# Regression

## CISC 7026: Introduction to Deep Learning

University of Macau

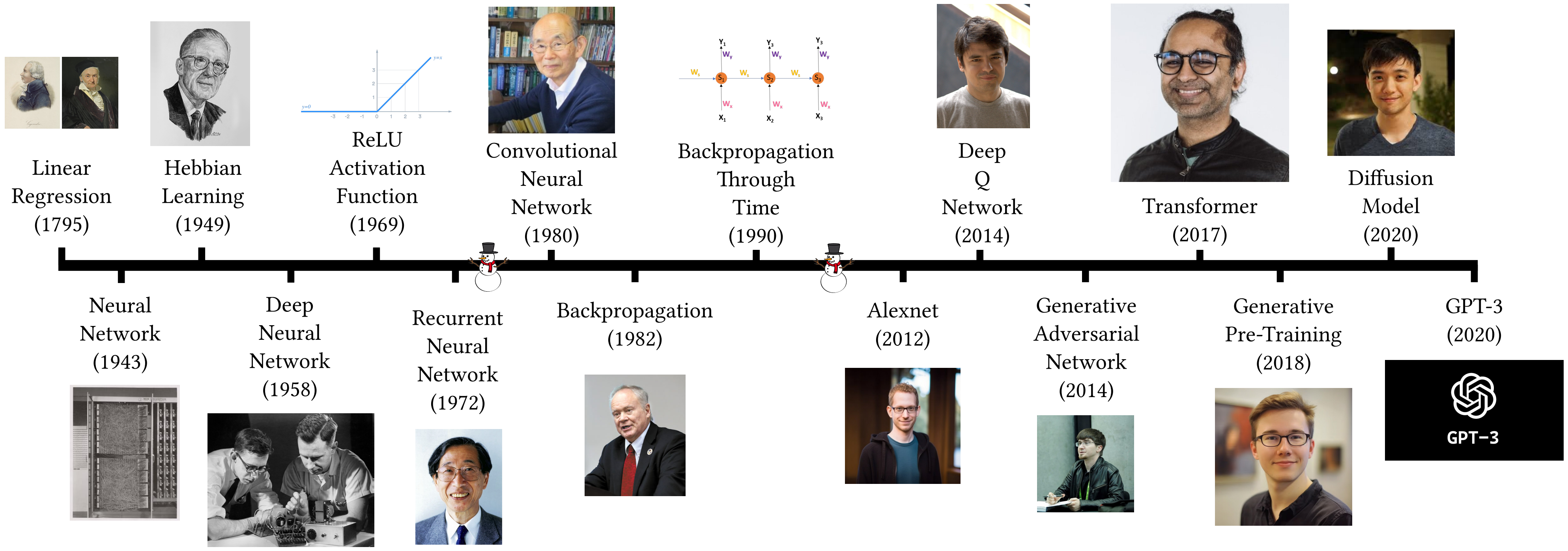
# Today, we will learn about linear regression

Today, we will learn about linear regression

Probably the oldest method for machine learning (Gauss and Legendre)

Today, we will learn about linear regression

Probably the oldest method for machine learning (Gauss and Legendre)



# ML

Many problems in ML can be reduced to **regression** or **classification**

# ML

Many problems in ML can be reduced to **regression** or **classification**

**Regression** asks how many

Many problems in ML can be reduced to **regression** or **classification**

**Regression** asks how many

- How much money will I make?

Many problems in ML can be reduced to **regression** or **classification**

**Regression** asks how many

- How much money will I make?
- How much rain will there be tomorrow?

Many problems in ML can be reduced to **regression** or **classification**

**Regression** asks how many

- How much money will I make?
- How much rain will there be tomorrow?
- How far away is this object?

# ML

Many problems in ML can be reduced to **regression** or **classification**

**Regression** asks how many

- How much money will I make?
- How much rain will there be tomorrow?
- How far away is this object?

**Classification** asks which one

Many problems in ML can be reduced to **regression** or **classification**

**Regression** asks how many

- How much money will I make?
- How much rain will there be tomorrow?
- How far away is this object?

**Classification** asks which one

- Is this a dog or muffin?

# ML

Many problems in ML can be reduced to **regression** or **classification**

**Regression** asks how many

- How much money will I make?
- How much rain will there be tomorrow?
- How far away is this object?

**Classification** asks which one

- Is this a dog or muffin?
- Will it rain tomorrow? Yes or no?

# ML

Many problems in ML can be reduced to **regression** or **classification**

**Regression** asks how many

- How much money will I make?
- How much rain will there be tomorrow?
- How far away is this object?

**Classification** asks which one

- Is this a dog or muffin?
- Will it rain tomorrow? Yes or no?
- What color is this object?

# ML

Many problems in ML can be reduced to **regression** or **classification**

**Regression** asks how many
- How much money will I make?
- How much rain will there be tomorrow?
- How far away is this object?

**Classification** asks which one
- Is this a dog or muffin?
- Will it rain tomorrow? Yes or no?
- What color is this object?

Let us start with regression

# Linear Regression

Today, we will come up with a regression problem and then solve it!

# Linear Regression

Today, we will come up with a regression problem and then solve it!

1. Define an example problem

# Linear Regression

Today, we will come up with a regression problem and then solve it!

1. Define an example problem
2. Define our machine learning model $f$

# Linear Regression

Today, we will come up with a regression problem and then solve it!

1. Define an example problem
2. Define our machine learning model $f$
3. Define a loss function $\mathcal{L}$

# Linear Regression

Today, we will come up with a regression problem and then solve it!

1. Define an example problem
2. Define our machine learning model $f$
3. Define a loss function $\mathcal{L}$
4. Use $\mathcal{L}$ to learn the parameters $\theta$ of $f$

# Linear Regression

1. **Define an example problem**
2. Define our machine learning model $f$
3. Define a loss function $\mathcal{L}$
4. Use $\mathcal{L}$ to learn the parameters $\theta$ of $f$

# Linear Regression

The World Health Organization (WHO) has collected data on life expectancy

# Linear Regression

The World Health Organization (WHO) has collected data on life expectancy


Available for free at https://www.who.int/data/gho/data/themes/mortality-and-global-health-estimates/ghe-life-expectancy-and-healthy-life-expectancy

# Linear Regression

The WHO collected data from roughly 3,000 people from 193 countries

# Linear Regression

The WHO collected data from roughly 3,000 people from 193 countries

For each person, they recorded:

# Linear Regression

The WHO collected data from roughly 3,000 people from 193 countries

For each person, they recorded:

- Home country

# Linear Regression

The WHO collected data from roughly 3,000 people from 193 countries

For each person, they recorded:

- Home country
- Alcohol consumption

# Linear Regression

The WHO collected data from roughly 3,000 people from 193 countries

For each person, they recorded:

- Home country
- Alcohol consumption
- Education

# Linear Regression

The WHO collected data from roughly 3,000 people from 193 countries

For each person, they recorded:

- Home country
- Alcohol consumption
- Education
- Gross domestic product (GDP) of the country

# Linear Regression

The WHO collected data from roughly 3,000 people from 193 countries

For each person, they recorded:

- Home country
- Alcohol consumption
- Education
- Gross domestic product (GDP) of the country
- Immunizations for Measles and Hepatitis B

# Linear Regression

The WHO collected data from roughly 3,000 people from 193 countries

For each person, they recorded:

- Home country
- Alcohol consumption
- Education
- Gross domestic product (GDP) of the country
- Immunizations for Measles and Hepatitis B
- How long this person lived

# Linear Regression

The WHO collected data from roughly 3,000 people from 193 countries

For each person, they recorded:

- Home country
- Alcohol consumption
- Education
- Gross domestic product (GDP) of the country
- Immunizations for Measles and Hepatitis B
- How long this person lived

We can use this data to make future predictions

# Linear Regression

Since everyone here is very educated, we will focus on how education affects life expectancy

# Linear Regression

Since everyone here is very educated, we will focus on how education affects life expectancy

There are studies showing a causal effect on education on health

# Linear Regression

Since everyone here is very educated, we will focus on how education affects life expectancy

There are studies showing a causal effect on education on health

- *The causal effects of education on health outcomes in the UK Biobank.* Davies et al. *Nature Human Behaviour.*

# Linear Regression

Since everyone here is very educated, we will focus on how education affects life expectancy

There are studies showing a causal effect on education on health

- *The causal effects of education on health outcomes in the UK Biobank.* Davies et al. *Nature Human Behaviour.*
- By staying in school, you are likely to live longer

# Linear Regression

**Task:** Given your education, predict your life expectancy

# Linear Regression

**Task:** Given your education, predict your life expectancy

$X \in \mathbb{R}_+$ : Years in school

# Linear Regression

**Task:** Given your education, predict your life expectancy

$X \in \mathbb{R}_+$ : Years in school

$Y \in \mathbb{R}_+$ : Age of death

# Linear Regression

**Task:** Given your education, predict your life expectancy

$X \in \mathbb{R}_+$ : Years in school

$Y \in \mathbb{R}_+$ : Age of death

**Approach:** Learn the parameters $\theta$ such that

$$f(x, \theta) = y; \quad x \in X, y \in Y$$

# Linear Regression

**Task:** Given your education, predict your life expectancy

$X \in \mathbb{R}_+$ : Years in school

$Y \in \mathbb{R}_+$ : Age of death

**Approach:** Learn the parameters $\theta$ such that

$$f(x, \theta) = y; \quad x \in X, y \in Y$$

**Goal:** Given someone's education, predict how long they will live

# Linear Regression

1. **Define an example problem**
2. Define our machine learning model $f$
3. Define a loss function $\mathcal{L}$
4. Use $\mathcal{L}$ to learn the parameters $\theta$ of $f$

# Linear Regression

1. Define an example problem
2. **Define our machine learning model** $f$
3. Define a loss function $\mathcal{L}$
4. Use $\mathcal{L}$ to learn the parameters $\theta$ of $f$

# Linear Regression

Soon, $f$ will be a deep neural network

# Linear Regression

Soon, $f$ will be a deep neural network

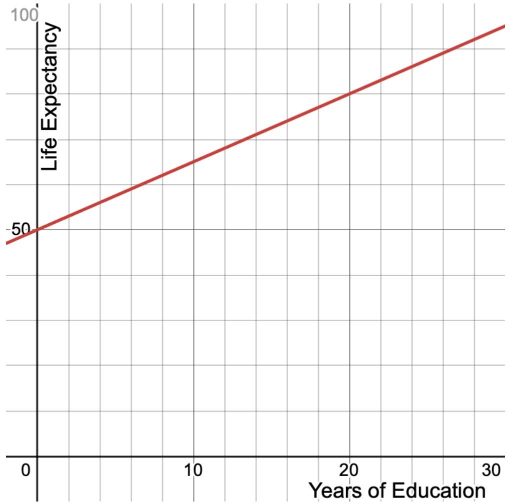
For now, it is easier if we make $f$ a **linear function**

# Linear Regression

Soon, $f$ will be a deep neural network

For now, it is easier if we make $f$ a **linear function**

$$f(x, \boldsymbol{\theta}) = f\left(x, \begin{bmatrix} \theta_1 \\ \theta_0 \end{bmatrix}\right) = \theta_1 x + \theta_0$$

