Model Complexity

STAT 4710

September 14, 2023

Rolling into Unit 2

✔ **Unit 1:** R for data mining

Unit 2: Prediction fundamentals

- **Lecture 1:** Model complexity
- **Lecture 2:** Bias-variance trade-off
- **Lecture 3:** Cross-validation
- **Lecture 4:** Classification
- **Lecture 5:** Unit review and quiz in class

Unit 3: Regression-based methods

Unit 4: Tree-based methods

Unit 5: Deep learning

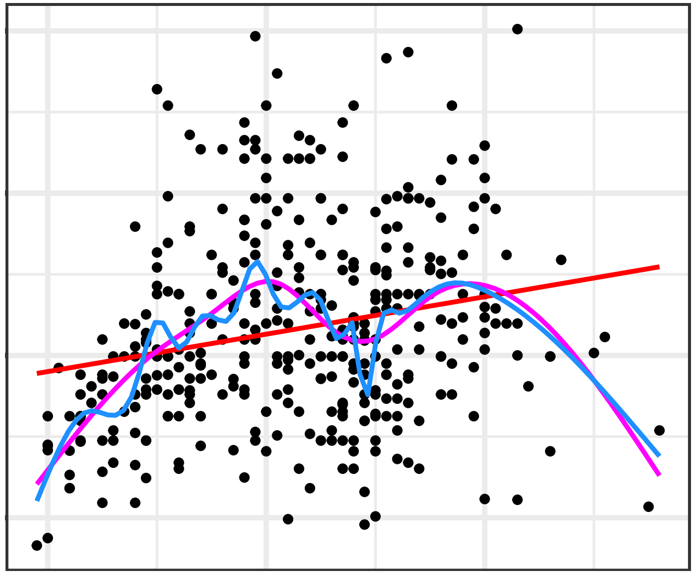
Lecture outline

Model complexity:
How flexibly a predictive model can fit its training data.

Lecture outline

Model complexity:
How flexibly a predictive model can fit its training data.

1. Case study: Fitting curves to scatter plots

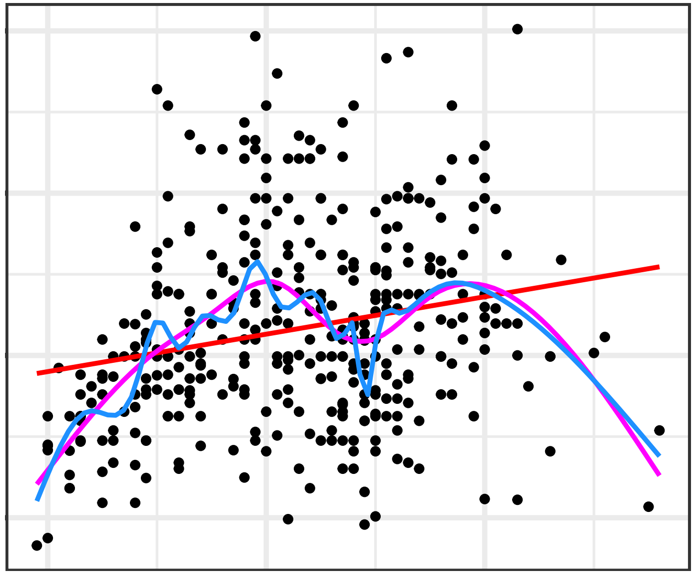


Lecture outline

Model complexity:
How flexibly a predictive model can fit its training data.

1. Case study: Fitting curves to scatter plots

2. Definition of model complexity

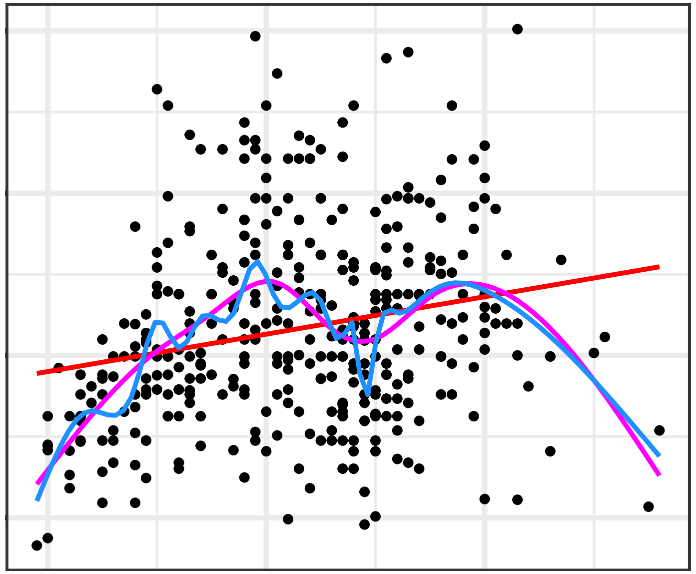


Lecture outline

Model complexity:

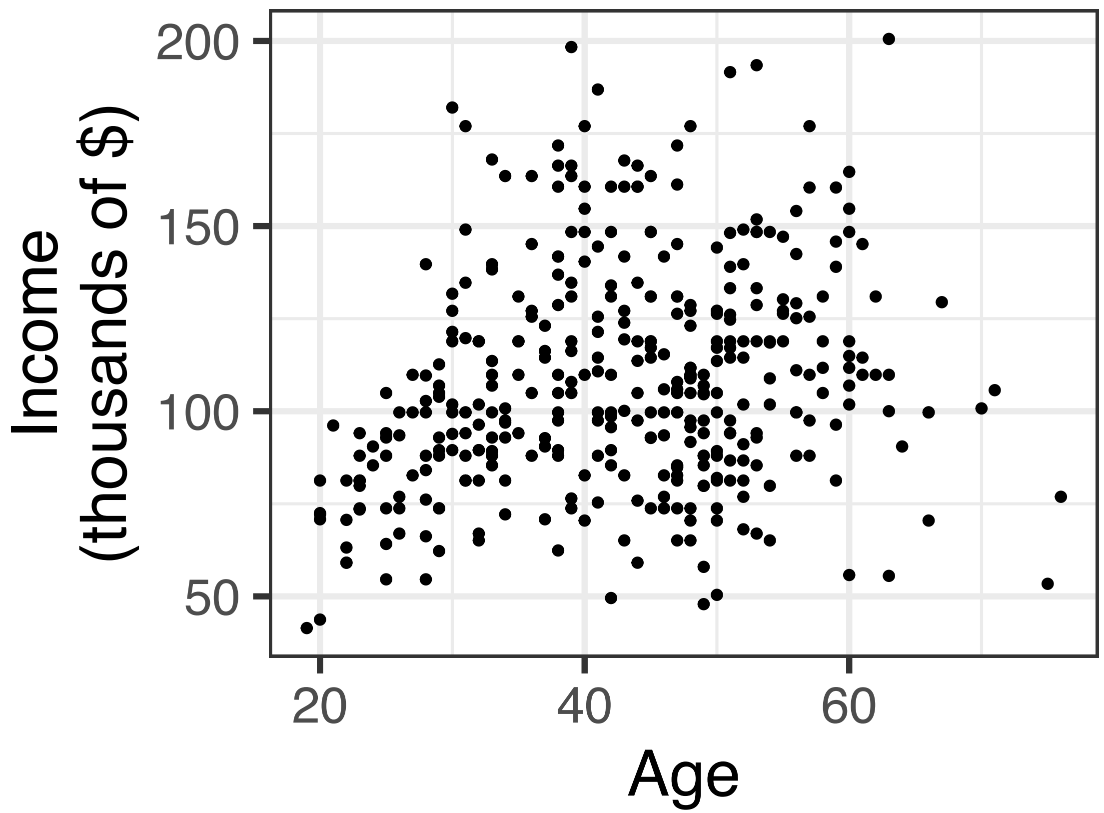
How flexibly a predictive model can fit its training data.

1. Case study: Fitting curves to scatter plots
2. Definition of model complexity
3. How model complexity impacts predictive performance



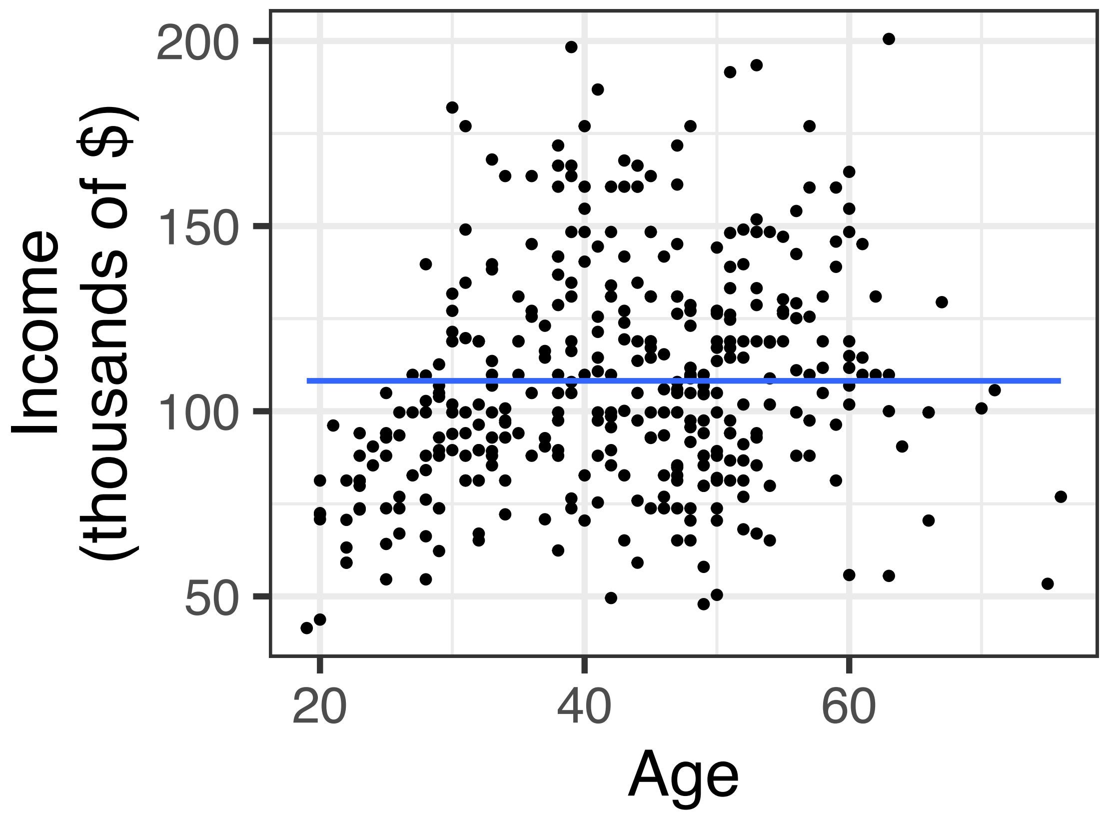
Example: Fit trend of income based on age

What does the trend look like?



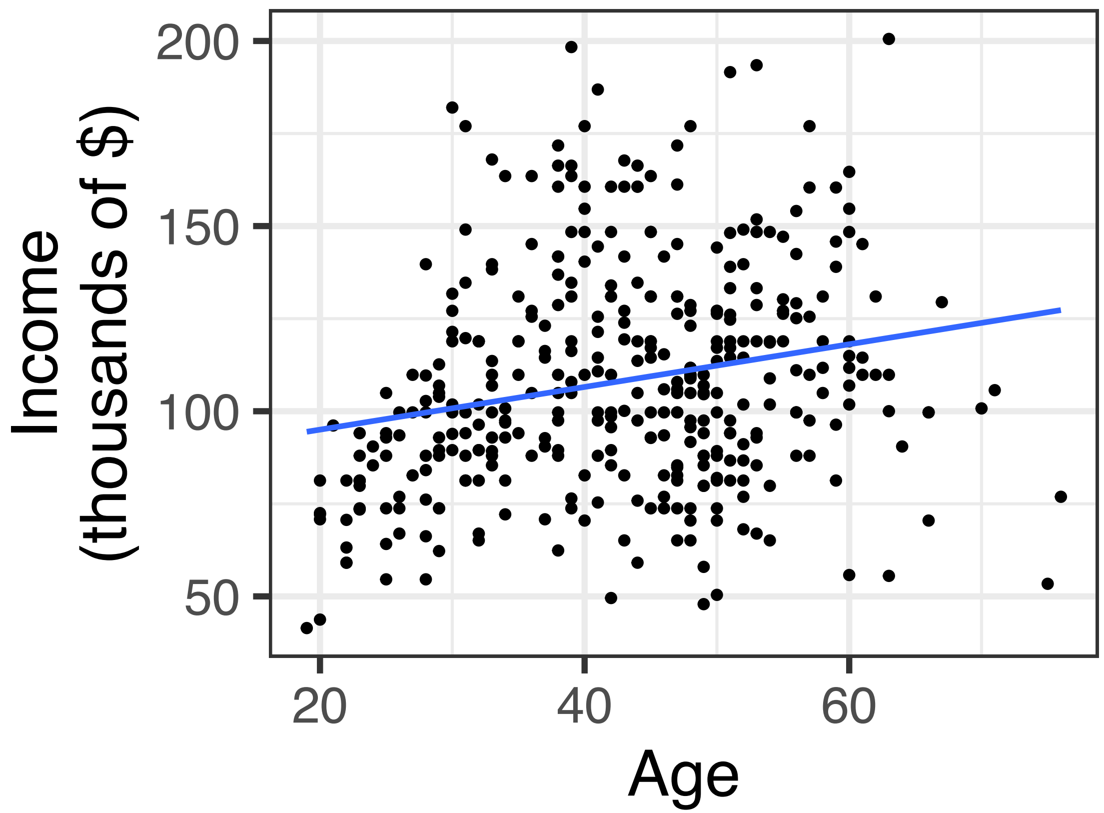
Intercept-only model (no trend)

