{\mathrm{LS}}} = \mathbf{0}_{d\times 1}$$

- Solving this system yields the close-form expression:

  $$\boldsymbol{w}_S^{\mathrm{LS}} = (X^\top X)^{-1} X^\top \boldsymbol{y} = X^+ \boldsymbol{y}$$

Overview

- Approaches to the Regression Problem
  - Not Data-Driven
  - Data-Driven: Nonparameteric
  - Data-Driven: Parameteric
- Linear Regression (a Parameteric Approach)
  - Model and Objective
  - Parameter Estimation
  - Generalization to Multi-Dimensional Input
- Polynomial Regression

Fitting a Polynomial: 1-Dimensional Input

- In linear regression with scalar input, we learn the slope $w \in \mathbb{R}$ of a line such that $y \approx wx$.

Fitting a Polynomial: 1-Dimensional Input

- In linear regression with scalar input, we learn the slope $w \in \mathbb{R}$ of a line such that $y \approx wx$.
- In **polynomial regression**, we learn the coefficients $w_1 \dots w_p, w_{p+1} \in \mathbb{R}$ of a polynomial of degree p such that

$$y \approx w_1 x^p + \cdots + w_p x + \underbrace{w_{p+1}}_{\text{bias term}}$$

Fitting a Polynomial: 1-Dimensional Input

- In linear regression with scalar input, we learn the slope $w \in \mathbb{R}$ of a line such that $y \approx wx$.

- In **polynomial regression**, we learn the coefficients $w_1 \dots w_p, w_{p+1} \in \mathbb{R}$ of a polynomial of degree p such that

$$y \approx w_1 x^p + \cdots + w_p x + \underbrace{w_{p+1}}_{\text{bias term}}$$

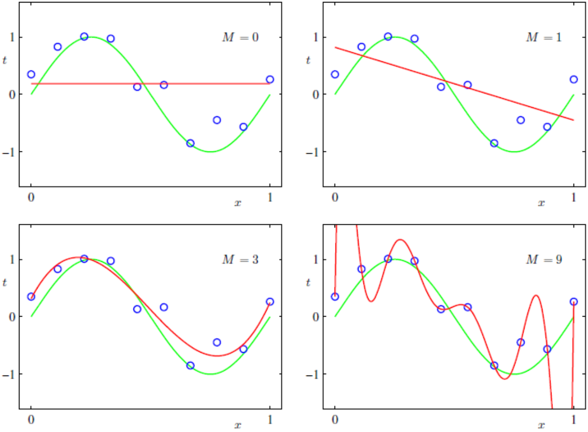
- How? Upon receiving input x, apply **polynomial feature expansion** to calculate a *new* representation of x:

$$x \mapsto \begin{bmatrix} x^p \\ \vdots \\ x \\ 1 \end{bmatrix}$$

  Follow by linear regression with $(p+1)$-dimensional input.

Degree of Polynomial = Model Complexity

- $p = 0$: Fit a bias term
- $p = 1$: Fit a slope and a bias term (i.e., an affine function)
- $p = 2$: Learn a quadratic function
- $\ldots$



https://machinelearningac.wordpress.com/2011/09/15/model-selection-and-the-triple-tradeof

Polynomial Regression with Multi-Dimensional Input

Example: $p = 2$

$$\begin{bmatrix} x_1 \\ \vdots \\ x_d \end{bmatrix} \mapsto \begin{bmatrix} x_1^2 \\ \vdots \\ x_d^2 \\ x_1 x_2 \\ \vdots \\ x_d x_{d-1} \\ x_1 \\ \vdots \\ x_d \\ 1 \end{bmatrix}$$

In general: time to calculate feature expansion $O(d^p)$ is exponential in p.

Summary

- **Regression** is the problem of learning a real-valued mapping $f : \mathcal{X} \to \mathbb{R}$.
- **Linear regressor** is a simplest parametric model that uses parameter $\boldsymbol{w} \in \mathbb{R}^d$ to define $f_{\boldsymbol{w}}(\boldsymbol{x}) = \boldsymbol{w} \cdot \boldsymbol{x}$.
- Fitting a linear regressor on a dataset by a **least squares objective** so easy that it has a closed-form solution.
- **Polynomial regression**: feature expansion followed by linear regression

Last Remarks

- What if we have a model/objective such that training doesn't have a closed-form solution?
- Instead of manually fixing dictating the input representation (e.g., a polynomial of degree 3), can we automatically learn a good *representation function* $\phi(\boldsymbol{x})$ as part of optimization?
- We will answer these questions later in the course (hint: gradient descent, neural networks).