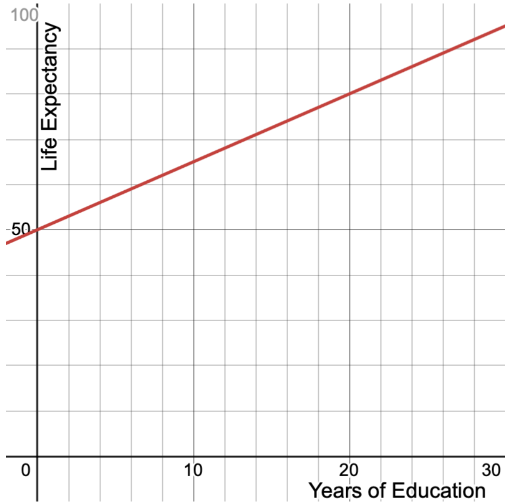


# Linear Regression

Soon, $f$ will be a deep neural network

For now, it is easier if we make $f$ a **linear function**

$$f(x, \boldsymbol{\theta}) = f\left(x, \begin{bmatrix} \theta_1 \\ \theta_0 \end{bmatrix}\right) = \theta_1 x + \theta_0$$




Now, we need to find the parameters $\boldsymbol{\theta} = \begin{bmatrix} \theta_1 \\ \theta_0 \end{bmatrix}$ that makes $f(x, \boldsymbol{\theta}) = y$

# Linear Regression

1. Define an example problem
2. **Define our machine learning model** $f$
3. Define a loss function $\mathcal{L}$
4. Use $\mathcal{L}$ to learn the parameters $\theta$ of $f$

# Linear Regression

1. Define an example problem
2. Define our machine learning model $f$
3. **Define a loss function $\mathcal{L}$**
4. Use $\mathcal{L}$ to learn the parameters $\theta$ of $f$

# Linear Regression

Now, we need to find the parameters $\boldsymbol{\theta} = \begin{bmatrix} \theta_1 \\ \theta_0 \end{bmatrix}$ that make $f(x, \boldsymbol{\theta}) = y$

# Linear Regression

Now, we need to find the parameters $\boldsymbol{\theta} = \begin{bmatrix} \theta_1 \\ \theta_0 \end{bmatrix}$ that make $f(x, \boldsymbol{\theta}) = y$

**Question:** How do we find $\boldsymbol{\theta}$? (Hint: We want $f(x, \boldsymbol{\theta}) = y$)

# Linear Regression

Now, we need to find the parameters $\boldsymbol{\theta} = \begin{bmatrix} \theta_1 \\ \theta_0 \end{bmatrix}$ that make $f(x, \boldsymbol{\theta}) = y$

**Question:** How do we find $\boldsymbol{\theta}$? (Hint: We want $f(x, \boldsymbol{\theta}) = y$)

**Answer:** We will minimize the **loss** (error) between $f(x, \boldsymbol{\theta})$ and $y$, for all

$$x \in X, y \in Y$$

# Linear Regression

We compute the loss using the **loss function** $\mathcal{L} : X \times Y \times \Theta \mapsto \mathbb{R}$

# Linear Regression

We compute the loss using the **loss function** $\mathcal{L} : X \times Y \times \Theta \mapsto \mathbb{R}$

$$\mathcal{L}(x, y, \boldsymbol{\theta})$$

# Linear Regression

We compute the loss using the **loss function** $\mathcal{L} : X \times Y \times \Theta \mapsto \mathbb{R}$

$$\mathcal{L}(x, y, \boldsymbol{\theta})$$

The loss function tells us how close $f(x, \boldsymbol{\theta})$ is to $y$

# Linear Regression

We compute the loss using the **loss function** $\mathcal{L} : X \times Y \times \Theta \mapsto \mathbb{R}$

$$\mathcal{L}(x, y, \boldsymbol{\theta})$$

The loss function tells us how close $f(x, \boldsymbol{\theta})$ is to $y$

By **minimizing** the loss function, we make $f(x, \boldsymbol{\theta}) = y$

# Linear Regression

We compute the loss using the **loss function** $\mathcal{L} : X \times Y \times \Theta \mapsto \mathbb{R}$

$$\mathcal{L}(x, y, \boldsymbol{\theta})$$

The loss function tells us how close $f(x, \boldsymbol{\theta})$ is to $y$

By **minimizing** the loss function, we make $f(x, \boldsymbol{\theta}) = y$

There are many possible loss functions, but for regression we often use the **square error**

# Linear Regression

We compute the loss using the **loss function** $\mathcal{L} : X \times Y \times \Theta \mapsto \mathbb{R}$

$$\mathcal{L}(x, y, \boldsymbol{\theta})$$

The loss function tells us how close $f(x, \boldsymbol{\theta})$ is to $y$

By **minimizing** the loss function, we make $f(x, \boldsymbol{\theta}) = y$

There are many possible loss functions, but for regression we often use the **square error**

$$\text{error}(y, \hat{y}) = (y - \hat{y})^2$$

# Linear Regression

Let's derive the error function

# Linear Regression

Let's derive the error function

$$f(x, \boldsymbol{\theta}) = y$$

f(x) should predict y

# Linear Regression

Let's derive the error function

$$f(x, \boldsymbol{\theta}) = y$$

f(x) should predict y

$$f(x, \boldsymbol{\theta}) - y = 0$$

Move y to LHS

# Linear Regression

Let's derive the error function

| | |
|---|---|
| $f(x, \boldsymbol{\theta}) = y$ | f(x) should predict y |
| $f(x, \boldsymbol{\theta}) - y = 0$ | Move y to LHS |
| $(f(x, \boldsymbol{\theta}) - y)^2 = 0$ | Square for minimization |

# Linear Regression

Let's derive the error function

$$f(x, \boldsymbol{\theta}) = y$$

f(x) should predict y

$$f(x, \boldsymbol{\theta}) - y = 0$$

Move y to LHS

$$(f(x, \boldsymbol{\theta}) - y)^2 = 0$$

Square for minimization

$$\text{error}(f(x, \boldsymbol{\theta}), y) = (f(x, \boldsymbol{\theta}) - y)^2$$

# Linear Regression

We can write the loss function for a single datapoint $x_i, y_i$ as

$$\mathcal{L}(x_i, y_i, \boldsymbol{\theta}) = \text{error}(f(x_i, \boldsymbol{\theta}), y_i) = (f(x_i, \boldsymbol{\theta}) - y_i)^2$$

# Linear Regression

We can write the loss function for a single datapoint $x_i, y_i$ as

$$\mathcal{L}(x_i, y_i, \boldsymbol{\theta}) = \text{error}(f(x_i, \boldsymbol{\theta}), y_i) = (f(x_i, \boldsymbol{\theta}) - y_i)^2$$

We want to find the parameters $\boldsymbol{\theta}$ that minimize $\mathcal{L}$

# Linear Regression

We can write the loss function for a single datapoint $x_i, y_i$ as

$$\mathcal{L}(x_i, y_i, \boldsymbol{\theta}) = \mathrm{error}(f(x_i, \boldsymbol{\theta}), y_i) = (f(x_i, \boldsymbol{\theta}) - y_i)^2$$

We want to find the parameters $\boldsymbol{\theta}$ that minimize $\mathcal{L}$

$$\arg\min_{\boldsymbol{\theta}} \mathcal{L}(x_i, y_i, \boldsymbol{\theta}) = \arg\min_{\boldsymbol{\theta}} \mathrm{error}(f(x_i, \boldsymbol{\theta}), y_i) = \arg\min_{\boldsymbol{\theta}} (f(x_i, \boldsymbol{\theta}) - y_i)^2$$

# Linear Regression

We can write the loss function for a single datapoint $x_i, y_i$ as

$$\mathcal{L}(x_i, y_i, \boldsymbol{\theta}) = \text{error}(f(x_i, \boldsymbol{\theta}), y_i) = (f(x_i, \boldsymbol{\theta}) - y_i)^2$$

We want to find the parameters $\boldsymbol{\theta}$ that minimize $\mathcal{L}$

$$\arg\min_{\boldsymbol{\theta}} \mathcal{L}(x_i, y_i, \boldsymbol{\theta}) = \arg\min_{\boldsymbol{\theta}} \text{error}(f(x_i, \boldsymbol{\theta}), y_i) = \arg\min_{\boldsymbol{\theta}} (f(x_i, \boldsymbol{\theta}) - y_i)^2$$

**Question:** Any issues with $\mathcal{L}$? Will it give us a good prediction for all $x$?

# Linear Regression

We can write the loss function for a single datapoint $x_i, y_i$ as

$$\mathcal{L}(x_i, y_i, \boldsymbol{\theta}) = \text{error}(f(x_i, \boldsymbol{\theta}), y_i) = (f(x_i, \boldsymbol{\theta}) - y_i)^2$$

We want to find the parameters $\boldsymbol{\theta}$ that minimize $\mathcal{L}$

$$\arg\min_{\boldsymbol{\theta}} \mathcal{L}(x_i, y_i, \boldsymbol{\theta}) = \arg\min_{\boldsymbol{\theta}} \text{error}(f(x_i, \boldsymbol{\theta}), y_i) = \arg\min_{\boldsymbol{\theta}} (f(x_i, \boldsymbol{\theta}) - y_i)^2$$

**Question:** Any issues with $\mathcal{L}$? Will it give us a good prediction for all $x$?

**Answer:** We only consider a single datapoint! We want to learn $\boldsymbol{\theta}$ for the entire dataset

# Linear Regression

For a single $x_i, y_i$:

$$\arg\min_{\boldsymbol{\theta}} \mathcal{L}(x_i, y_i, \boldsymbol{\theta}) = \arg\min_{\boldsymbol{\theta}} \text{error}(f(x_i, \boldsymbol{\theta}), y_i) = \arg\min_{\boldsymbol{\theta}} (f(x_i, \boldsymbol{\theta}) - y_i)^2$$

# Linear Regression

For a single $x_i, y_i$:

$$\arg\min_{\boldsymbol{\theta}} \mathcal{L}(x_i, y_i, \boldsymbol{\theta}) = \arg\min_{\boldsymbol{\theta}} \operatorname{error}(f(x_i, \boldsymbol{\theta}), y_i) = \arg\min_{\boldsymbol{\theta}} (f(x_i, \boldsymbol{\theta}) - y_i)^2$$

For the entire dataset:

$$\boldsymbol{x} = \begin{bmatrix} x_1 & x_2 & \dots & x_n \end{bmatrix}^\top, \boldsymbol{y} = \begin{bmatrix} y_1 & y_2 & \dots & y_n \end{bmatrix}^\top$$

$$\arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\theta}) = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{n} \operatorname{error}(f(x_i, \boldsymbol{\theta}), y_i) = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{n} (f(x_i, \boldsymbol{\theta}) - y_i)$$

# Linear Regression

For a single $x_i, y_i$:

$$\arg\min_{\boldsymbol{\theta}} \mathcal{L}(x_i, y_i, \boldsymbol{\theta}) = \arg\min_{\boldsymbol{\theta}} \mathrm{error}(f(x_i, \boldsymbol{\theta}), y_i) = \arg\min_{\boldsymbol{\theta}} (f(x_i, \boldsymbol{\theta}) - y_i)^2$$

For the entire dataset:

$$\boldsymbol{x} = \begin{bmatrix} x_1 & x_2 & \dots & x_n \end{bmatrix}^\top, \boldsymbol{y} = \begin{bmatrix} y_1 & y_2 & \dots & y_n \end{bmatrix}^\top$$

$$\arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\theta}) = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{n} \mathrm{error}(f(x_i, \boldsymbol{\theta}), y_i) = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{n} (f(x_i, \boldsymbol{\theta}) - y_i)$$

Minimizing this loss function will give us the optimal parameters!