$$\text{income} = \beta_0 + \epsilon$$



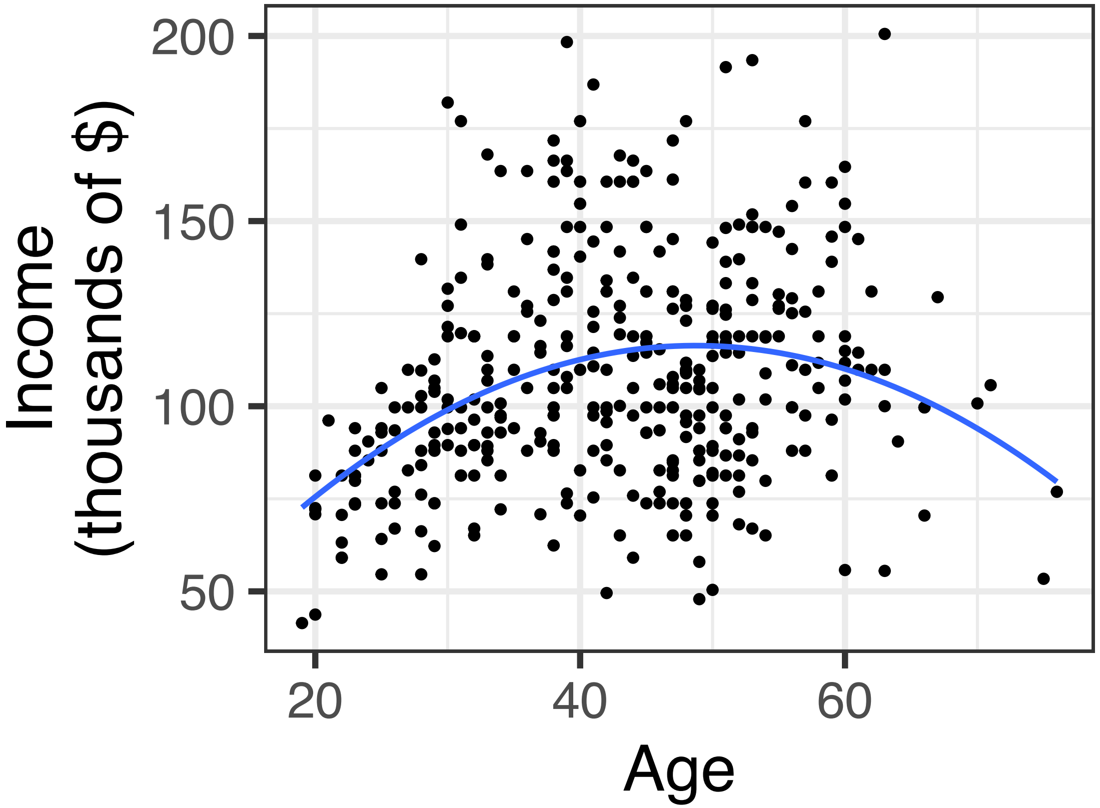
Linear model (linear trend)

$$\text{income} = \beta_0 + \beta_1 \cdot \text{age} + \epsilon$$



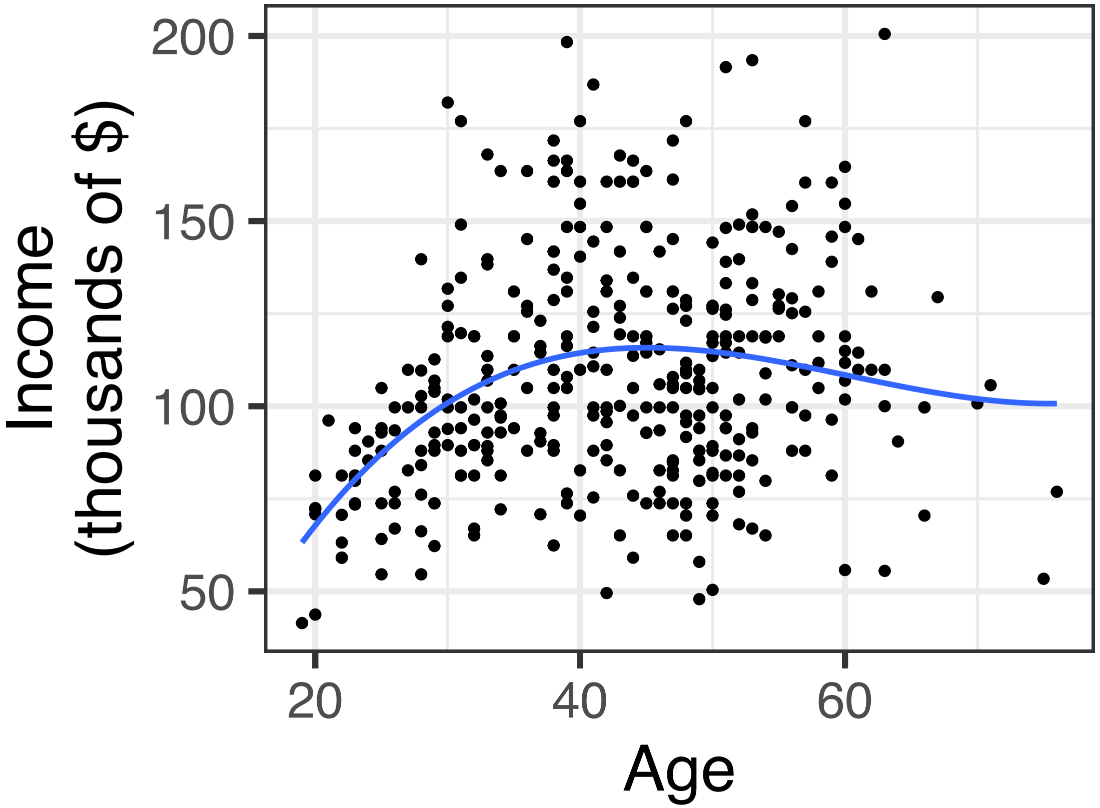
Polynomial model (quadratic trend)

$$\text{income} = \beta_0 + \beta_1 \cdot \text{age} + \beta_2 \cdot \text{age}^2 + \epsilon$$



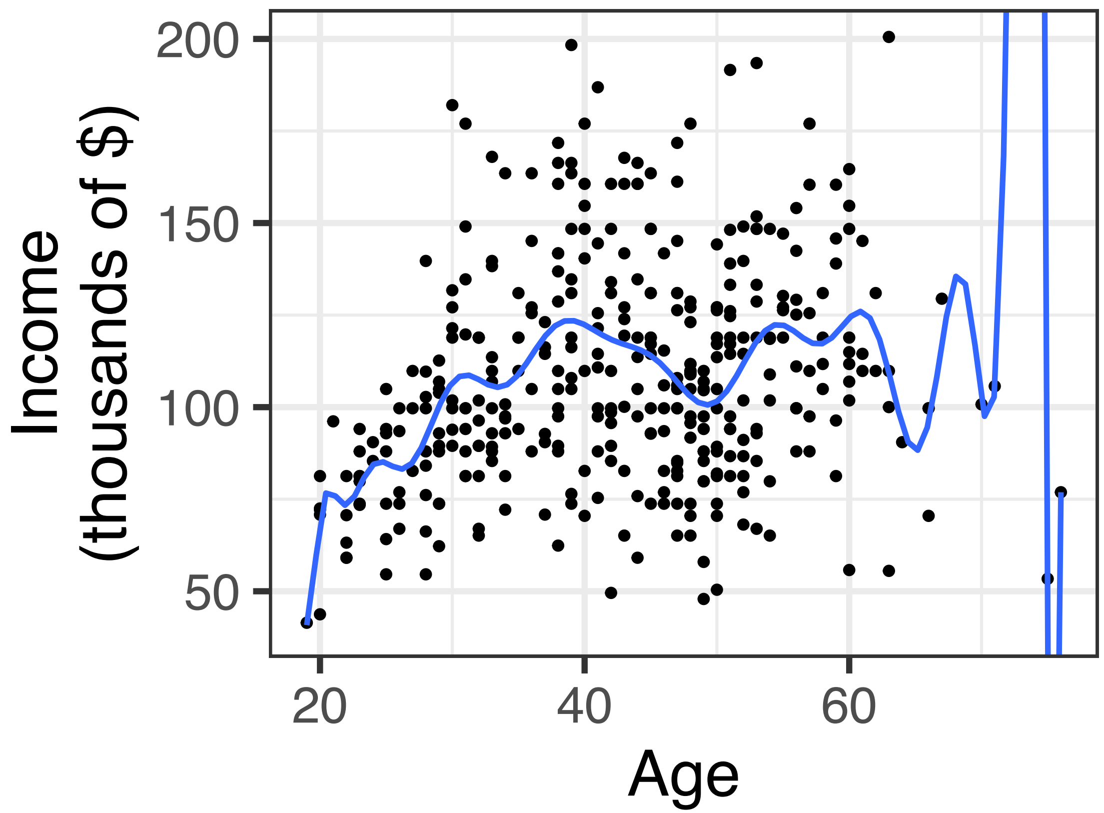
Polynomial model (cubic trend)

$$\text{income} = \beta_0 + \beta_1 \cdot \text{age} + \beta_2 \cdot \text{age}^2 + \beta_3 \cdot \text{age}^3 + \epsilon$$



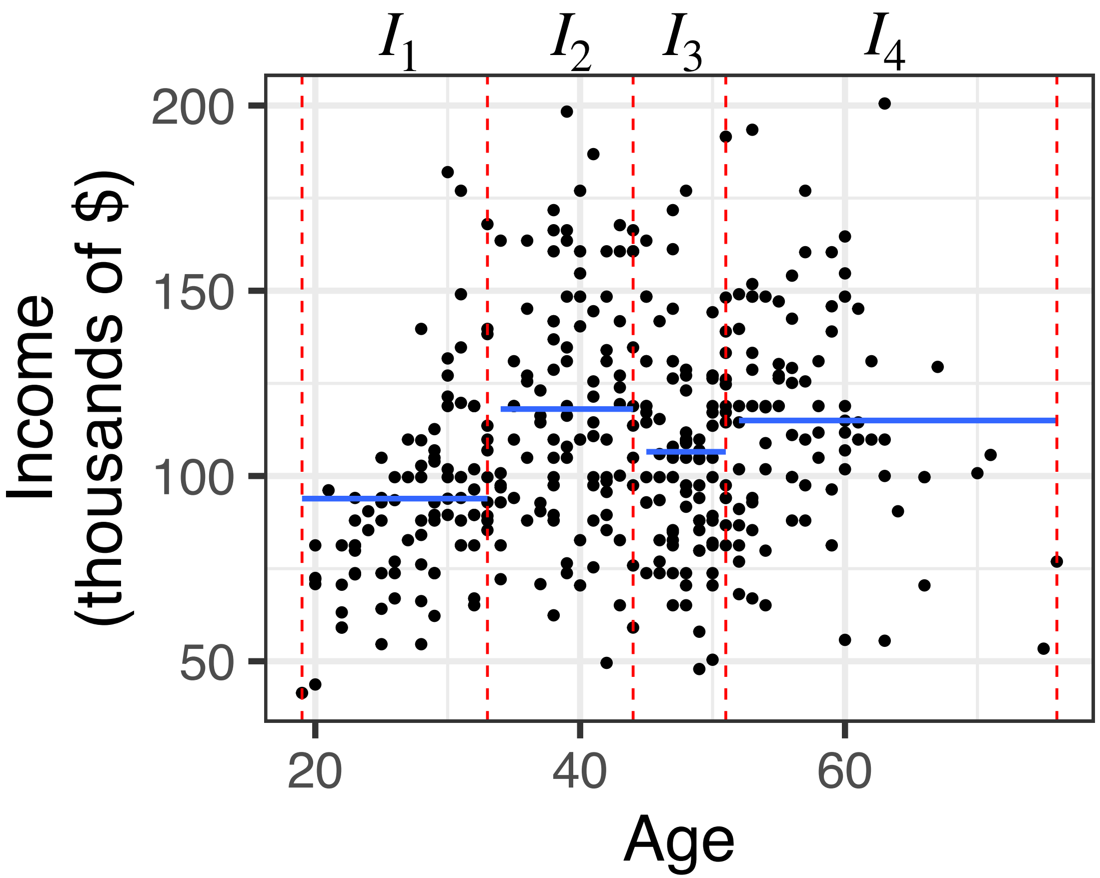
20th degree polynomial model

$$\text{income} = \beta_0 + \beta_1 \cdot \text{age} + \beta_2 \cdot \text{age}^2 + \cdots + \beta_{20} \cdot \text{age}^{20} + \epsilon$$



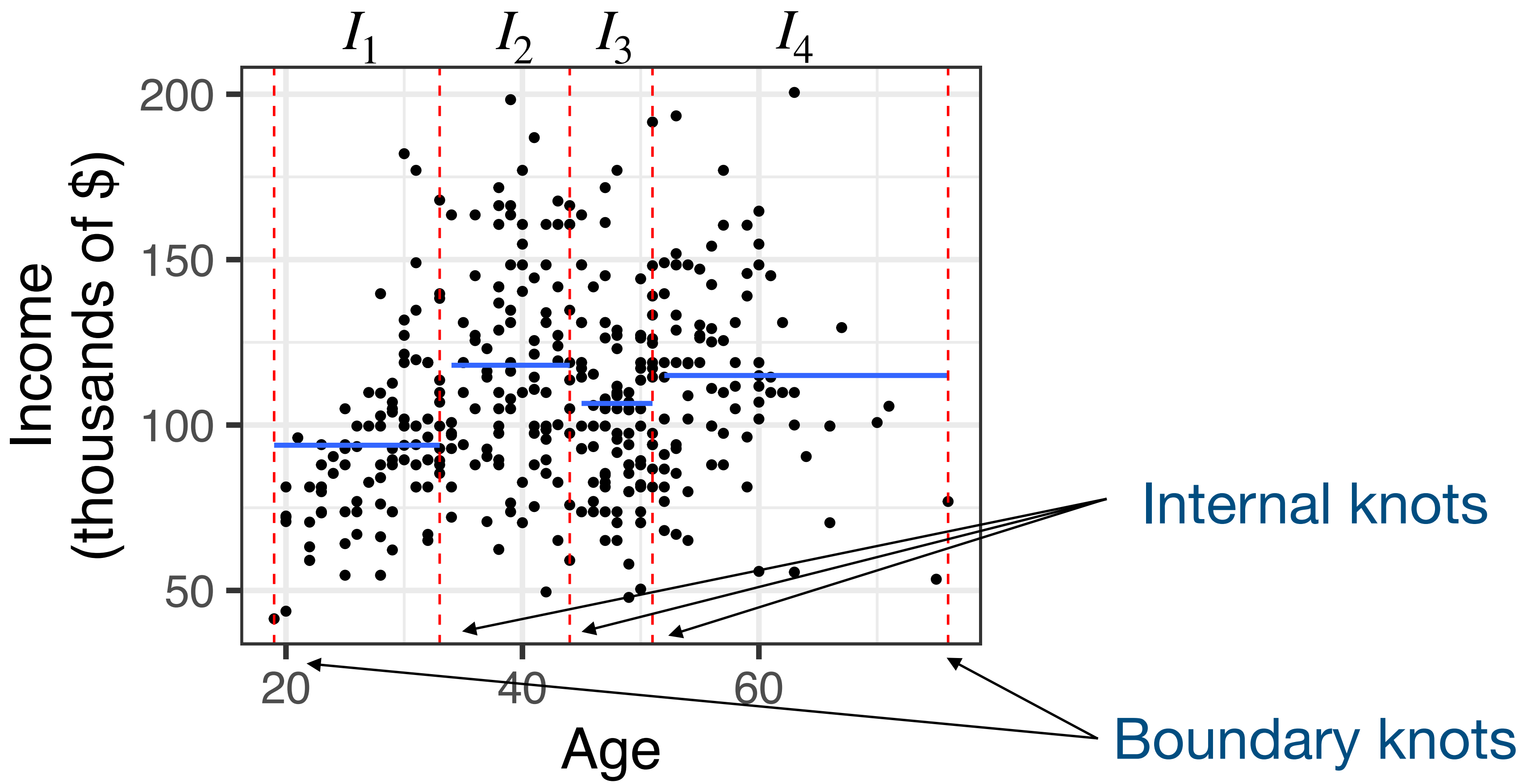
Piece-wise polynomial (piece-wise constant)

$$\text{income} = \beta_1 \cdot 1(\text{age} \in I_1) + \cdots + \beta_4 \cdot 1(\text{age} \in I_4) + \epsilon$$