# Linear Regression

1. Define an example problem
2. Define our machine learning model $f$
3. **Define a loss function $\mathcal{L}$**
4. Use $\mathcal{L}$ to learn the parameters $\theta$ of $f$

# Linear Regression

1. Define an example problem
2. Define our machine learning model $f$
3. Define a loss function $\mathcal{L}$
4. **Use $\mathcal{L}$ to learn the parameters $\theta$ of $f$**

# Linear Regression

**Question:** How do we minimize:

$$\arg\min_{\boldsymbol{\theta}} \mathcal{L}(x_i, y_i, \boldsymbol{\theta}) = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{n} \text{error}(f(x_i, \boldsymbol{\theta}), y_i) = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{n} (f(x_i, \boldsymbol{\theta}) - y_i$$

# Linear Regression

**Question:** How do we minimize:

$$\text{g} \min_{\boldsymbol{\theta}} \mathcal{L}(x_i, y_i, \boldsymbol{\theta}) = \arg \min_{\boldsymbol{\theta}} \sum_{i=1}^{n} \text{error}(f(x_i, \boldsymbol{\theta}), y_i) = \arg \min_{\boldsymbol{\theta}} \sum_{i=1}^{n} (f(x_i, \boldsymbol{\theta}) - y_i$$

**Answer:** For now, magic! We need more knowledge before we can derive this.

# Linear Regression

First, we will construct a **design matrix** $\boldsymbol{X}_D$ containing input data $x$

# Linear Regression

First, we will construct a **design matrix** $\boldsymbol{X}_D$ containing input data $x$

$$\boldsymbol{X}_D = \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{bmatrix}$$

# Linear Regression

We add the column of ones so that we can multiply $\boldsymbol{X}_D^\top$ with $\boldsymbol{\theta}$ to get a linear function $\theta_1 x + \theta_0$ evaluated at each data point

$$\boldsymbol{X}_D \boldsymbol{\theta} = \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{bmatrix} \begin{bmatrix} \theta_1 \\ \theta_0 \end{bmatrix} = \begin{bmatrix} \theta_1 x_1 + \theta_0 \\ \theta_1 x_2 + \theta_0 \\ \vdots \\ \theta_1 x_n + \theta_0 \end{bmatrix}$$

# Linear Regression

With our design matrix $\boldsymbol{X}_D$ and desired output $\boldsymbol{y}$,

$$\boldsymbol{X}_D = \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{bmatrix}, \boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

and our parameters $\boldsymbol{\theta}$,

$$\boldsymbol{\theta} = \begin{bmatrix} \theta_1 \\ \theta_0 \end{bmatrix},$$

$$\boldsymbol{\theta} = \left(\boldsymbol{X}_D^\top \boldsymbol{X}_D\right)^{-1} \boldsymbol{X}_D^\top \boldsymbol{y}$$

We can find the parameters that minimize $\mathcal{L}$

# Linear Regression

1. Define an example problem
2. Define our machine learning model $f$
3. Define a loss function $\mathcal{L}$
4. **Use $\mathcal{L}$ to learn the parameters $\theta$ of $f$**

# Linear Regression

1. Define an example problem
2. Define our machine learning model $f$
3. Define a loss function $\mathcal{L}$
4. Use $\mathcal{L}$ to learn the parameters $\theta$ of $f$

# Example

Back to the example...

# Example

Back to the example...

**Task:** Given your education, predict your life expectancy

# Example

Back to the example...

**Task:** Given your education, predict your life expectancy

$X \in \mathbb{R}_+$ : Years in school

# Example

Back to the example...

**Task:** Given your education, predict your life expectancy

$X \in \mathbb{R}_+$ : Years in school

$Y \in \mathbb{R}_+$ : Age of death

# Example

Back to the example...

**Task:** Given your education, predict your life expectancy

$X \in \mathbb{R}_+$ : Years in school

$Y \in \mathbb{R}_+$ : Age of death

**Approach:** Learn the parameters $\theta$ such that

$$f(x, \theta) = y; \quad x \in X, y \in Y$$

# Example

Back to the example...

**Task:** Given your education, predict your life expectancy

$X \in \mathbb{R}_+$ : Years in school

$Y \in \mathbb{R}_+$ : Age of death

**Approach:** Learn the parameters $\theta$ such that

$$f(x, \theta) = y; \quad x \in X, y \in Y$$

**Goal:** Given someone's education, predict how long they will live

# Example

Back to the example...

**Task:** Given your education, predict your life expectancy

$X \in \mathbb{R}_+$ : Years in school

$Y \in \mathbb{R}_+$ : Age of death

**Approach:** Learn the parameters $\theta$ such that

$$f(x, \theta) = y; \quad x \in X, y \in Y$$

**Goal:** Given someone's education, predict how long they will live

You will be doing this in your first assignment!

# Tips for assignment 1

# Tips for assignment 1

```python
def f(theta, design):
  # Linear function
  return design @ theta
```

# Tips for assignment 1

```python
def f(theta, design):
  # Linear function
  return design @ theta
```

Not all matrices can be inverted! Ensure the matrices are square and the condition number is low

```python
A.shape
cond = jax.numpy.linalg.cond(A)
```

## Tips for assignment 1

```python
def f(theta, design):
  # Linear function
  return design @ theta
```

Not all matrices can be inverted! Ensure the matrices are square and the condition number is low

```python
A.shape
cond = jax.numpy.linalg.cond(A)
```

Everything you need is in the lecture notes

# Linear Regression

1. Define an example problem
2. Define our machine learning model $f$
3. Define a loss function $\mathcal{L}$
4. Use $\mathcal{L}$ to learn the parameters $\theta$ of $f$

# Relax

# Example

**Task:** Given your education, predict your life expectancy

# Example

**Task:** Given your education, predict your life expectancy

Plot the datapoints $(x_1, y_1), (x_2, y_2), \ldots$

# Example

**Task:** Given your education, predict your life expectancy

Plot the datapoints $(x_1, y_1), (x_2, y_2), ...$

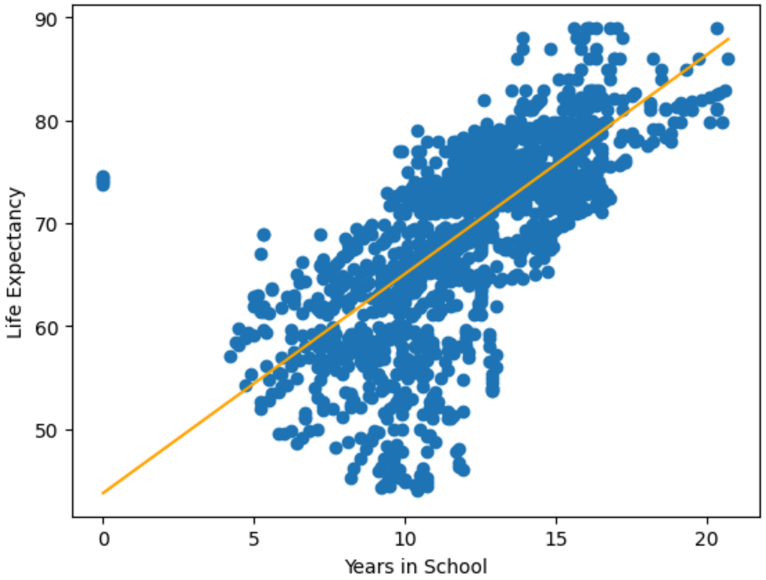
Plot the curve $f(x, \boldsymbol{\theta}) = \theta_1 x + \theta_0; \quad x \in [0, 25]$

# Example

**Task:** Given your education, predict your life expectancy

Plot the datapoints $(x_1, y_1), (x_2, y_2), ...$

Plot the curve $f(x, \boldsymbol{\theta}) = \theta_1 x + \theta_0; \quad x \in [0, 25]$

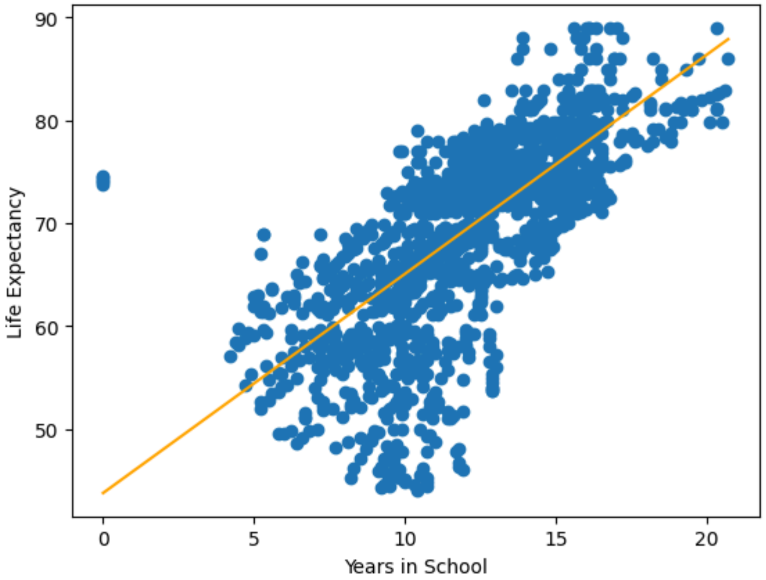


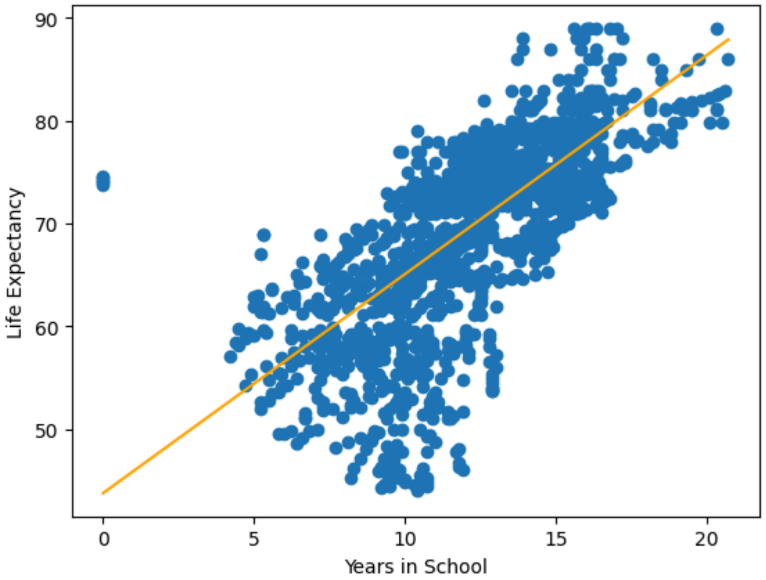
# Example

**Task:** Given your education, predict your life expectancy

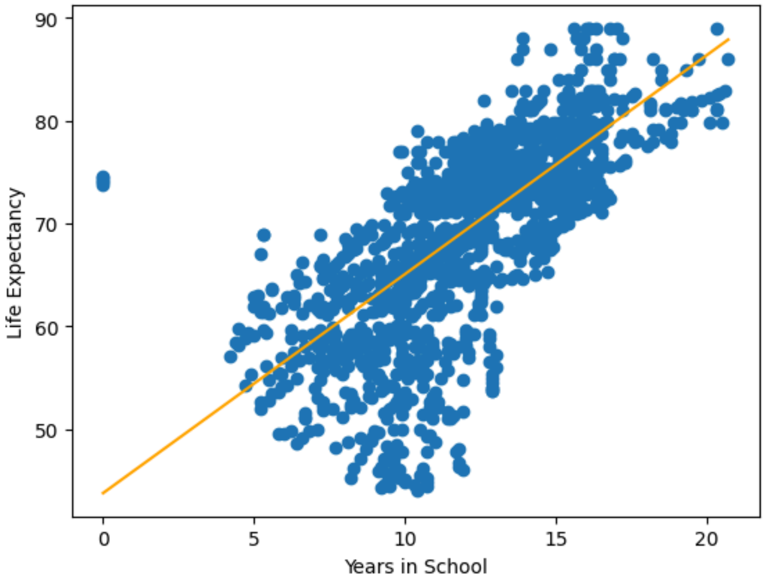
Plot the datapoints $(x_1, y_1), (x_2, y_2), \ldots$

Plot the curve $f(x, \boldsymbol{\theta}) = \theta_1 x + \theta_0; \quad x \in [0, 25]$






We figured out linear regression!