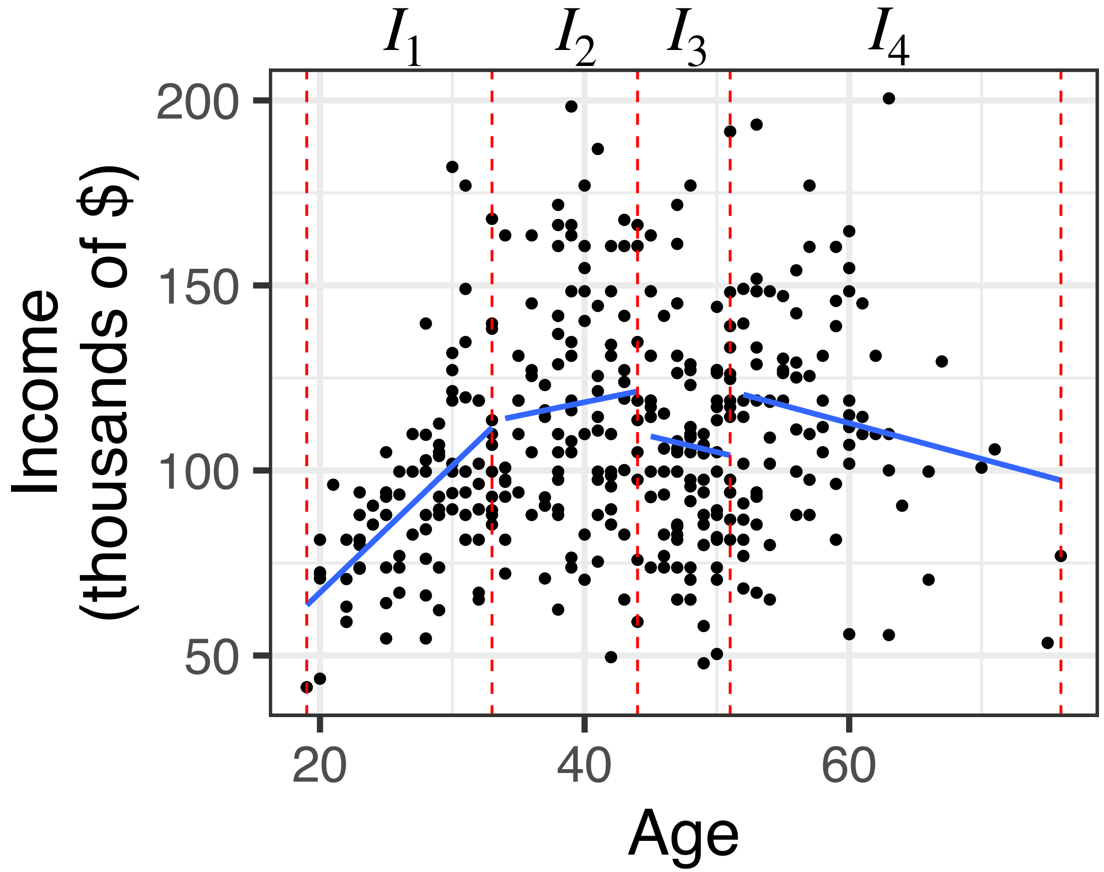
Piece-wise polynomial (piece-wise constant)

$$\text{income} = \beta_1 \cdot 1(\text{age} \in I_1) + \cdots + \beta_4 \cdot 1(\text{age} \in I_4) + \epsilon$$



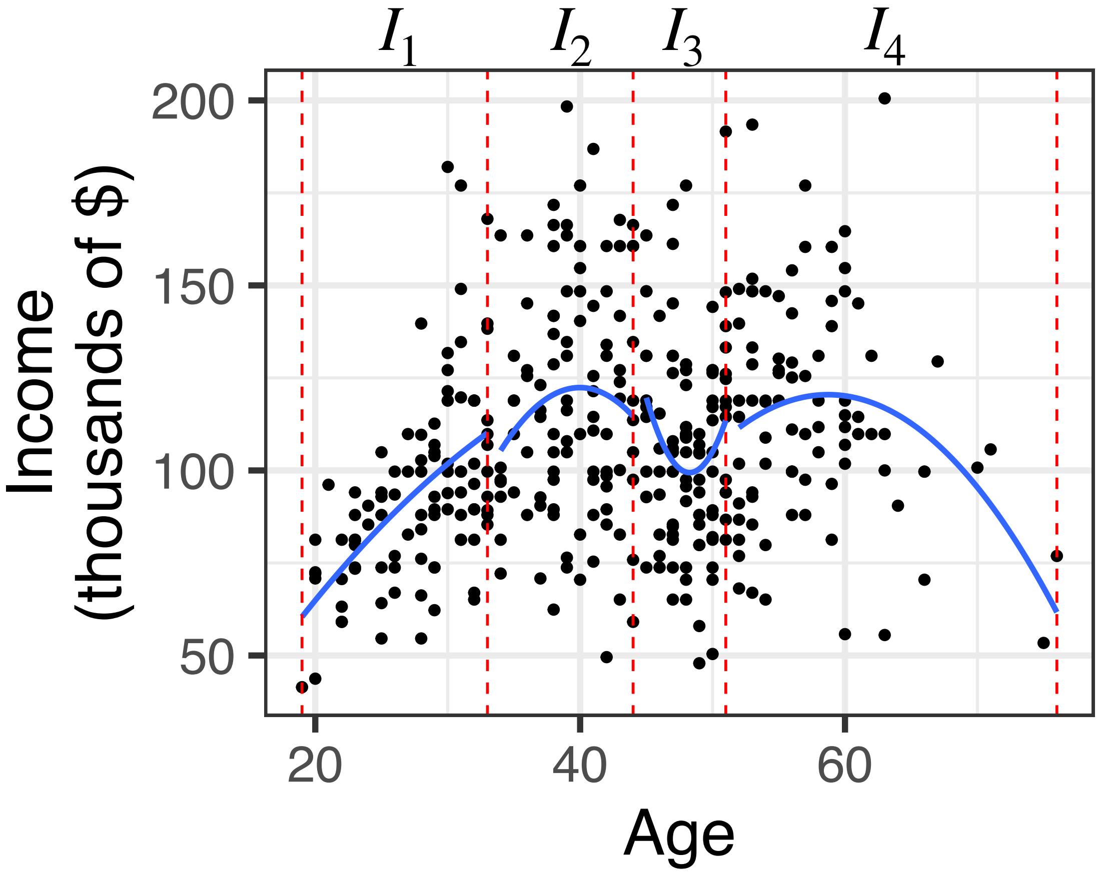
Piece-wise polynomial (piece-wise linear)

$$\text{income} = (\beta_{01} + \beta_{11}\text{age}) \cdot 1(\text{age} \in I_1) + \cdots + (\beta_{04} + \beta_{14}\text{age}) \cdot 1(\text{age} \in I_4) + \epsilon$$



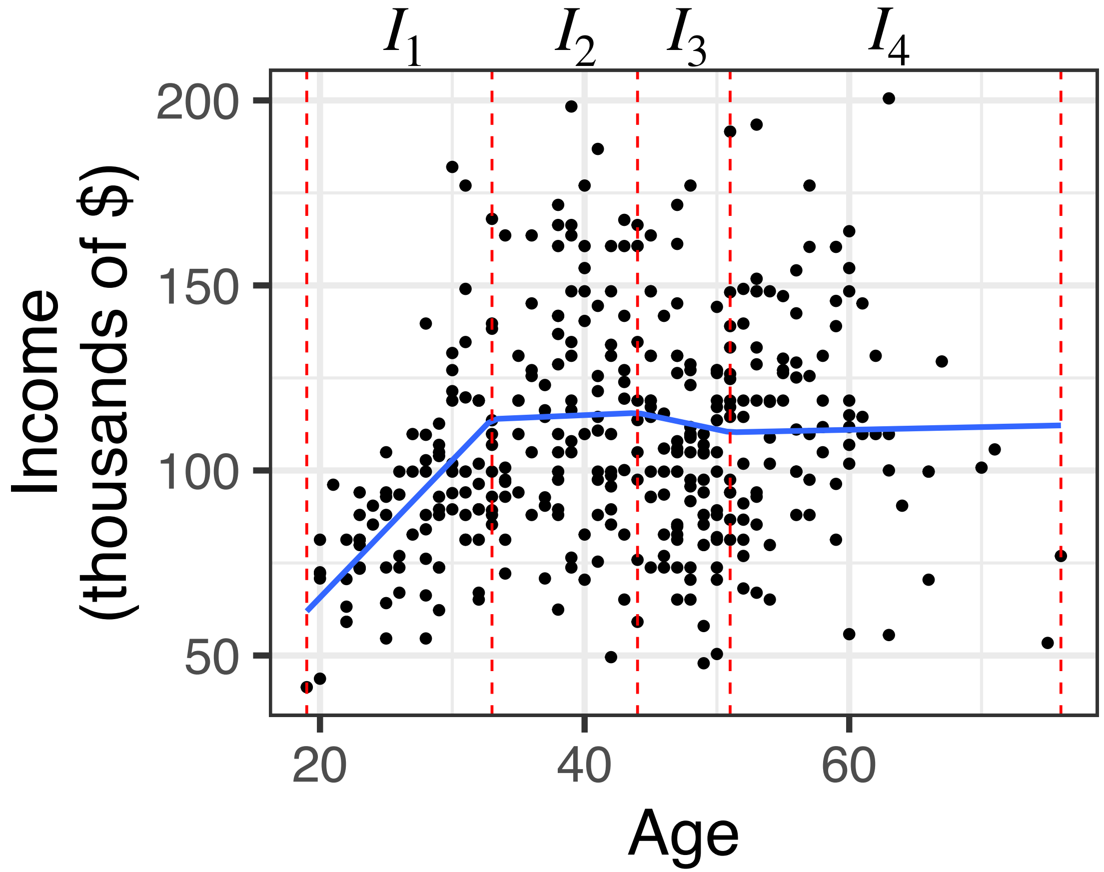
Piece-wise polynomial (piece-wise quadratic)

$$\text{income} = (\beta_{01} + \beta_{11}\text{age} + \beta_{21}\text{age}^2) \cdot 1(\text{age} \in I_1) + \cdots + (\ldots) \cdot 1(\text{age} \in I_4) + \epsilon$$



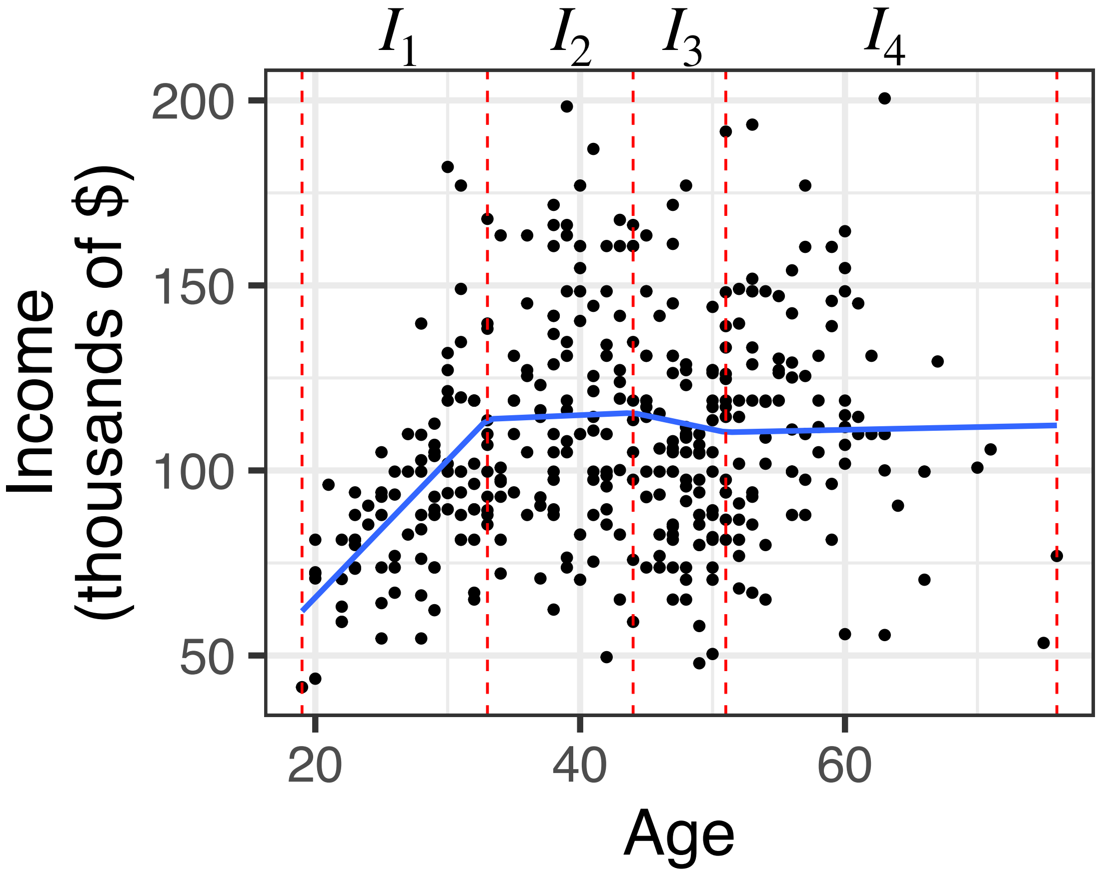
Spline (piece-wise linear)

$$\text{income} = (\beta_{01} + \beta_{11}\text{age}) \cdot 1(\text{age} \in I_1) + \cdots + (\beta_{04} + \beta_{14}\text{age}) \cdot 1(\text{age} \in I_4) + \epsilon$$



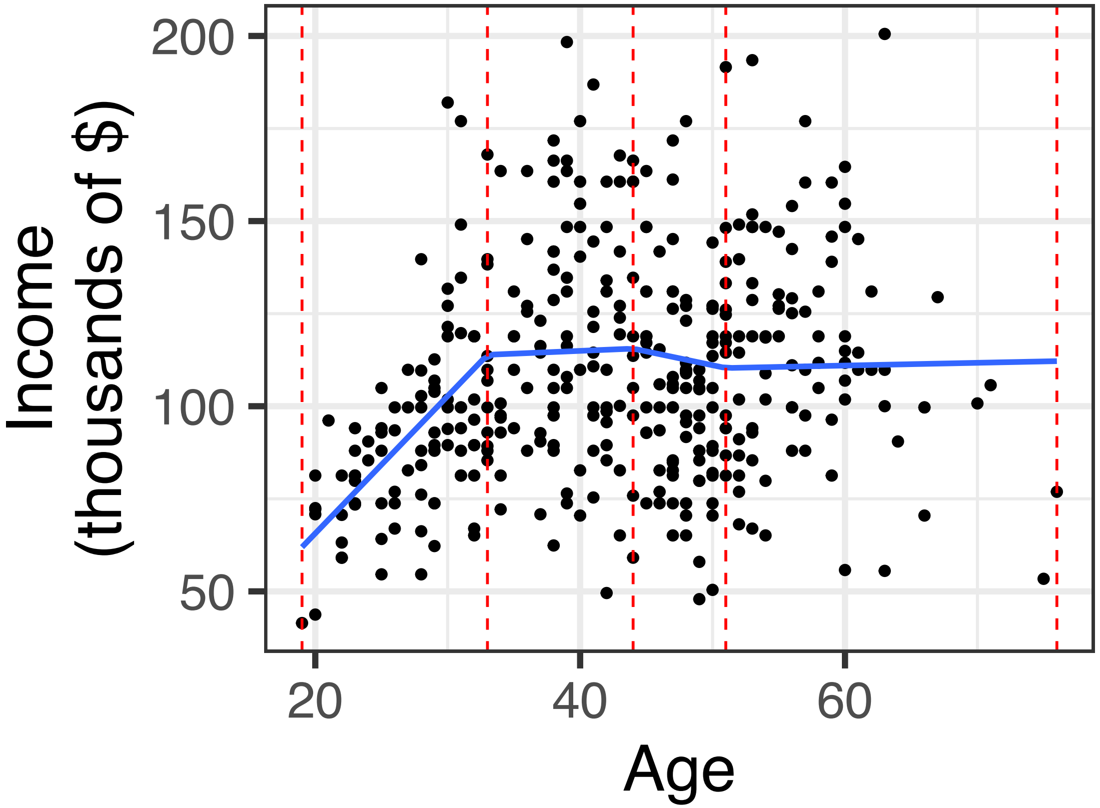
Spline (piece-wise linear)

$$\text{income} = (\beta_{01} + \beta_{11}\text{age}) \cdot 1(\text{age} \in I_1) + \cdots + (\beta_{04} + \beta_{14}\text{age}) \cdot 1(\text{age} \in I_4) + \epsilon$$



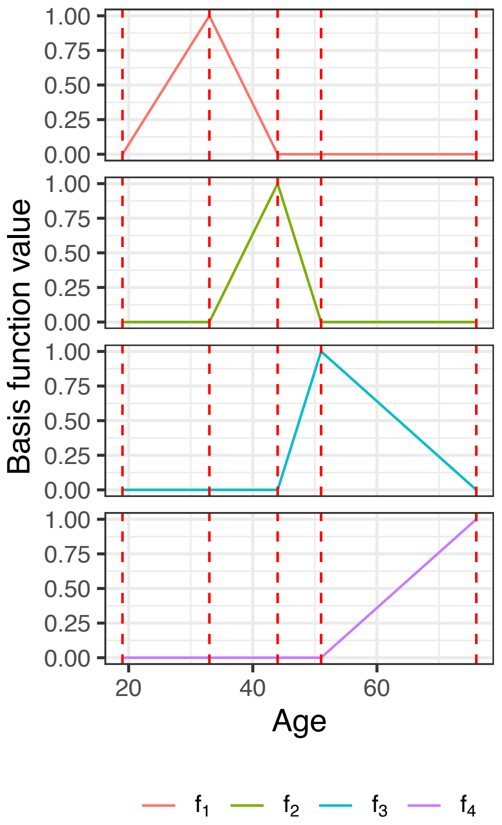
Spline (piece-wise linear)

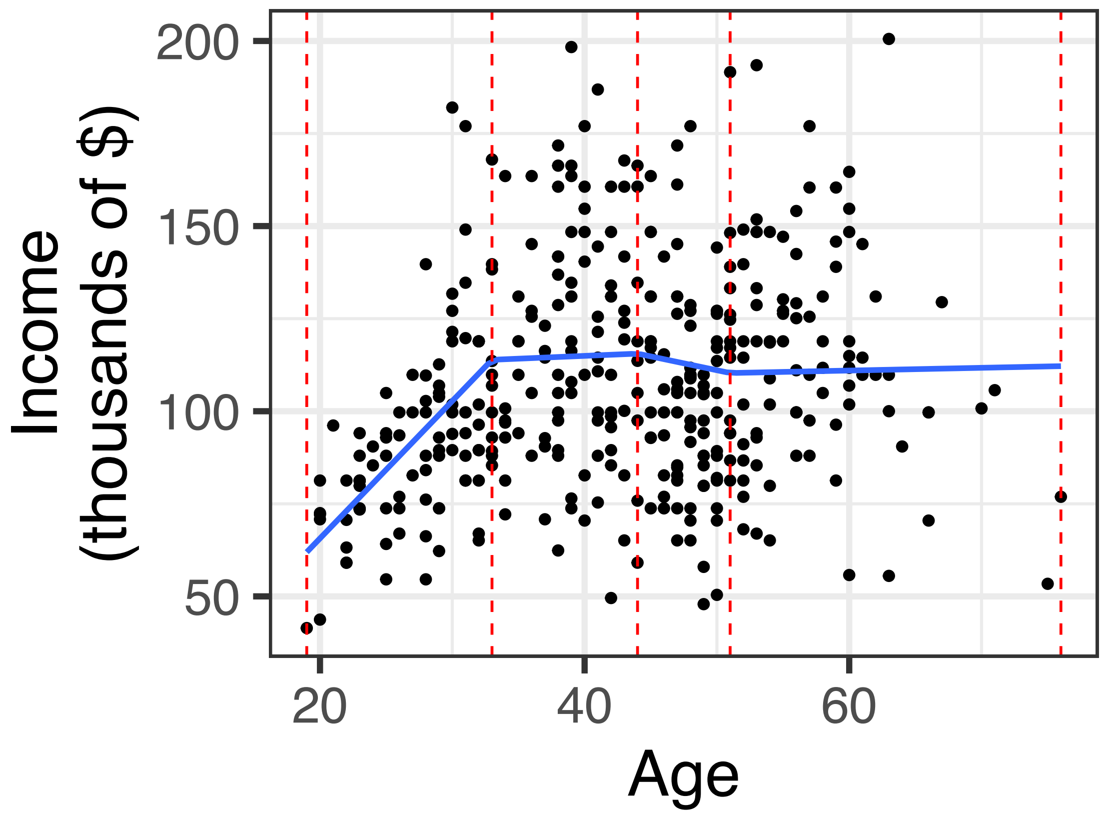
$$\text{income} = \beta_0 + \beta_1 \cdot g_1(\text{age}) + \cdots + \beta_{p-1} \cdot g_{p-1}(\text{age}) + \epsilon$$



Spline (piece-wise linear)

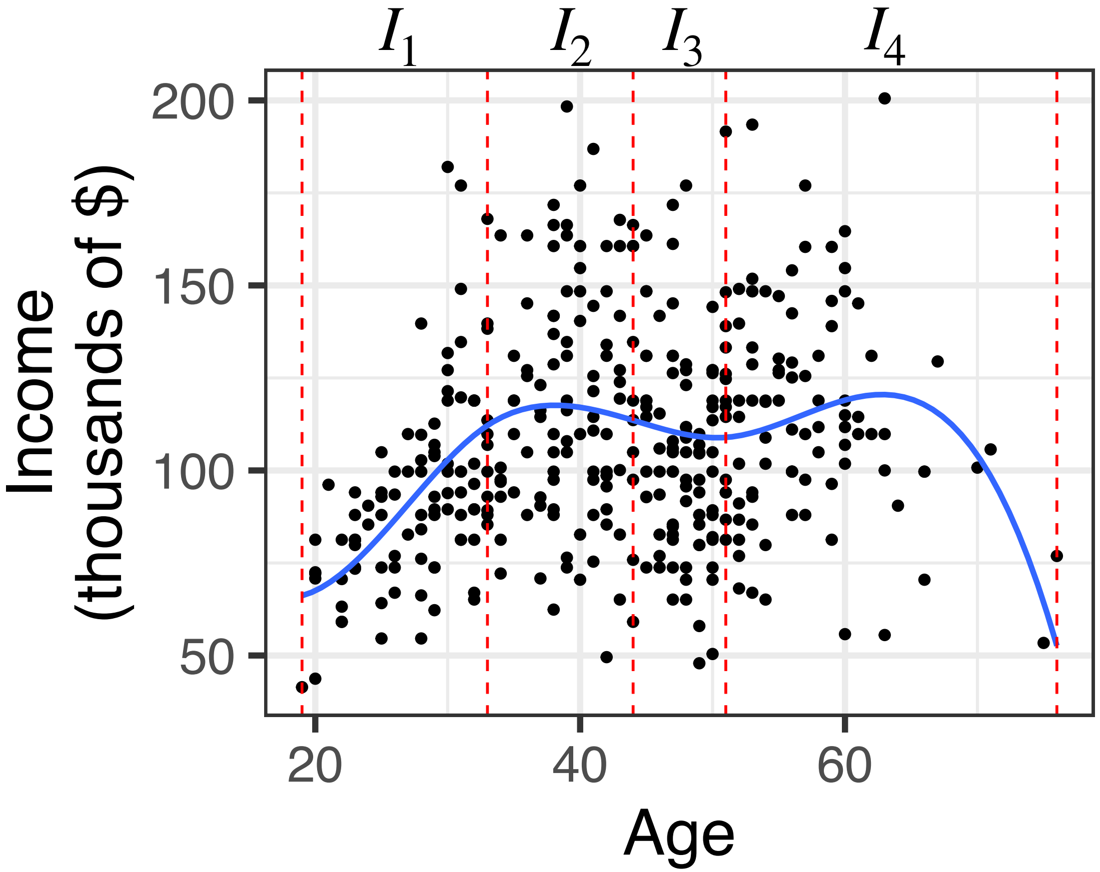
$$\text{income} = \beta_0 + \beta_1 \cdot g_1(\text{age}) + \cdots + \beta_{p-1} \cdot g_{p-1}(\text{age}) + \epsilon$$





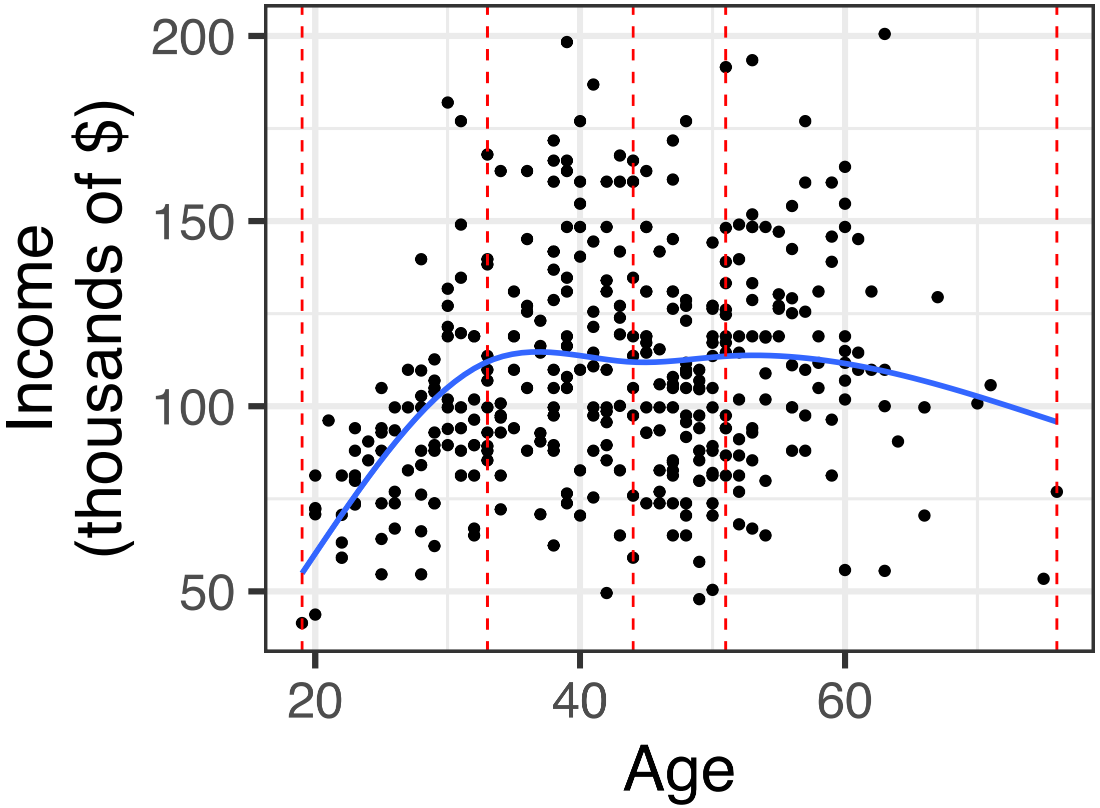
Spline (piece-wise cubic)

$$\text{income} = \beta_0 + \beta_1 \cdot g_1(\text{age}) + \cdots + \beta_{p-1} \cdot g_{p-1}(\text{age}) + \epsilon$$