We figured out linear regression!

But can we do better?

1. Beyond linear functions

1. Beyond linear functions
2. Overfitting

1. Beyond linear functions
2. Overfitting
3. Outliers

1. Beyond linear functions
2. Overfitting
3. Outliers
4. Regularization

1. **Beyond linear functions**
2. Overfitting
3. Outliers
4. Regularization

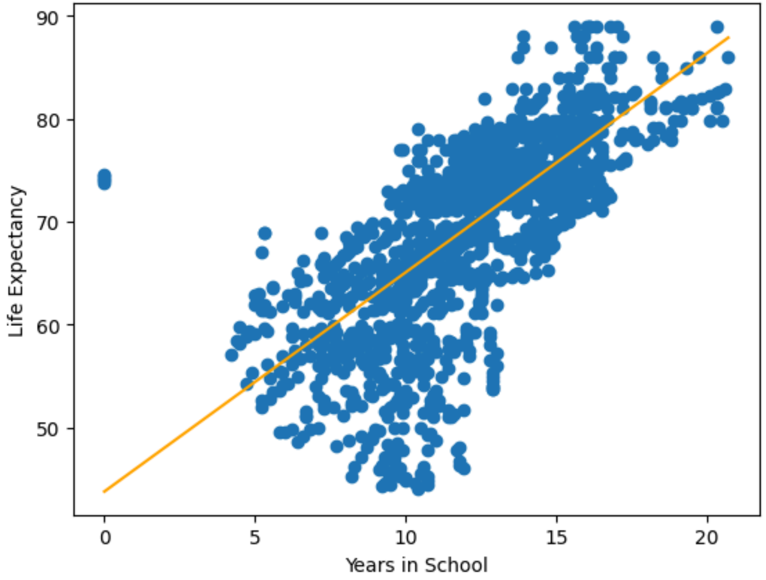
# Question:

**Question:**

Does the data look linear?

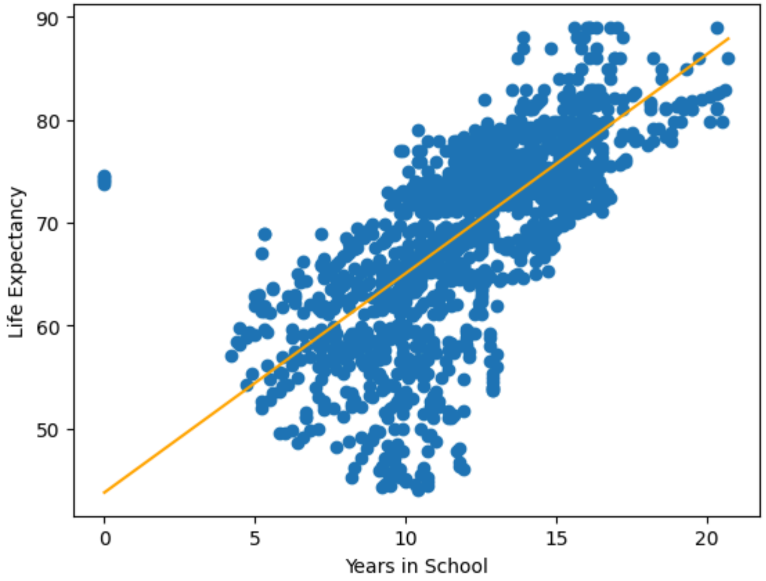
# Question:

Does the data look linear?



# Question:

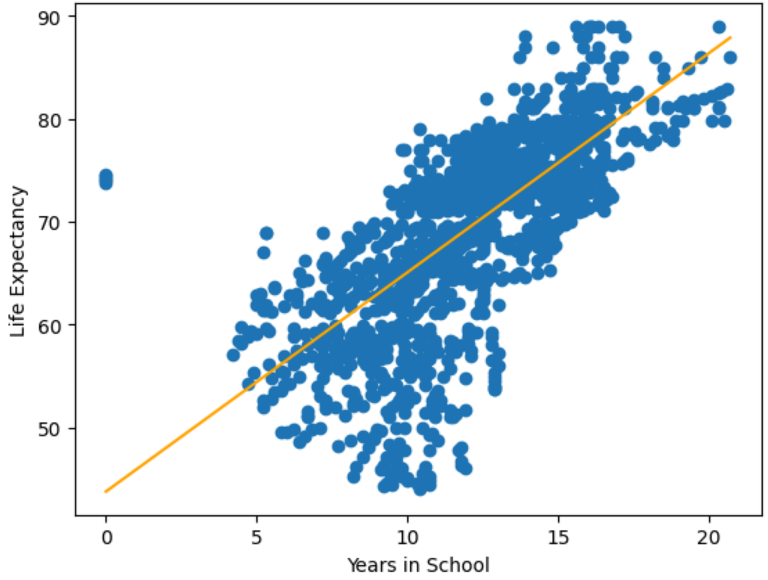
Does the data look linear?



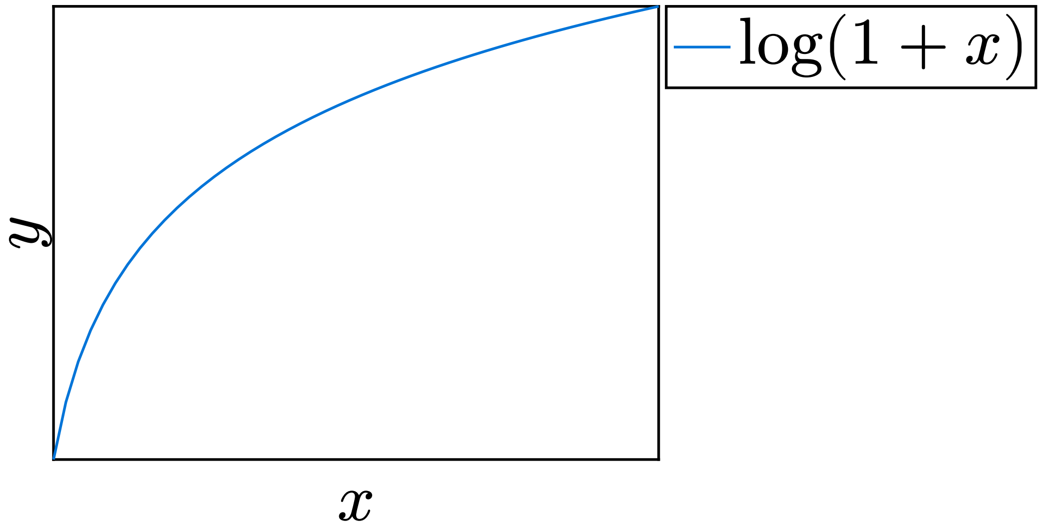

Or maybe more logarithmic?

# Question:

Does the data look linear?



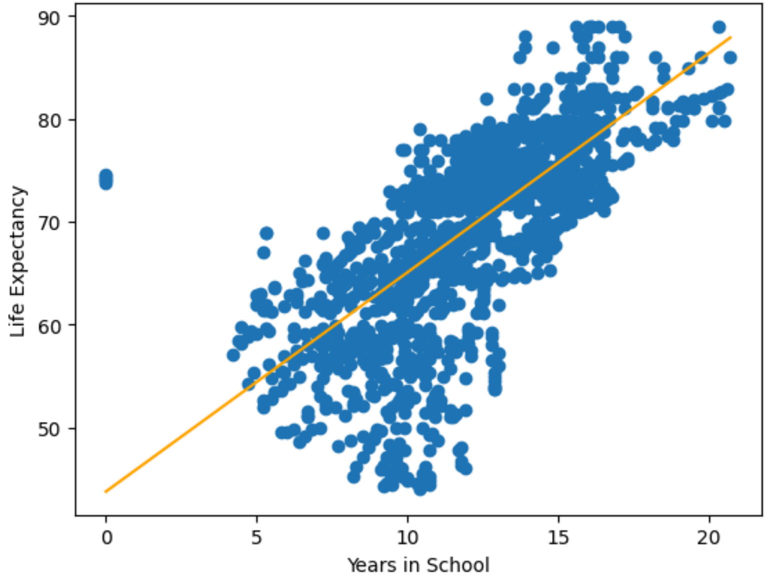

Or maybe more logarithmic?

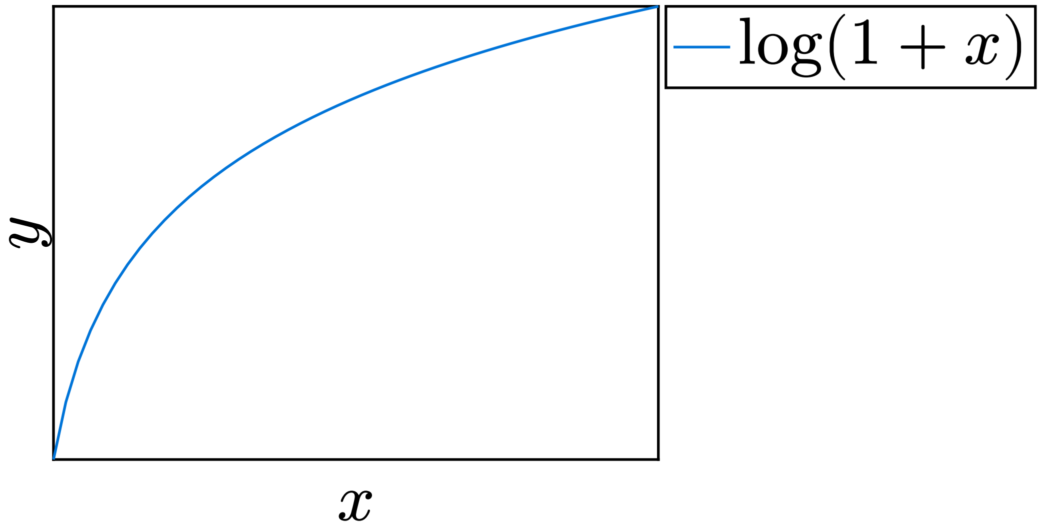


## Question:

Does the data look linear?