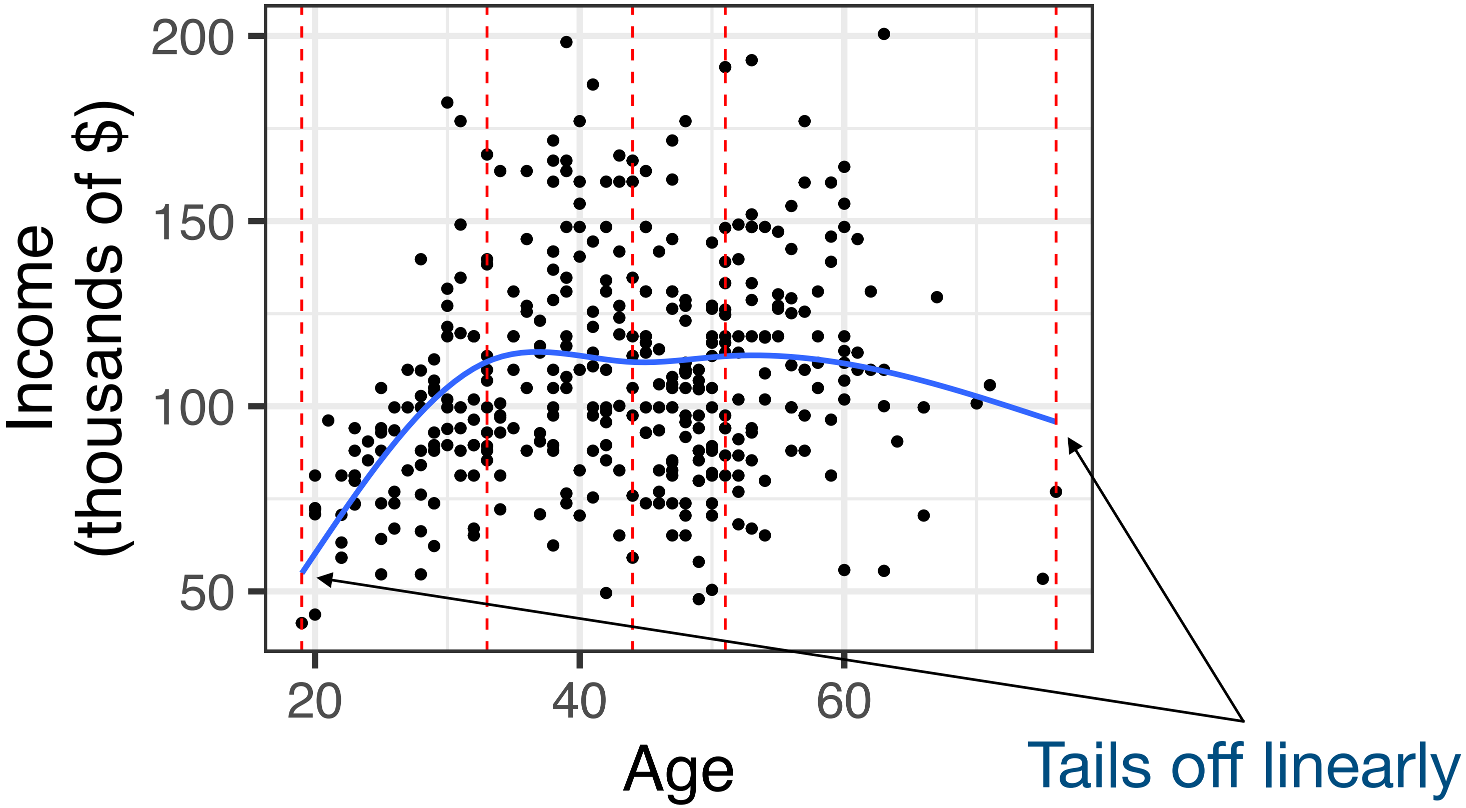
Natural cubic spline (with 5 total knots)

$$\text{income} = \beta_0 + \beta_1 \cdot g_1(\text{age}) + \cdots + \beta_{p-1} \cdot g_{p-1}(\text{age}) + \epsilon$$



Natural cubic spline (with 5 total knots)

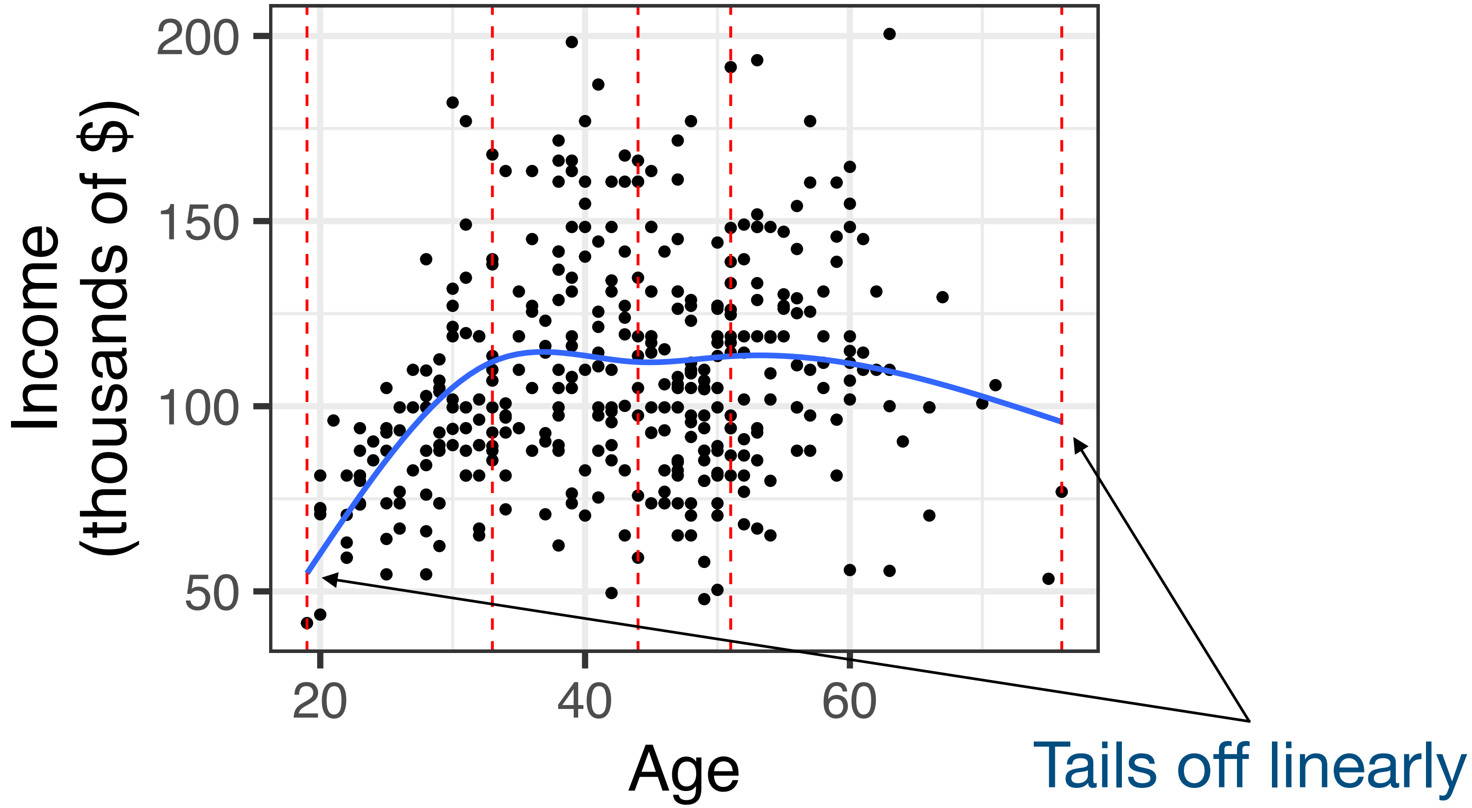
$$\text{income} = \beta_0 + \beta_1 \cdot g_1(\text{age}) + \cdots + \beta_{p-1} \cdot g_{p-1}(\text{age}) + \epsilon$$



Natural cubic spline (with 5 total knots)

$$\text{income} = \beta_0 + \beta_1 \cdot g_1(\text{age}) + \cdots + \beta_{p-1} \cdot g_{p-1}(\text{age}) + \epsilon$$

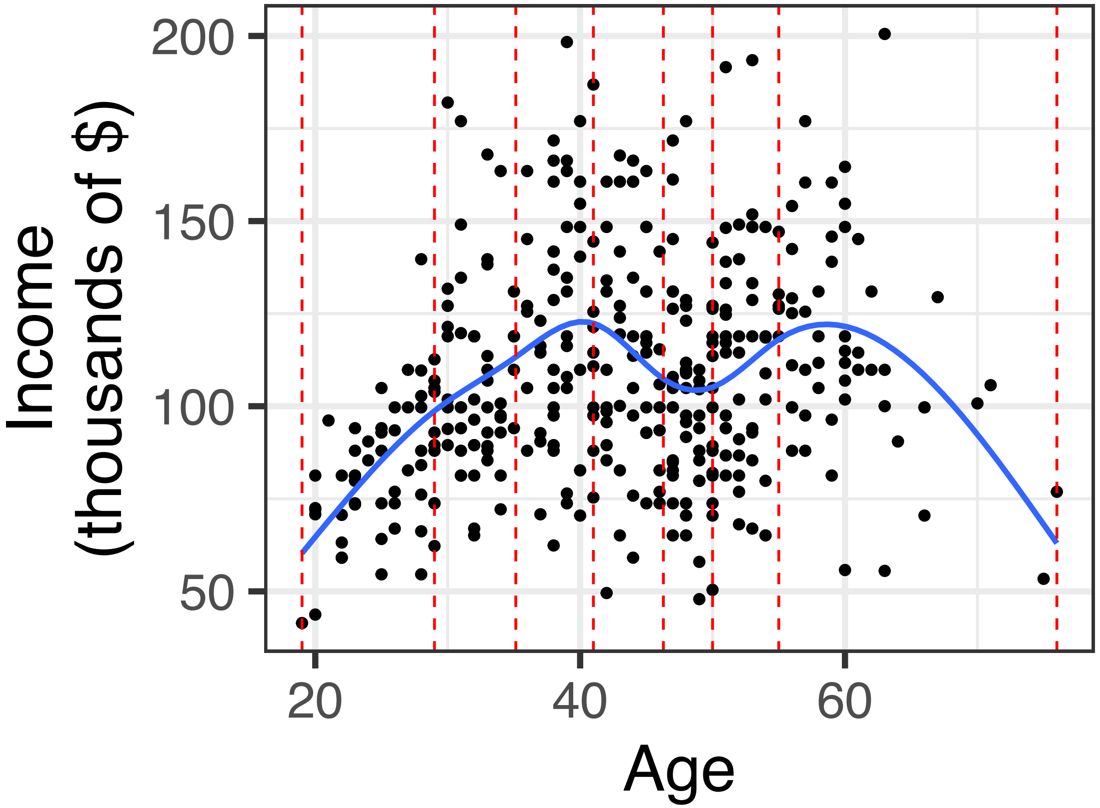
The preferred way to fit smooth curves to data.



Natural cubic spline (with 8 total knots)

$$\text{income} = \beta_0 + \beta_1 \cdot g_1(\text{age}) + \cdots + \beta_{p-1} \cdot g_{p-1}(\text{age}) + \epsilon$$

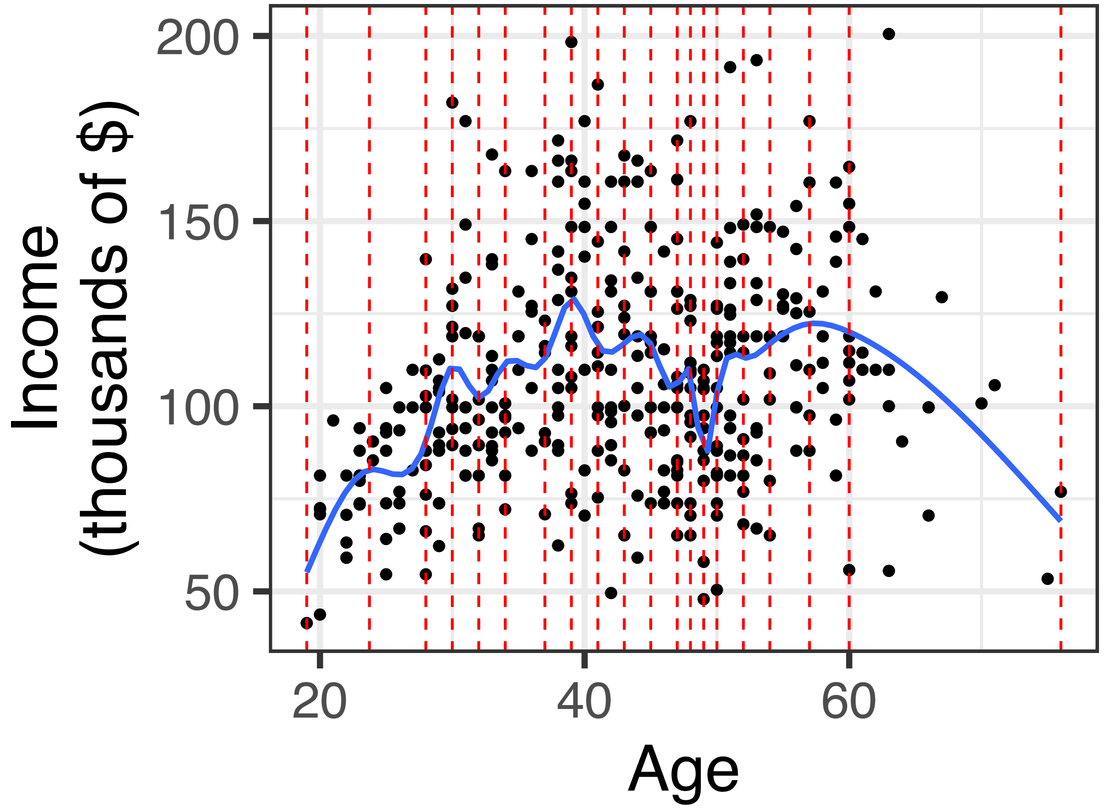
The preferred way to fit smooth curves to data.



Natural cubic spline (with 20 total knots)

$$\text{income} = \beta_0 + \beta_1 \cdot g_1(\text{age}) + \cdots + \beta_{p-1} \cdot g_{p-1}(\text{age}) + \epsilon$$

The preferred way to fit smooth curves to data.



Fitting linear regression models via least squares

Fitting linear regression models via least squares

All of these models are linear regression models of the form

$$y = \beta_0 + \beta_1 \cdot g_1(x) + \cdots + \beta_{p-1} \cdot g_{p-1}(x) + \epsilon.$$

Fitting linear regression models via least squares

All of these models are linear regression models of the form

$$y = \beta_0 + \beta_1 \cdot g_1(x) + \cdots + \beta_{p-1} \cdot g_{p-1}(x) + \epsilon.$$

Given training data $(x_1, y_1), \ldots, (x_n, y_n)$, they are fit using least squares:

$$(\widehat{\beta}_0, \ldots, \widehat{\beta}_{p-1}) \equiv \underset{\beta_0, \ldots, \beta_{p-1}}{\arg\min} \sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 \cdot g_1(x_i) + \cdots + \beta_{p-1} \cdot g_{p-1}(x_i)))^2,$$

Fitting linear regression models via least squares

All of these models are linear regression models of the form

$$y = \beta_0 + \beta_1 \cdot g_1(x) + \cdots + \beta_{p-1} \cdot g_{p-1}(x) + \epsilon.$$

Given training data $(x_1, y_1), \ldots, (x_n, y_n)$, they are fit using least squares:

$$(\widehat{\beta}_0, \ldots, \widehat{\beta}_{p-1}) \equiv \underset{\beta_0, \ldots, \beta_{p-1}}{\arg\min} \sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 \cdot g_1(x_i) + \cdots + \beta_{p-1} \cdot g_{p-1}(x_i)))^2,$$

i.e. $(\widehat{\beta}_0, \ldots, \widehat{\beta}_{p-1})$ is the coefficient vector minimizing the the squared distance between the training responses y_i and their predictions.

The model complexity of least squares

The model complexity of least squares

The model fitting process involves a search over p parameters:

$$(\widehat{\beta}_0, \ldots, \widehat{\beta}_{p-1}) \equiv \underset{\beta_0,\ldots,\beta_{p-1}}{\arg\min} \sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 \cdot g_1(x_i) + \cdots + \beta_{p-1} \cdot g_{p-1}(x_i)))^2.$$

The model complexity of least squares

The model fitting process involves a search over p parameters:

$$(\widehat{\beta}_0, \ldots, \widehat{\beta}_{p-1}) \equiv \underset{\beta_0,\ldots,\beta_{p-1}}{\arg\min} \sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 \cdot g_1(x_i) + \cdots + \beta_{p-1} \cdot g_{p-1}(x_i)))^2.$$

The complexity of least squares for the model

The model complexity of least squares

The model fitting process involves a search over p parameters:

$$(\widehat{\beta}_0, \ldots, \widehat{\beta}_{p-1}) \equiv \underset{\beta_0,\ldots,\beta_{p-1}}{\arg\min} \sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 \cdot g_1(x_i) + \cdots + \beta_{p-1} \cdot g_{p-1}(x_i)))^2.$$

The complexity of least squares for the model

$$y = \beta_0 + \beta_1 \cdot g_1(x) + \cdots + \beta_{p-1} \cdot g_{p-1}(x) + \epsilon$$

The model complexity of least squares

The model fitting process involves a search over p parameters:

$$(\widehat{\beta}_0, \ldots, \widehat{\beta}_{p-1}) \equiv \underset{\beta_0,\ldots,\beta_{p-1}}{\arg\min} \sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 \cdot g_1(x_i) + \cdots + \beta_{p-1} \cdot g_{p-1}(x_i)))^2.$$

The complexity of least squares for the model

$$y = \beta_0 + \beta_1 \cdot g_1(x) + \cdots + \beta_{p-1} \cdot g_{p-1}(x) + \epsilon$$

is quantified by its *degrees of freedom (df),* the number of free parameters β_j.

The model complexity of least squares

The model fitting process involves a search over p parameters:

$$(\hat{\beta}_0, \ldots, \hat{\beta}_{p-1}) \equiv \underset{\beta_0,\ldots,\beta_{p-1}}{\arg\min} \sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 \cdot g_1(x_i) + \cdots + \beta_{p-1} \cdot g_{p-1}(x_i)))^2.$$

The complexity of least squares for the model

$$y = \beta_0 + \beta_1 \cdot g_1(x) + \cdots + \beta_{p-1} \cdot g_{p-1}(x) + \epsilon$$

is quantified by its *degrees of freedom (df),* the number of free parameters β_j.

For example, the model above has p degrees of freedom.

The complexity of polynomial and spline fits

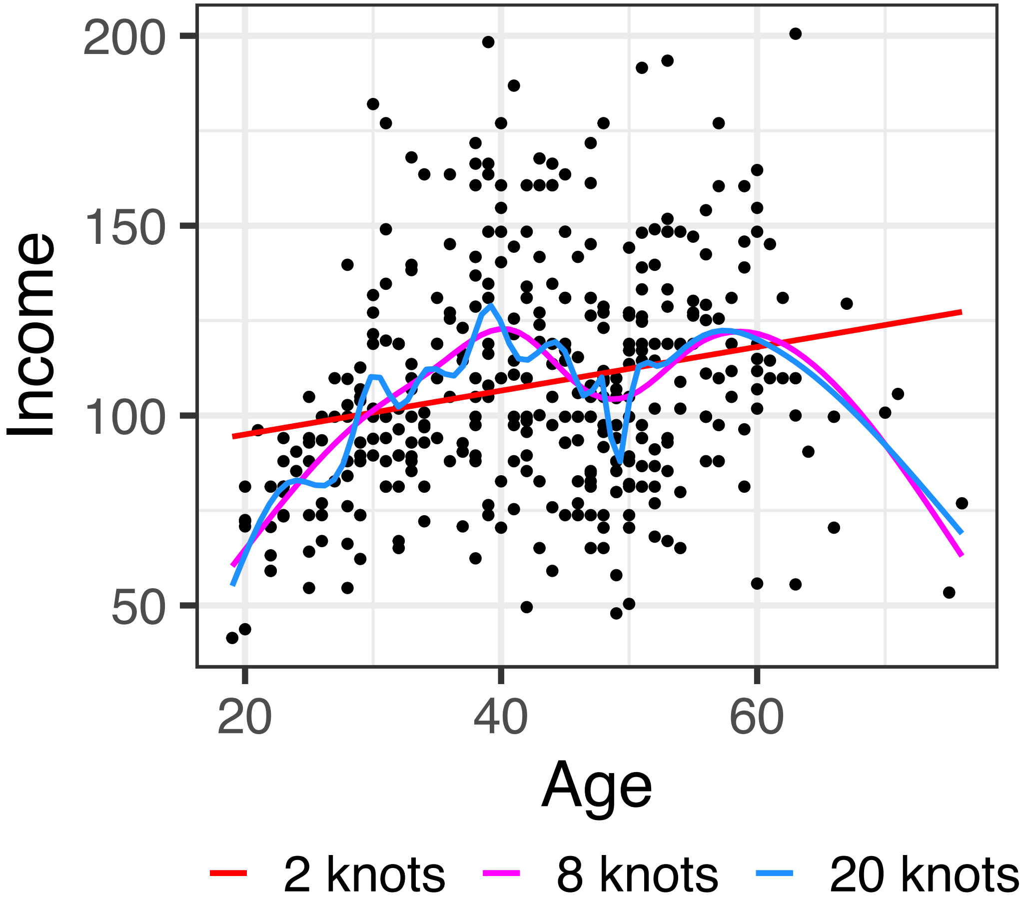
The complexity of polynomial and spline fits

- Polynomials of degree p have $p + 1$ degrees of freedom.

The complexity of polynomial and spline fits

- Polynomials of degree p have $p + 1$ degrees of freedom.

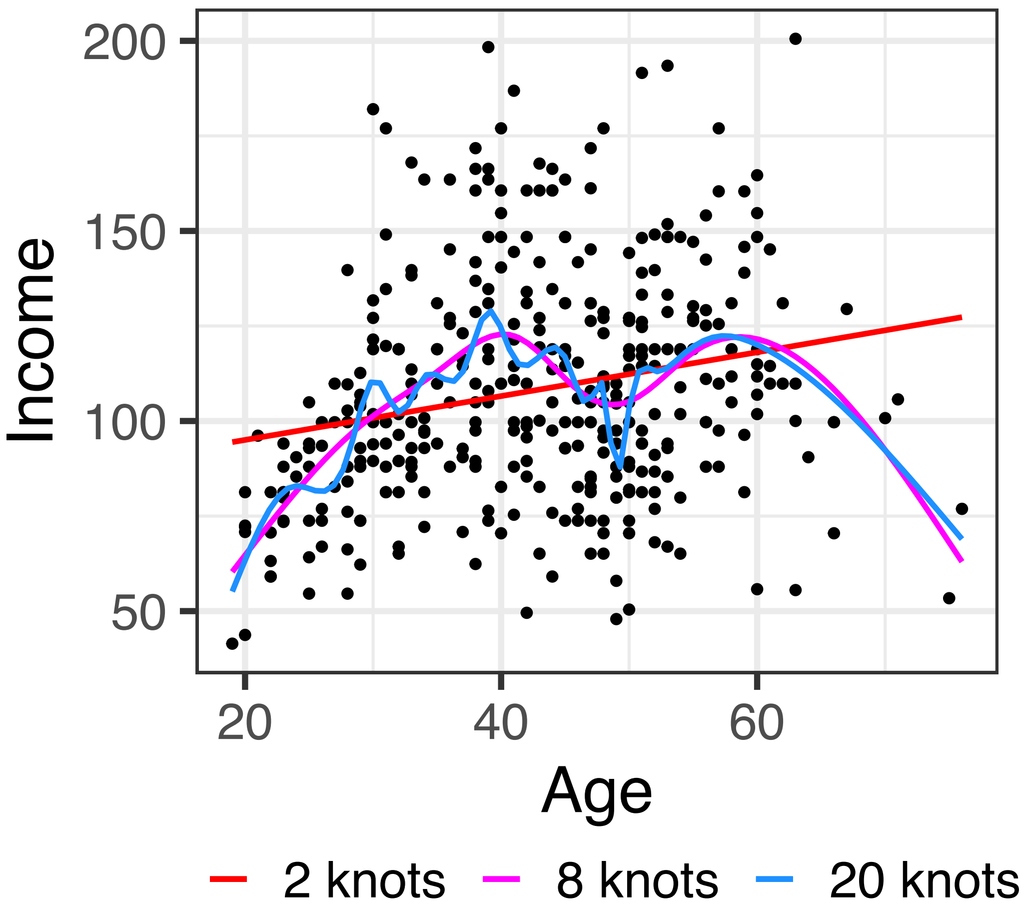
- Natural cubic splines with K total knots have K degrees of freedom.



The complexity of polynomial and spline fits

- Polynomials of degree p have $p + 1$ degrees of freedom.

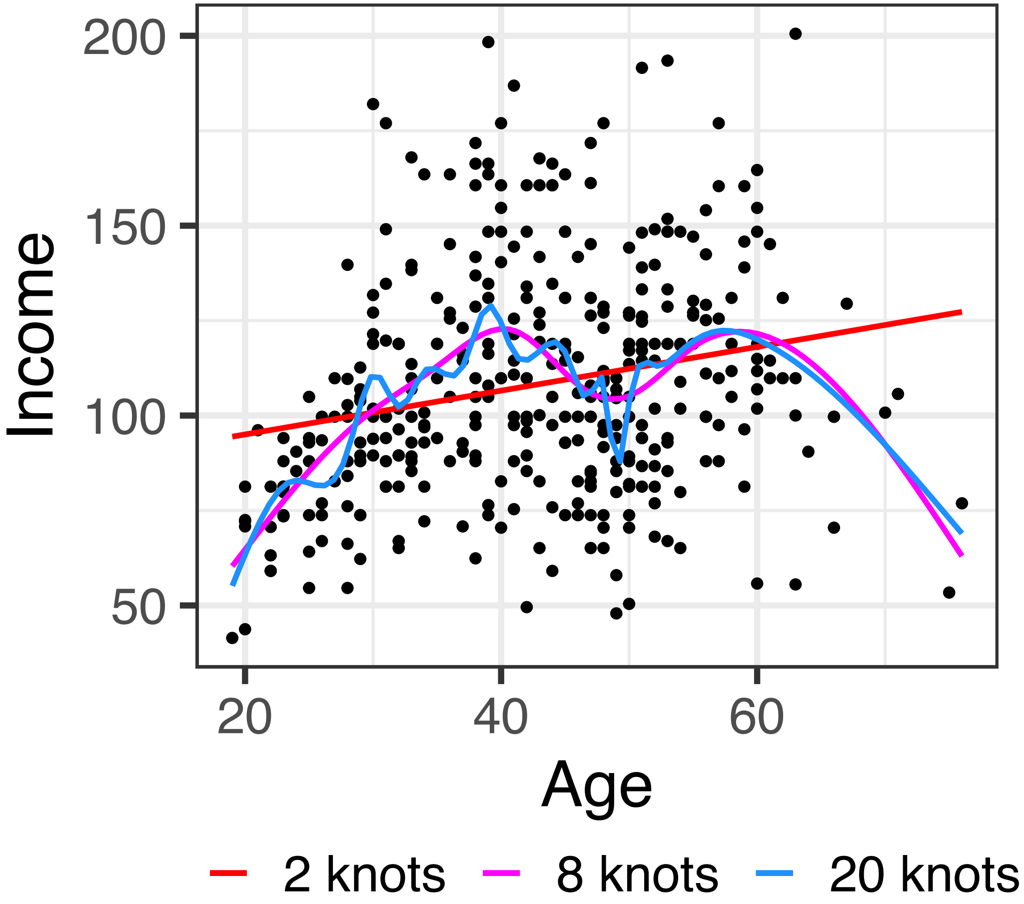
- Natural cubic splines with K total knots have K degrees of freedom.*



*Caution: Sometimes the intercept is excluded from the spline definition, so a spline with K total knots is sometimes considered to have df = $K - 1$.