Or maybe more logarithmic?




However, linear regression must be linear!

**Question:** What does it mean when we say linear regression is linear?

**Question:** What does it mean when we say linear regression is linear?

**Answer:** The function $f(x, \theta)$ is a linear function of $x$

**Question:** What does it mean when we say linear regression is linear?

**Answer:** The function $f(x, \theta)$ is a linear function of $x$

**Trick:** Change of variables to make $f$ nonlinear: $x_{\text{new}} = \log(1 + x_{\text{data}})$

**Question:** What does it mean when we say linear regression is linear?

**Answer:** The function $f(x, \theta)$ is a linear function of $x$

**Trick:** Change of variables to make $f$ nonlinear: $x_{\text{new}} = \log(1 + x_{\text{data}})$

$$\boldsymbol{X}_D = \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{bmatrix} \Rightarrow \boldsymbol{X}_D = \begin{bmatrix} \log(1 + x_1) & 1 \\ \log(1 + x_2) & 1 \\ \vdots & \vdots \\ \log(1 + x_n) & 1 \end{bmatrix}$$

Now, $f$ is a linear function of $\log(1 + x)$ – a nonlinear function of $x$!

New design matrix...

$$\boldsymbol{X}_D = \begin{bmatrix} \log(1 + x_1) & 1 \\ \log(1 + x_2) & 1 \\ \vdots & \vdots \\ \log(1 + x_n) & 1 \end{bmatrix}$$

New function...

$$f\left(x, \begin{bmatrix} \theta_1 \\ \theta_0 \end{bmatrix}\right) = \theta_1 \log(1 + x) + \theta_0$$

New design matrix...

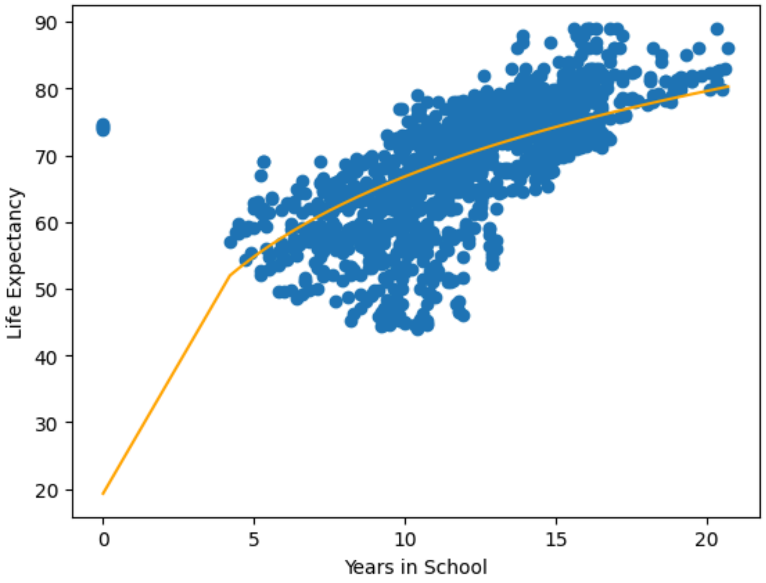
$$\boldsymbol{X}_D = \begin{bmatrix} \log(1 + x_1) & 1 \\ \log(1 + x_2) & 1 \\ \vdots & \vdots \\ \log(1 + x_n) & 1 \end{bmatrix}$$

New function...

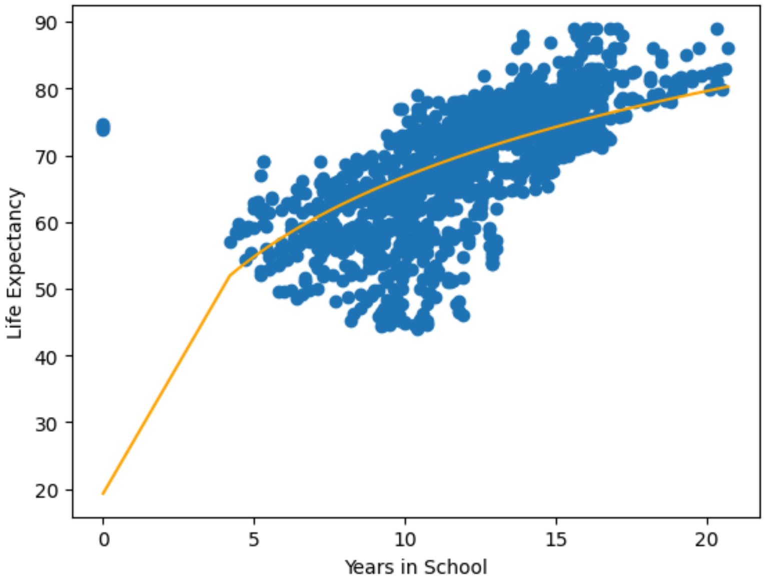
$$f\left(x, \begin{bmatrix} \theta_1 \\ \theta_0 \end{bmatrix}\right) = \theta_1 \log(1 + x) + \theta_0$$

Same solution...

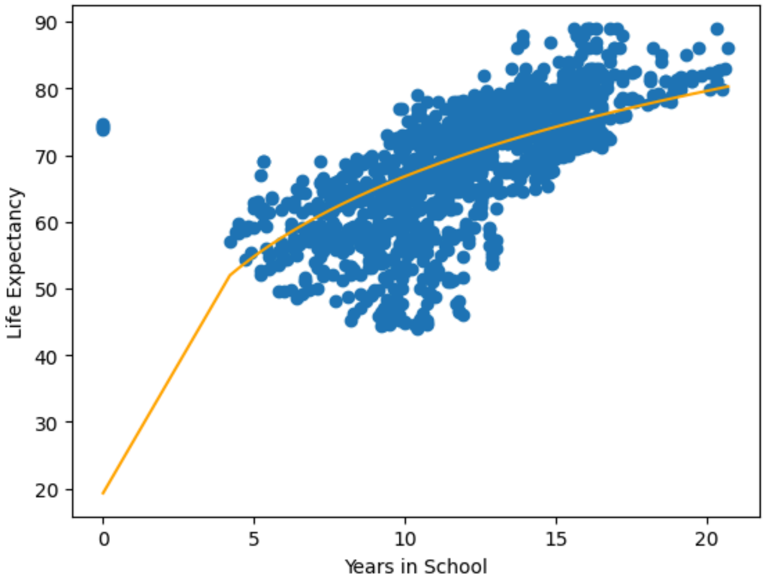
$$\boldsymbol{\theta} = \left(\boldsymbol{X}_D^\top \boldsymbol{X}_D\right)^{-1} \boldsymbol{X}_D^\top \boldsymbol{y}$$



Life Expectancy
Years in School
90
80
70
60
50
40
30
20
0
5
10
15
20




Better, but still not perfect




Better, but still not perfect

Can we do even better?

# What about polynomials?

# What about polynomials?

$$f(x) = ax^n + bx^{n-1} + ... + cx + d$$

What about polynomials?

$$f(x) = ax^n + bx^{n-1} + ... + cx + d$$

Polynomials can approximate **any** function (universal function approximator)

What about polynomials?

$$f(x) = ax^n + bx^{n-1} + ... + cx + d$$

Polynomials can approximate **any** function (universal function approximator)

Can we extend linear regression to polynomials?

Expand $x$ to a multi-dimensional input space...

Expand $x$ to a multi-dimensional input space...

$$\boldsymbol{X}_D = \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{bmatrix} \Rightarrow \boldsymbol{X}_D = \begin{bmatrix} x_1^n & x_1^{n-1} & \dots & x_1 & 1 \\ x_2^n & x_2^{n-1} & \dots & x_2 & 1 \\ \vdots & \vdots & \ddots & & \\ x_n & x_n^{n-1} & \dots & x_n & 1 \end{bmatrix}$$

And add some new parameters...

$$\boldsymbol{\theta} = [\theta_1 \ \ \theta_0]^\top \Rightarrow \boldsymbol{\theta} = [\theta_n \ \ \theta_{n-1} \ \ \dots \ \ \theta_1 \ \ \theta_0]^\top$$

$$\boldsymbol{X}_D \boldsymbol{\theta} = \begin{bmatrix} x_1^n & x_1^{n-1} & \dots & x_1 & 1 \\ x_2^n & x_2^{n-1} & \dots & x_2 & 1 \\ \vdots & \vdots & \ddots & & \\ x_n & x_n^{n-1} & \dots & x_n & 1 \end{bmatrix} \begin{bmatrix} \theta_n \\ \theta_{n-1} \\ \vdots \\ \theta_0 \end{bmatrix} = \begin{bmatrix} \theta_n x_1^n + \theta_{n-1} x_1^{n-1} + \dots + \theta_0 \\ \theta_n x_2 + \theta_{n-1} x_2^{n-1} + \dots + \theta_0 \\ \vdots \\ \theta_n x_n^n + \theta_{n-1} x_n^{n-1} + \dots + \theta_0 \end{bmatrix}$$

$$\boldsymbol{X}_D \boldsymbol{\theta} = \begin{bmatrix} x_1^n & x_1^{n-1} & \dots & x_1 & 1 \\ x_2^n & x_2^{n-1} & \dots & x_2 & 1 \\ \vdots & \vdots & \ddots & & \\ x_n & x_n^{n-1} & \dots & x_n & 1 \end{bmatrix} \begin{bmatrix} \theta_n \\ \theta_{n-1} \\ \vdots \\ \theta_0 \end{bmatrix} = \begin{bmatrix} \theta_n x_1^n + \theta_{n-1} x_1^{n-1} + \dots + \theta_0 \\ \theta_n x_2 + \theta_{n-1} x_2^{n-1} + \dots + \theta_0 \\ \vdots \\ \theta_n x_n^n + \theta_{n-1} x_n^{n-1} + \dots + \theta_0 \end{bmatrix}$$

New function…

$$f(x, \boldsymbol{\theta}) = \theta_n x^n + \theta_{n-1} x^{n-1}, \dots, \theta_1 + x^1 + \theta_0$$

$$\boldsymbol{X}_D \boldsymbol{\theta} = \begin{bmatrix} x_1^n & x_1^{n-1} & \ldots & x_1 & 1 \\ x_2^n & x_2^{n-1} & \ldots & x_2 & 1 \\ \vdots & \vdots & \ddots & & \\ x_n & x_n^{n-1} & \ldots & x_n & 1 \end{bmatrix} \begin{bmatrix} \theta_n \\ \theta_{n-1} \\ \vdots \\ \theta_0 \end{bmatrix} = \begin{bmatrix} \theta_n x_1^n + \theta_{n-1} x_1^{n-1} + \ldots + \theta_0 \\ \theta_n x_2 + \theta_{n-1} x_2^{n-1} + \ldots + \theta_0 \\ \vdots \\ \theta_n x_n^n + \theta_{n-1} x_n^{n-1} + \ldots + \theta_0 \end{bmatrix}$$

New function…

$$f(x, \boldsymbol{\theta}) = \theta_n x^n + \theta_{n-1} x^{n-1}, \ldots, \theta_1 + x^1 + \theta_0$$

Same solution…

$$\boldsymbol{\theta} = \left( \boldsymbol{X}_D^\top \boldsymbol{X}_D \right)^{-1} \boldsymbol{X}_D^\top \boldsymbol{y}$$

$$f(x, \boldsymbol{\theta}) = \theta_n x^n + \theta_{n-1} x^{n-1}, ..., \theta_1 + x^1 + \theta_0$$

$$f(x, \boldsymbol{\theta}) = \theta_n x^n + \theta_{n-1} x^{n-1}, ..., \theta_1 + x^1 + \theta_0$$

**Summary:** By changing the input space, we can fit a polynomial to the data using a linear fit!

1. **Beyond linear functions**
2. Overfitting
3. Outliers
4. Regularization

1. Beyond linear functions
2. **Overfitting**
3. Outliers
4. Regularization

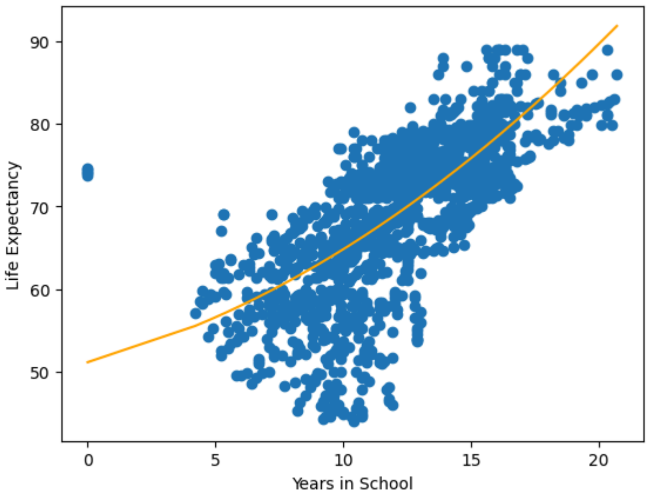
$$f(x, \boldsymbol{\theta}) = \theta_n x^n + \theta_{n-1} x^{n-1}, ..., \theta_1 + x^1 + \theta_0$$

$$f(x, \boldsymbol{\theta}) = \theta_n x^n + \theta_{n-1} x^{n-1}, ..., \theta_1 + x^1 + \theta_0$$

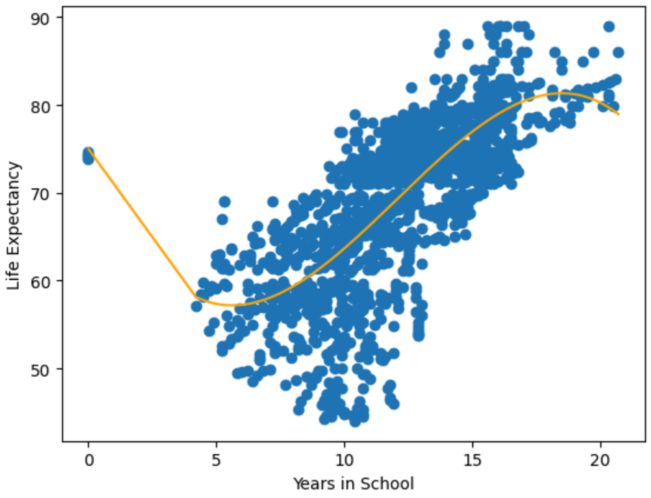
How do we choose $n$ (polynomial order) that provides the best fit?

$$f(x, \boldsymbol{\theta}) = \theta_n x^n + \theta_{n-1} x^{n-1}, ..., \theta_1 + x^1 + \theta_0$$

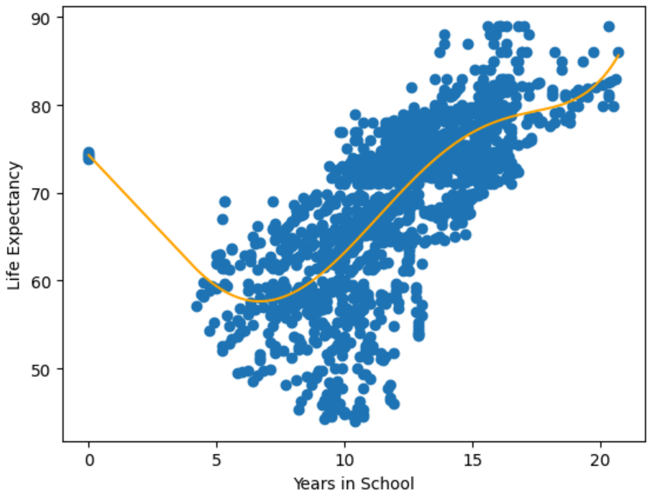
How do we choose $n$ (polynomial order) that provides the best fit?




$n = 2$

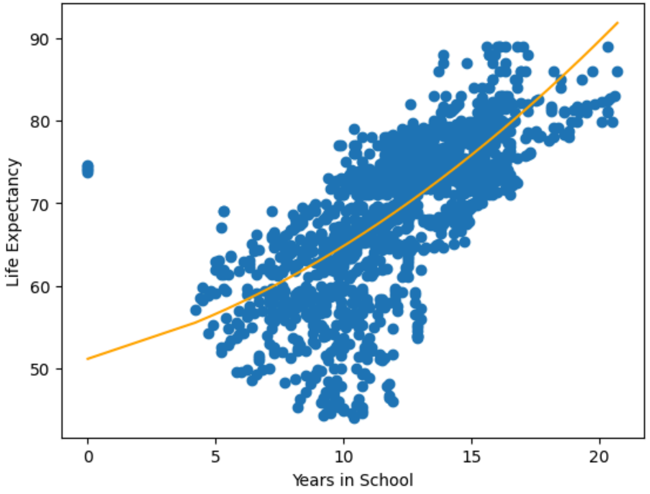



$n = 3$

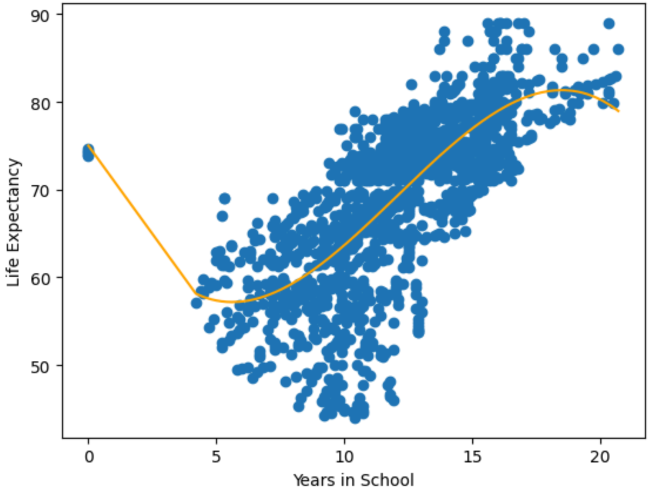



$n = 5$

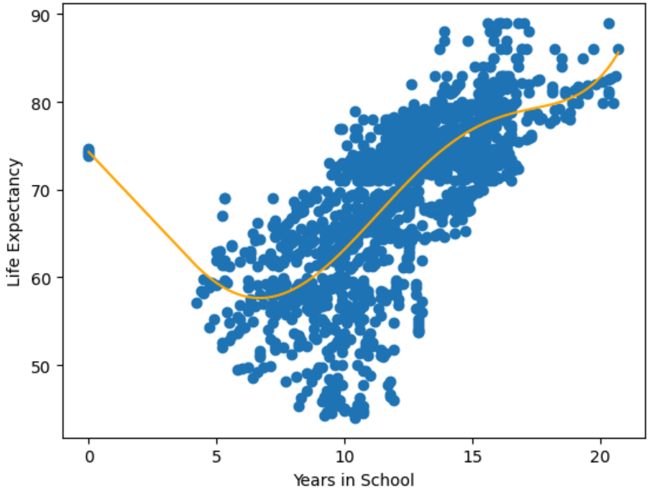
How do we choose $n$ (polynomial order) that provides the best fit?




$n = 2$



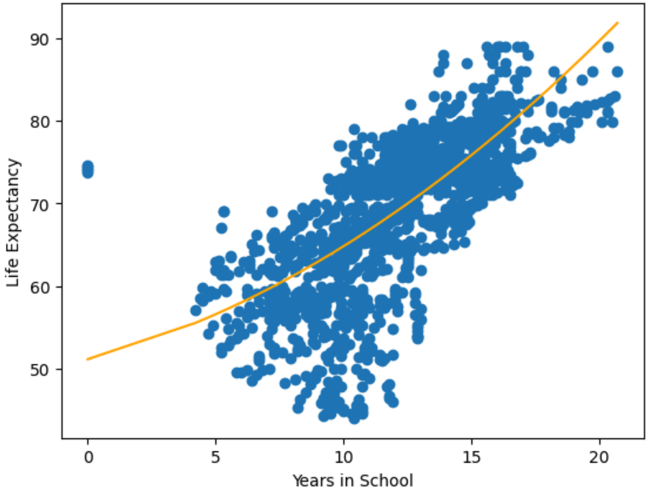

$n = 3$



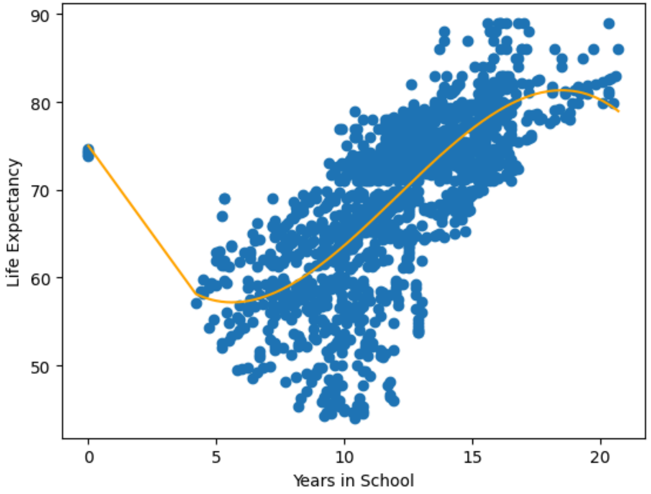

$n = 5$

Pick the $n$ with the smallest loss

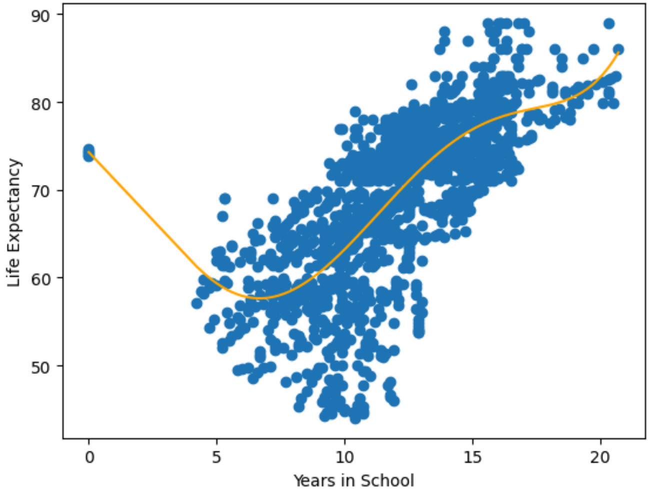
$$\arg\min_{\boldsymbol{\theta}, n} \mathcal{L}(\boldsymbol{x}, \boldsymbol{y}, (\boldsymbol{\theta}, n))$$