The complexity of polynomial and spline fits

- Polynomials of degree p have $p + 1$ degrees of freedom.
- Natural cubic splines with K total knots have K degrees of freedom.*
- Model complexity has an important effect on predictive performance:



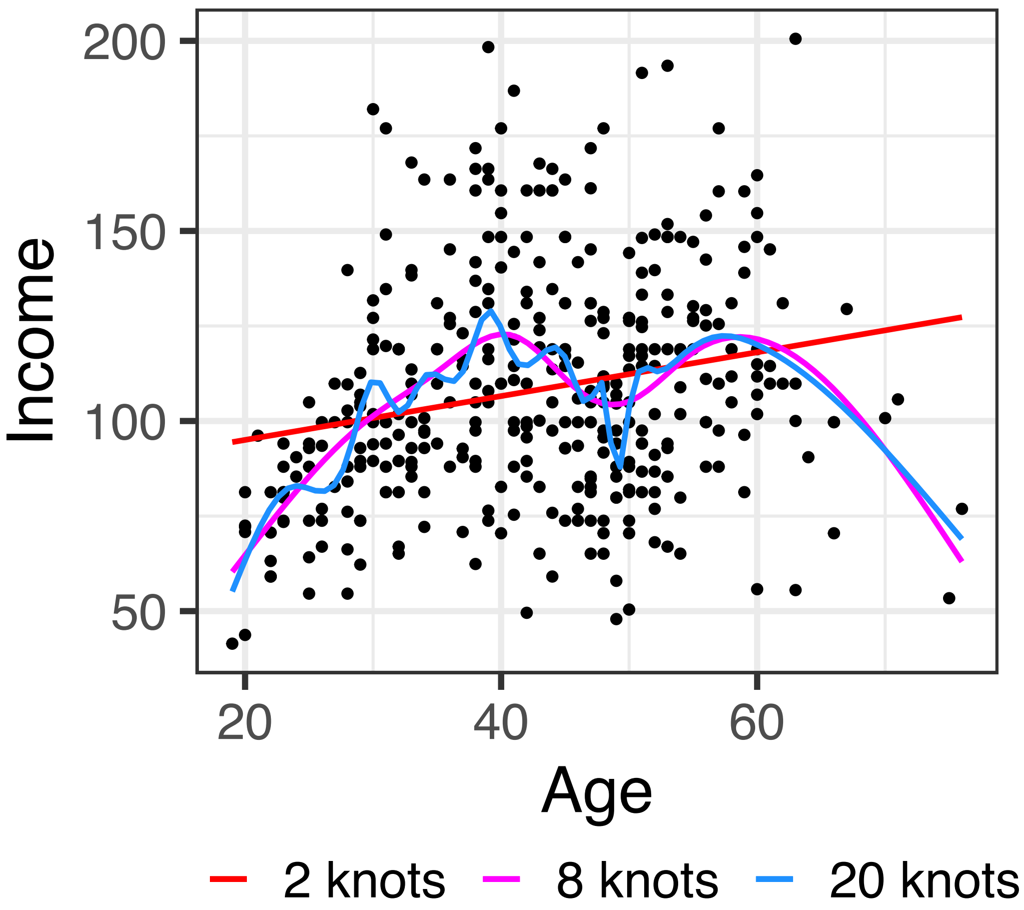
*Caution: Sometimes the intercept is excluded from the spline definition, so a spline with K total knots is sometimes considered to have df = $K - 1$.

The complexity of polynomial and spline fits

- Polynomials of degree p have $p + 1$ degrees of freedom.

- Natural cubic splines with K total knots have K degrees of freedom.*

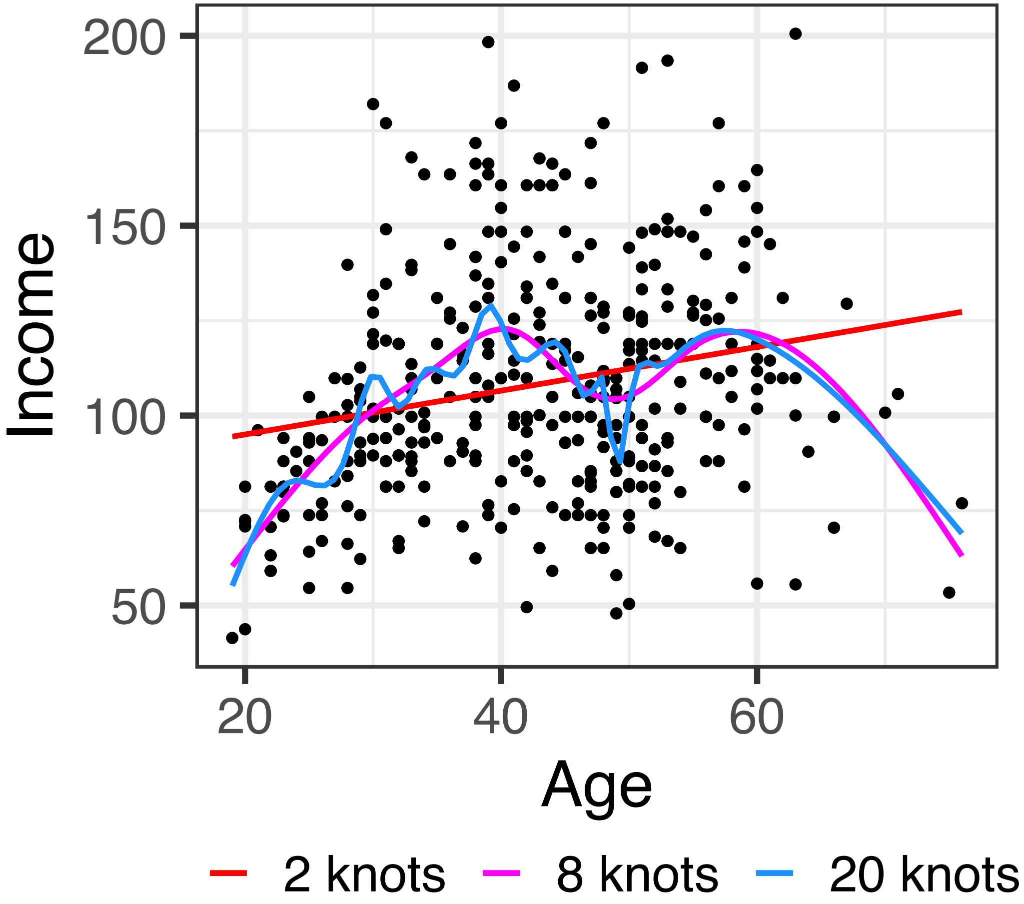
- Model complexity has an important effect on predictive performance:
    - Too flexible →
      too sensitive to noise in training data



*Caution: Sometimes the intercept is excluded from the spline definition, so a spline with K total knots is sometimes considered to have df = $K - 1$.

The complexity of polynomial and spline fits

- Polynomials of degree p have $p + 1$ degrees of freedom.
- Natural cubic splines with K total knots have K degrees of freedom.*
- Model complexity has an important effect on predictive performance:
    - Too flexible → too sensitive to noise in training data
    - Not flexible enough → can't capture the underlying trend



*Caution: Sometimes the intercept is excluded from the spline definition, so a spline with K total knots is sometimes considered to have df = $K - 1$.

Prediction performance

Prediction performance

Construct predictive model $\hat{f}$ based on training data $(X_1^{\text{train}}, Y_1^{\text{train}}), \ldots, (X_n^{\text{train}}, Y_n^{\text{train}})$.

Prediction performance

Construct predictive model $\hat{f}$ based on training data $(X_1^{\text{train}}, Y_1^{\text{train}}), \ldots, (X_n^{\text{train}}, Y_n^{\text{train}})$.

The root mean squared training error is

$$\text{Training RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(Y_i^{\text{train}} - \hat{f}(X_i^{\text{train}}))^2}.$$

Prediction performance

Construct predictive model $\hat{f}$ based on training data $(X_1^{\text{train}}, Y_1^{\text{train}}), \dots, (X_n^{\text{train}}, Y_n^{\text{train}})$.

The root mean squared training error is

$$\text{Training RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(Y_i^{\text{train}} - \hat{f}(X_i^{\text{train}}))^2}.$$

Will deploy $\hat{f}$ on test data $X_1^{\text{test}}, \dots, X_N^{\text{test}}$ to guess $\widehat{Y}_i^{\text{test}} = \hat{f}(X_i^{\text{test}})$ for each i. Each X_i^{test} comes with a response Y_i^{test}, unknown to the predictive model.

Prediction performance

Construct predictive model $\hat{f}$ based on training data $(X_1^{\text{train}}, Y_1^{\text{train}}), \dots, (X_n^{\text{train}}, Y_n^{\text{train}})$.

The root mean squared training error is

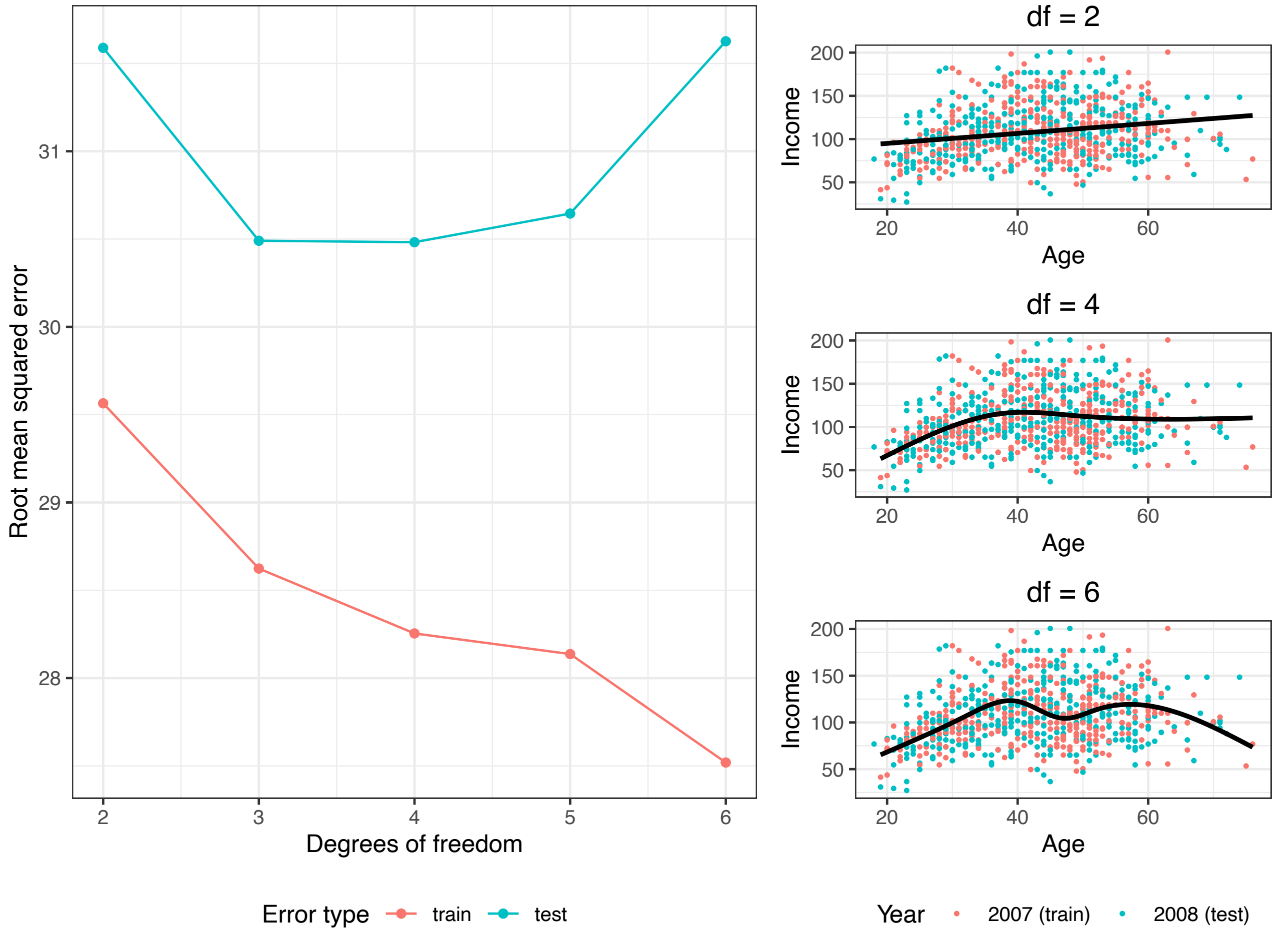
$$\text{Training RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(Y_i^{\text{train}} - \hat{f}(X_i^{\text{train}}))^2}.$$

Will deploy $\hat{f}$ on test data $X_1^{\text{test}}, \dots, X_N^{\text{test}}$ to guess $\hat{Y}_i^{\text{test}} = \hat{f}(X_i^{\text{test}})$ for each i. Each X_i^{test} comes with a response Y_i^{test}, unknown to the predictive model.

Prediction quality is quantified by test error: extent to which $Y_i^{\text{test}} \approx \hat{Y}_i^{\text{test}}$, e.g.

$$\text{Test RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(Y_i^{\text{test}} - \hat{f}(X_i^{\text{test}}))^2} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(Y_i^{\text{test}} - \hat{Y}_i^{\text{test}})^2}.$$

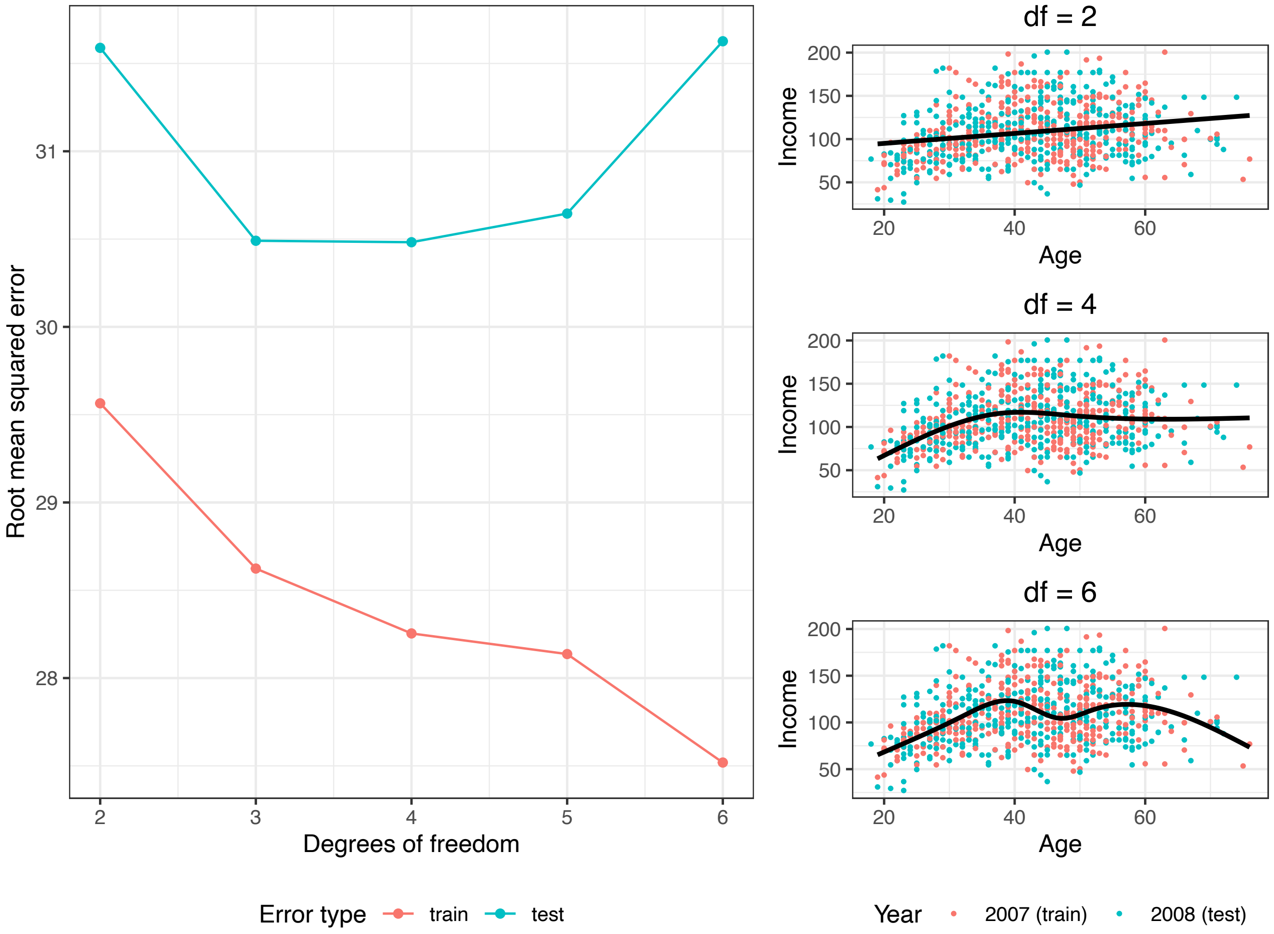
Model complexity impacts prediction performance



Model complexity impacts prediction performance

Model complexity: how closely the model $\hat{f}$ fits the training data:

$$Y_i^{\text{train}} = f(X_i^{\text{train}}) + \epsilon_i.$$

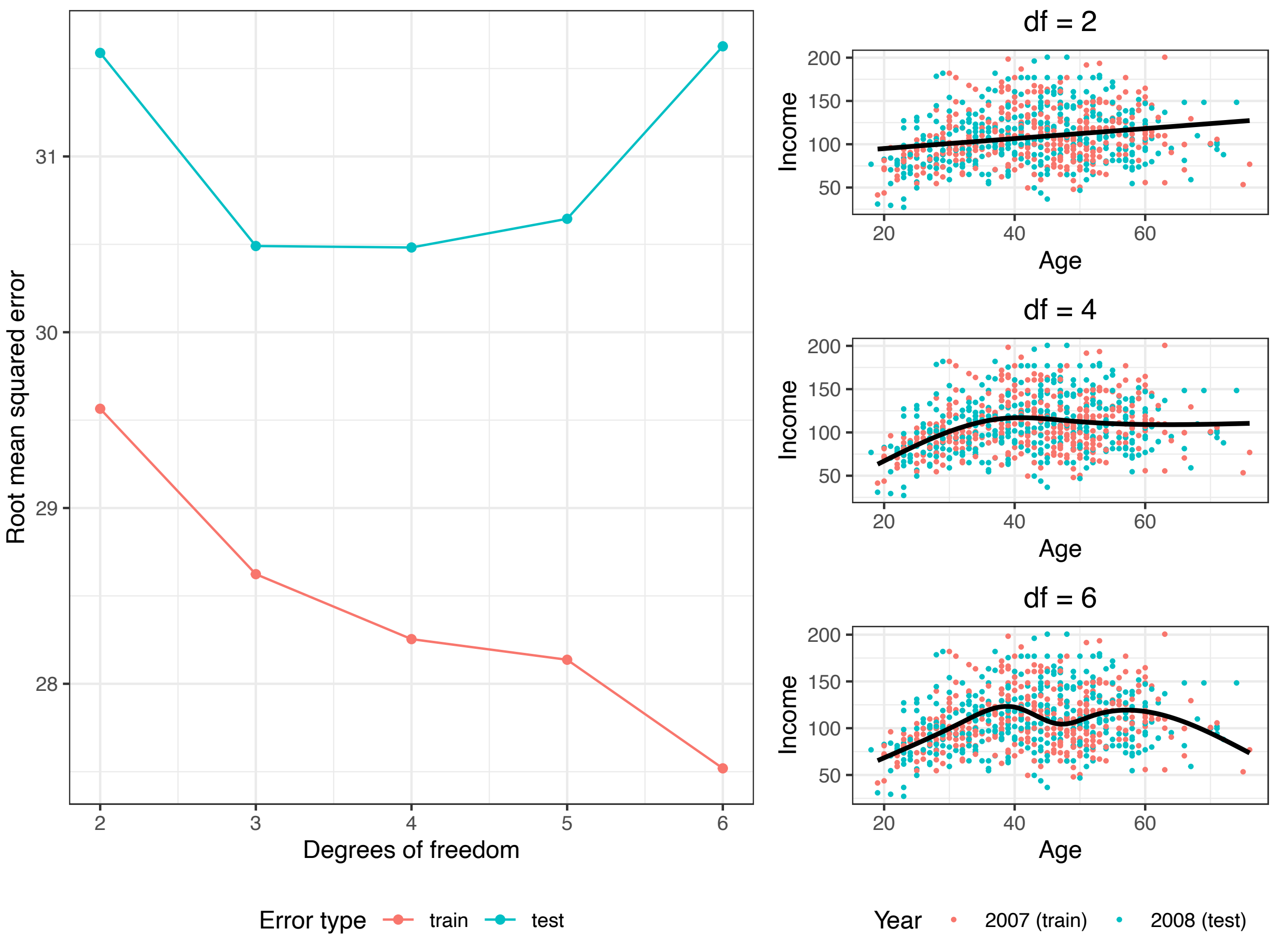


Model complexity impacts prediction performance

Model complexity: how closely the model $\hat{f}$ fits the training data:

$$Y_i^{\text{train}} = f(X_i^{\text{train}}) + \epsilon_i.$$

During training, $\hat{f}$ picks up on patterns in both f (the signal) and ϵ_i (the noise).



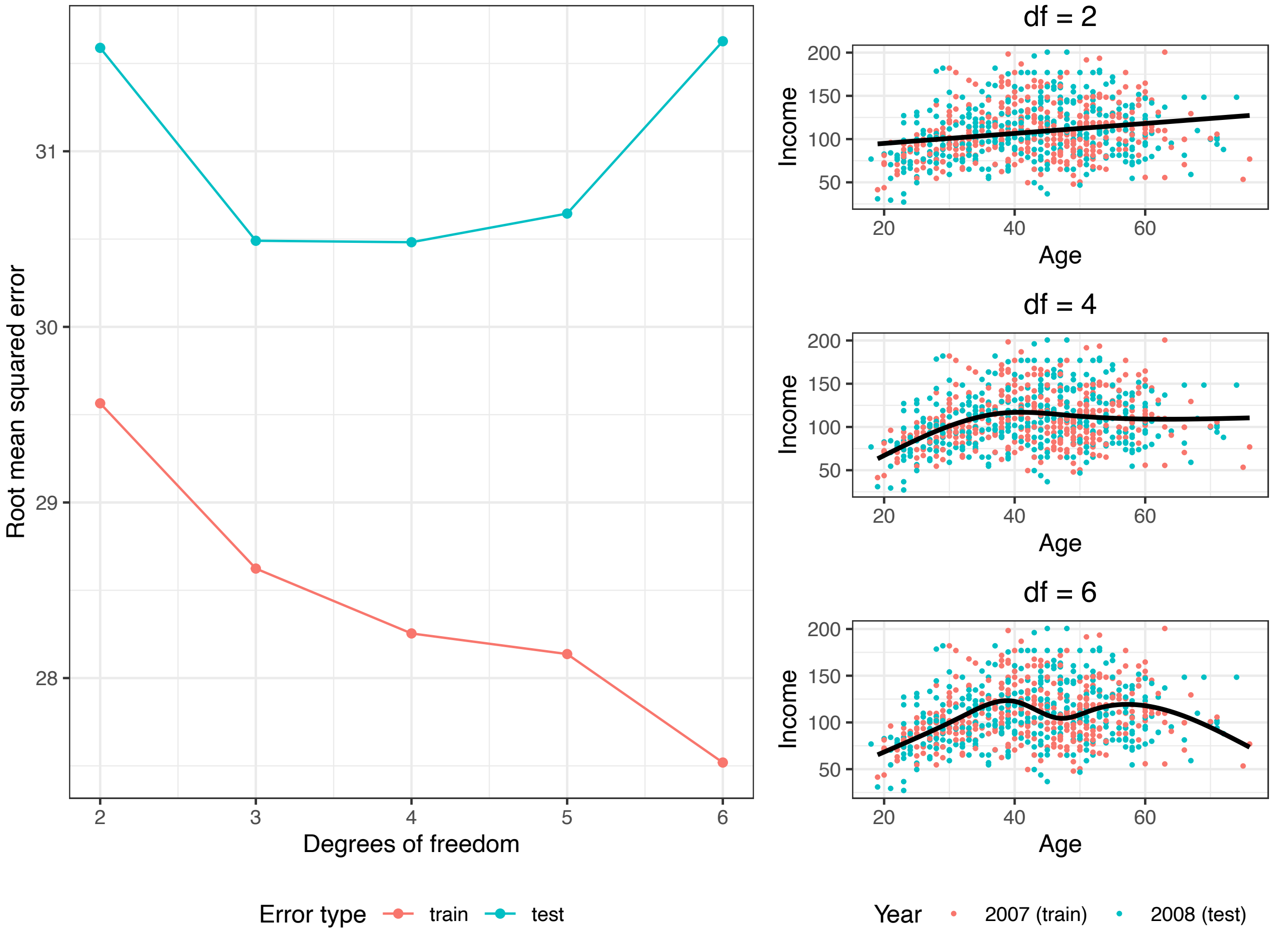
Model complexity impacts prediction performance

Model complexity: how closely the model $\hat{f}$ fits the training data:

$$Y_i^{\text{train}} = f(X_i^{\text{train}}) + \epsilon_i.$$

During training, $\hat{f}$ picks up on patterns in both f (the signal) and ϵ_i (the noise).

Training error of $\hat{f}$ decreases as we increase model complexity, but test error will be high if model complexity is too low or too high.



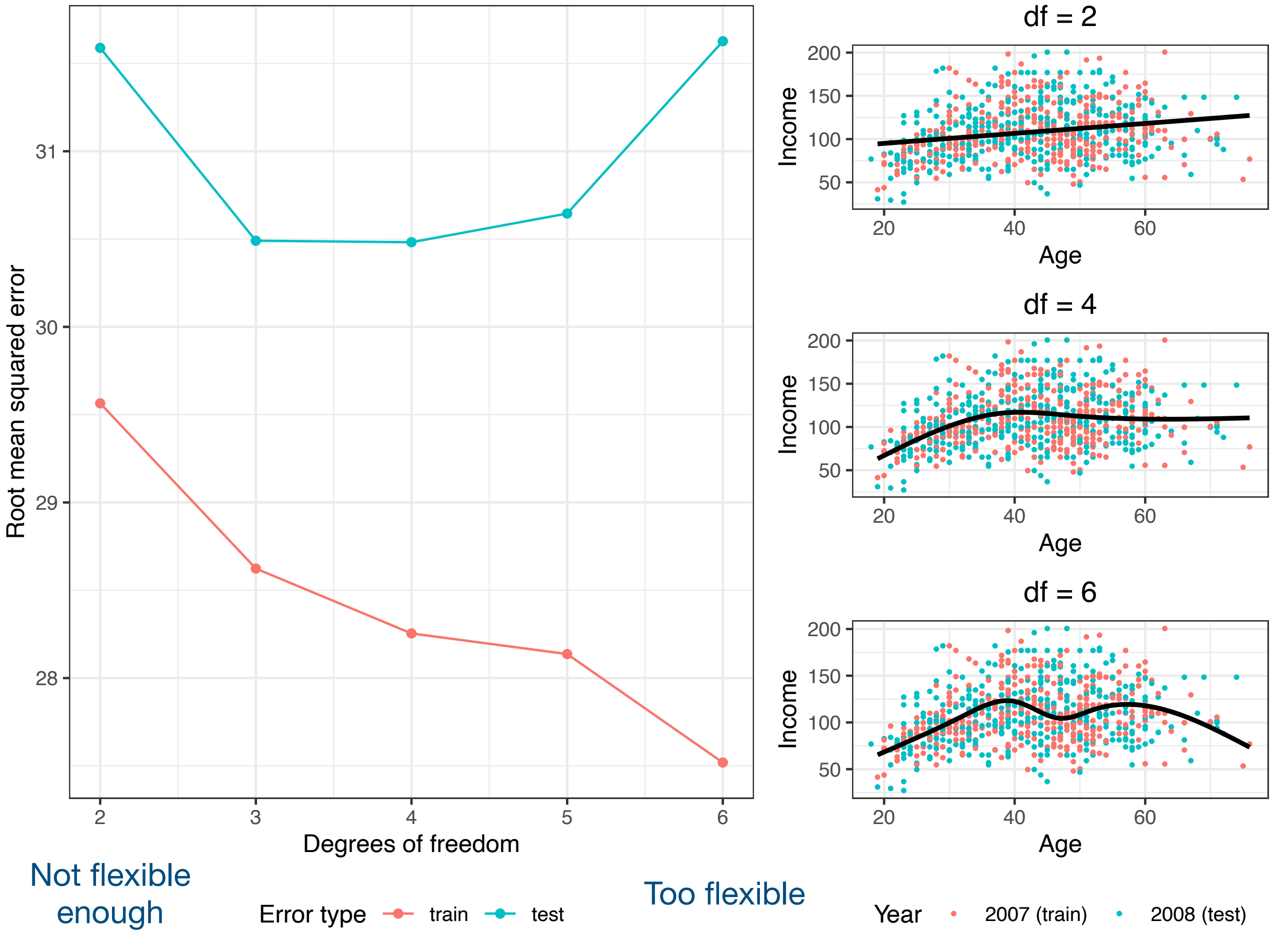
Model complexity impacts prediction performance

Model complexity: how closely the model $\hat{f}$ fits the training data:

$$Y_i^{\text{train}} = f(X_i^{\text{train}}) + \epsilon_i.$$

During training, $\hat{f}$ picks up on patterns in both f (the signal) and ϵ_i (the noise).

Training error of $\hat{f}$ decreases as we increase model complexity, but test error will be high if model complexity is too low or too high.



Model complexity impacts prediction performance

Model complexity: how closely the model $\hat{f}$ fits the training data:

$$Y_i^{\text{train}} = f(X_i^{\text{train}}) + \epsilon_i.$$

During training, $\hat{f}$ picks up on patterns in both f (the signal) and ϵ_i (the noise).

Training error of $\hat{f}$ decreases as we increase model complexity, but test error will be high if model complexity is too low or too high.

Training error is an underestimate of the test error, especially as the model complexity increases (overfitting).