$n = 2$

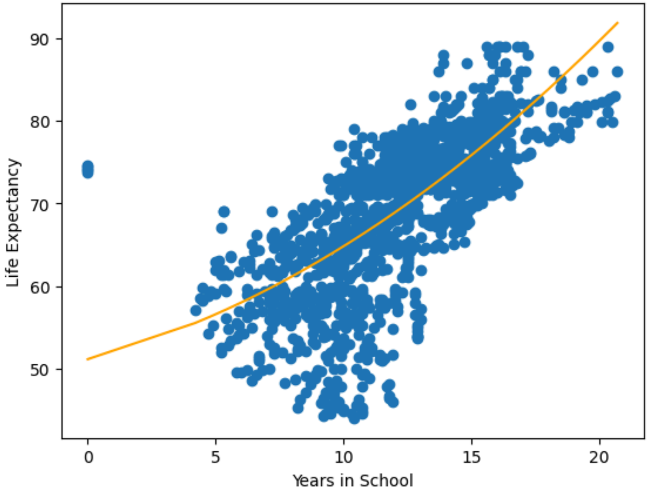


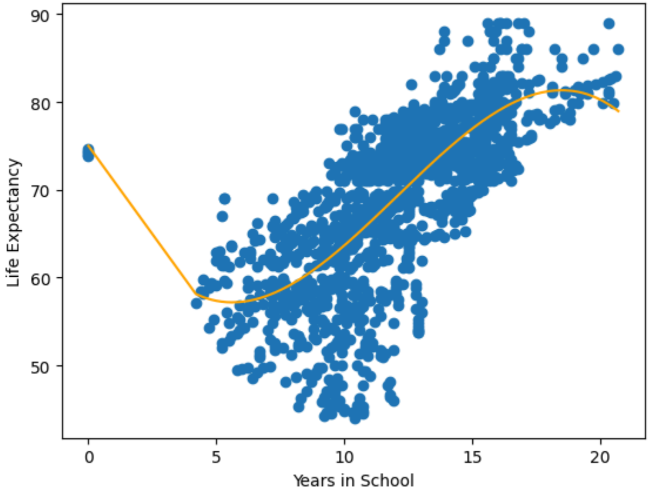

$n = 3$

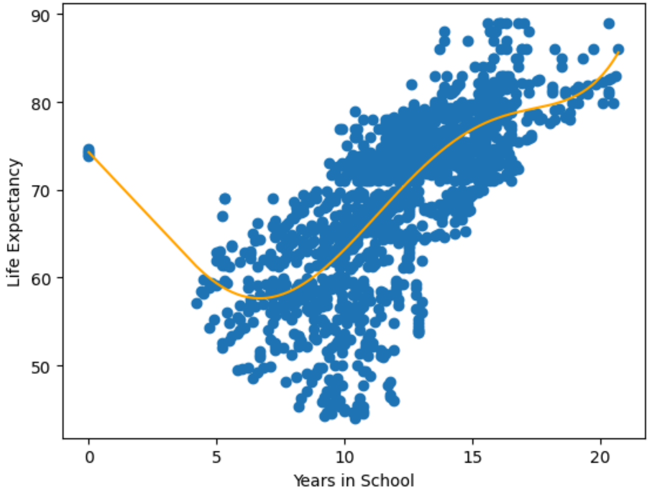



$n = 5$

**Question:** Which $n$ do you think has the smallest loss?



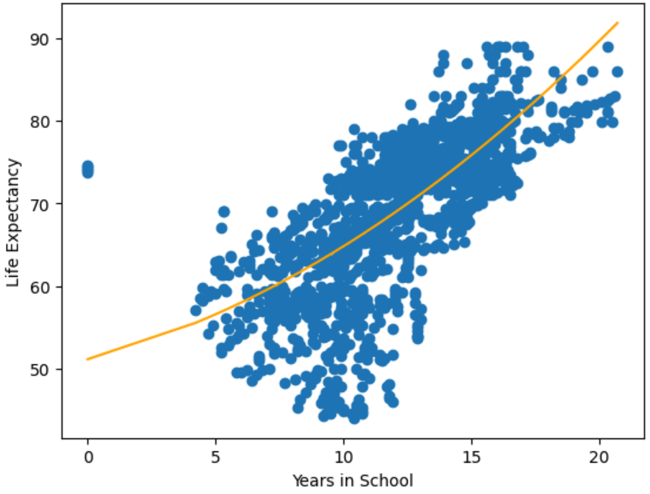



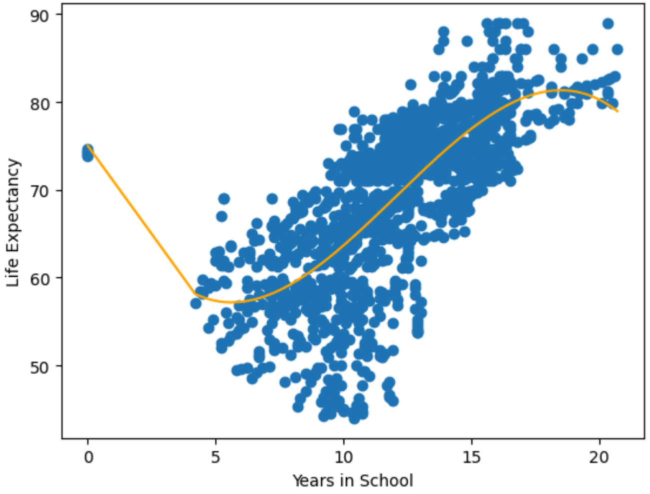

$n = 2$ $\quad$ $n = 3$ $\quad$ $n = 5$

**Question:** Which $n$ do you think has the smallest loss?

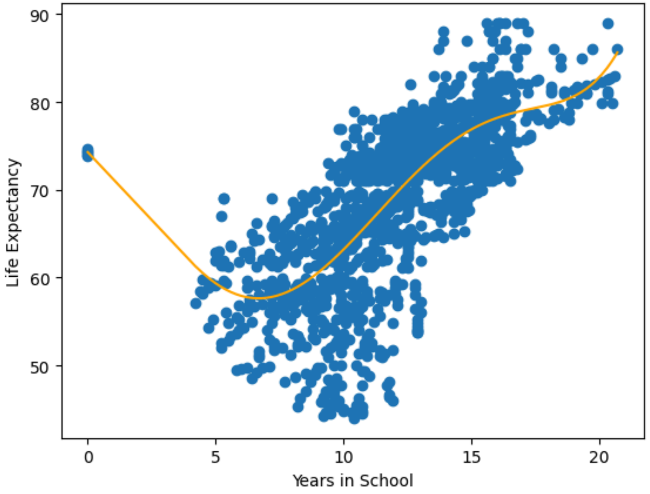
**Answer:** $n = 5$, but intuitively, $n = 5$ does not seem very good...




$n = 2$

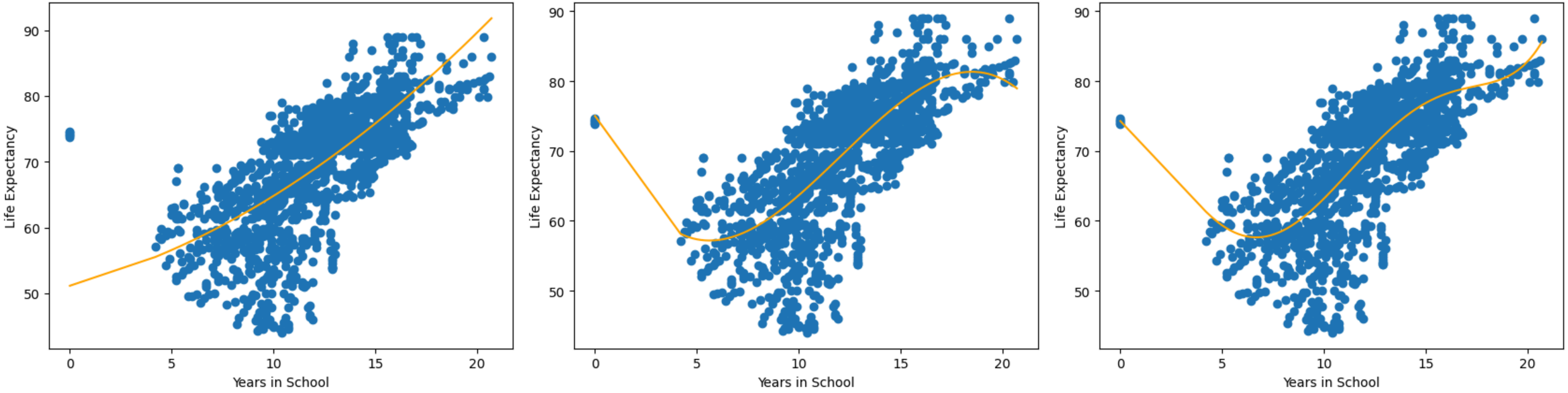



$n = 3$




$n = 5$

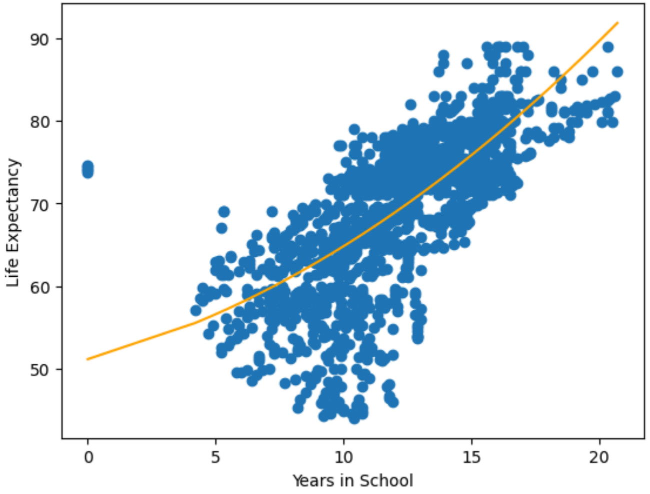
More specifically, $n = 5$ will not generalize to new data



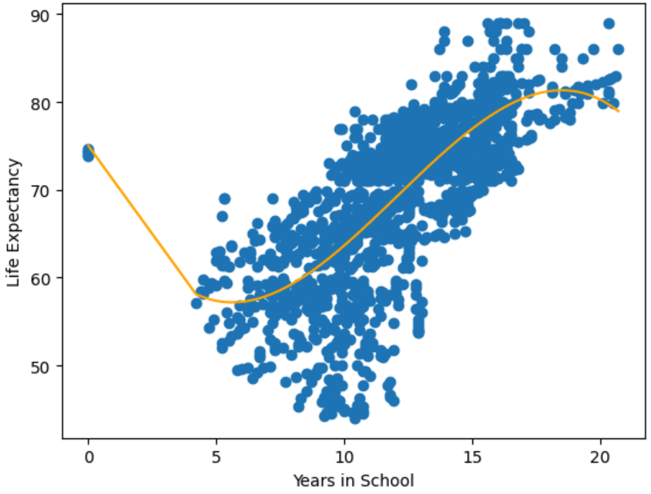

$n = 2$ $\quad$ $n = 3$ $\quad$ $n = 5$

More specifically, $n = 5$ will not generalize to new data

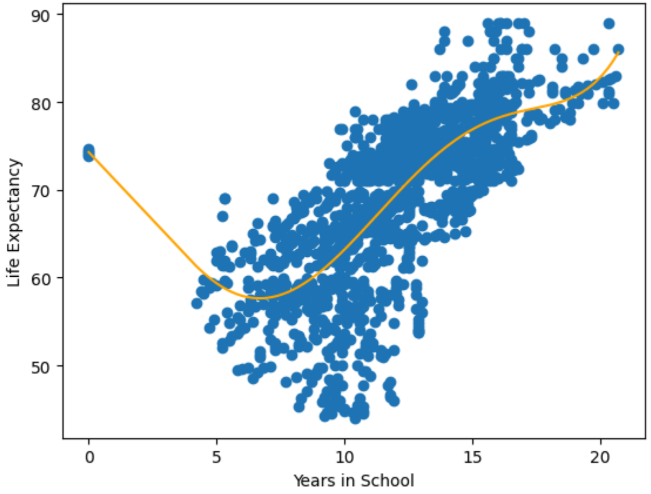
We will only use our model for new data (we already have the $y$ for a known $x$)!




$n = 2$




$n = 3$




$n = 5$

More specifically, $n = 5$ will not generalize to new data

We will only use our model for new data (we already have the $y$ for a known $x$)!

When our model has a small loss but does not generalize to new data, we call it **overfitting**

When our model has a small loss but does not generalize to new data, we call it **overfitting**

The model has fit too closely to the sampled data points, rather than the trend

When our model has a small loss but does not generalize to new data, we call it **overfitting**

The model has fit too closely to the sampled data points, rather than the trend

Models that overfit are not useful for making predictions

When our model has a small loss but does not generalize to new data, we call it **overfitting**

The model has fit too closely to the sampled data points, rather than the trend

Models that overfit are not useful for making predictions

Back to the question…

When our model has a small loss but does not generalize to new data, we call it **overfitting**

The model has fit too closely to the sampled data points, rather than the trend

Models that overfit are not useful for making predictions

Back to the question…

**Question:** How do we choose $n$ such that our polynomial model works for unseen/new data?

When our model has a small loss but does not generalize to new data, we call it **overfitting**

The model has fit too closely to the sampled data points, rather than the trend

Models that overfit are not useful for making predictions

Back to the question…

**Question:** How do we choose $n$ such that our polynomial model works for unseen/new data?

**Answer:** Compute the loss on unseen data!

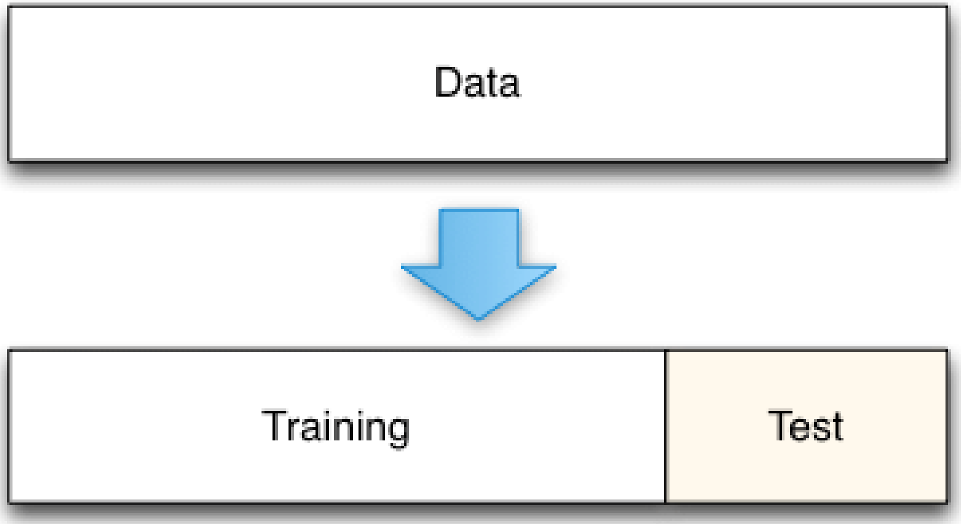
To compute the loss on unseen data, we will need unseen data

To compute the loss on unseen data, we will need unseen data

Let us create some unseen data!

To compute the loss on unseen data, we will need unseen data

Let us create some unseen data!



# Example

**Question:** How do we choose the training and testing datasets?

# Example

**Question:** How do we choose the training and testing datasets?

$$\text{Option 1: } \boldsymbol{x}_{\text{train}} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \boldsymbol{y}_{\text{train}} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}; \quad \boldsymbol{x}_{\text{test}} = \begin{bmatrix} x_4 \\ x_5 \end{bmatrix}; \boldsymbol{y}_{\text{test}} = \begin{bmatrix} y_4 \\ y_5 \end{bmatrix}$$

# Example

**Question:** How do we choose the training and testing datasets?

$$\text{Option 1: } \boldsymbol{x}_{\text{train}} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \boldsymbol{y}_{\text{train}} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}; \quad \boldsymbol{x}_{\text{test}} = \begin{bmatrix} x_4 \\ x_5 \end{bmatrix}; \boldsymbol{y}_{\text{test}} = \begin{bmatrix} y_4 \\ y_5 \end{bmatrix}$$

$$\text{Option 2: } \boldsymbol{x}_{\text{train}} = \begin{bmatrix} x_4 \\ x_1 \\ x_3 \end{bmatrix} \boldsymbol{y}_{\text{train}} = \begin{bmatrix} y_4 \\ y_1 \\ y_3 \end{bmatrix}; \quad \boldsymbol{x}_{\text{test}} = \begin{bmatrix} x_2 \\ x_5 \end{bmatrix}; \boldsymbol{y}_{\text{test}} = \begin{bmatrix} y_2 \\ y_5 \end{bmatrix}$$

# Example

**Question:** How do we choose the training and testing datasets?

$$\text{Option 1: } \boldsymbol{x}_{\text{train}} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \boldsymbol{y}_{\text{train}} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}; \quad \boldsymbol{x}_{\text{test}} = \begin{bmatrix} x_4 \\ x_5 \end{bmatrix}; \boldsymbol{y}_{\text{test}} = \begin{bmatrix} y_4 \\ y_5 \end{bmatrix}$$

$$\text{Option 2: } \boldsymbol{x}_{\text{train}} = \begin{bmatrix} x_4 \\ x_1 \\ x_3 \end{bmatrix} \boldsymbol{y}_{\text{train}} = \begin{bmatrix} y_4 \\ y_1 \\ y_3 \end{bmatrix}; \quad \boldsymbol{x}_{\text{test}} = \begin{bmatrix} x_2 \\ x_5 \end{bmatrix}; \boldsymbol{y}_{\text{test}} = \begin{bmatrix} y_2 \\ y_5 \end{bmatrix}$$

**Answer:** Always shuffle the data

# Example

**Question:** How do we choose the training and testing datasets?

$$\text{Option 1: } \boldsymbol{x}_{\text{train}} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \boldsymbol{y}_{\text{train}} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}; \quad \boldsymbol{x}_{\text{test}} = \begin{bmatrix} x_4 \\ x_5 \end{bmatrix}; \boldsymbol{y}_{\text{test}} = \begin{bmatrix} y_4 \\ y_5 \end{bmatrix}$$

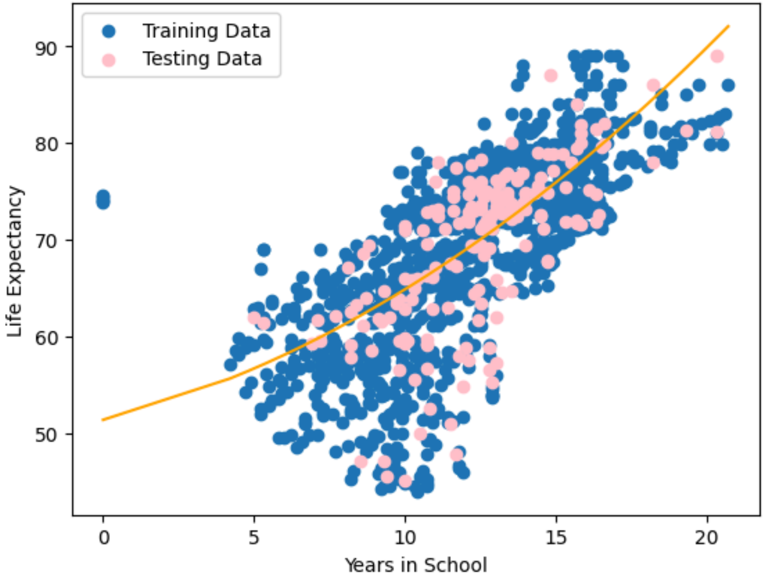
$$\text{Option 2: } \boldsymbol{x}_{\text{train}} = \begin{bmatrix} x_4 \\ x_1 \\ x_3 \end{bmatrix} \boldsymbol{y}_{\text{train}} = \begin{bmatrix} y_4 \\ y_1 \\ y_3 \end{bmatrix}; \quad \boldsymbol{x}_{\text{test}} = \begin{bmatrix} x_2 \\ x_5 \end{bmatrix}; \boldsymbol{y}_{\text{test}} = \begin{bmatrix} y_2 \\ y_5 \end{bmatrix}$$

**Answer:** Always shuffle the data

**Note:** The model must never see the testing dataset during training. This is very important!

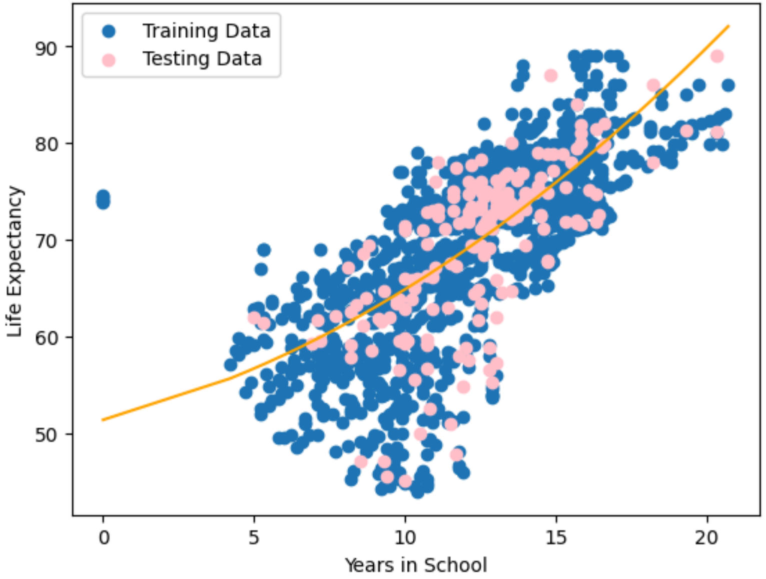
# We can now measure how the model generalizes to new data

We can now measure how the model generalizes to new data




Learn parameters from the train dataset, evaluate on the test dataset

# We can now measure how the model generalizes to new data




Learn parameters from the train dataset, evaluate on the test dataset

$$\mathcal{L}(\boldsymbol{X}_{\text{train}}, \boldsymbol{y}_{\text{train}}, \boldsymbol{\theta}) \qquad \mathcal{L}(\boldsymbol{X}_{\text{test}}, \boldsymbol{y}_{\text{test}}, \boldsymbol{\theta})$$

We use separate training and testing datasets on **all** machine learning models, not just linear regression

We use separate training and testing datasets on **all** machine learning models, not just linear regression