

Model Complexity

STAT 4710

September 15, 2022

Rolling into Unit 2



Unit 1: Intro to modern data mining

Unit 2: Tuning predictive models

Unit 3: Regression-based methods

Unit 4: Tree-based methods

Unit 5: Deep learning

Lecture 1: Model complexity

Lecture 2: Bias-variance trade-off

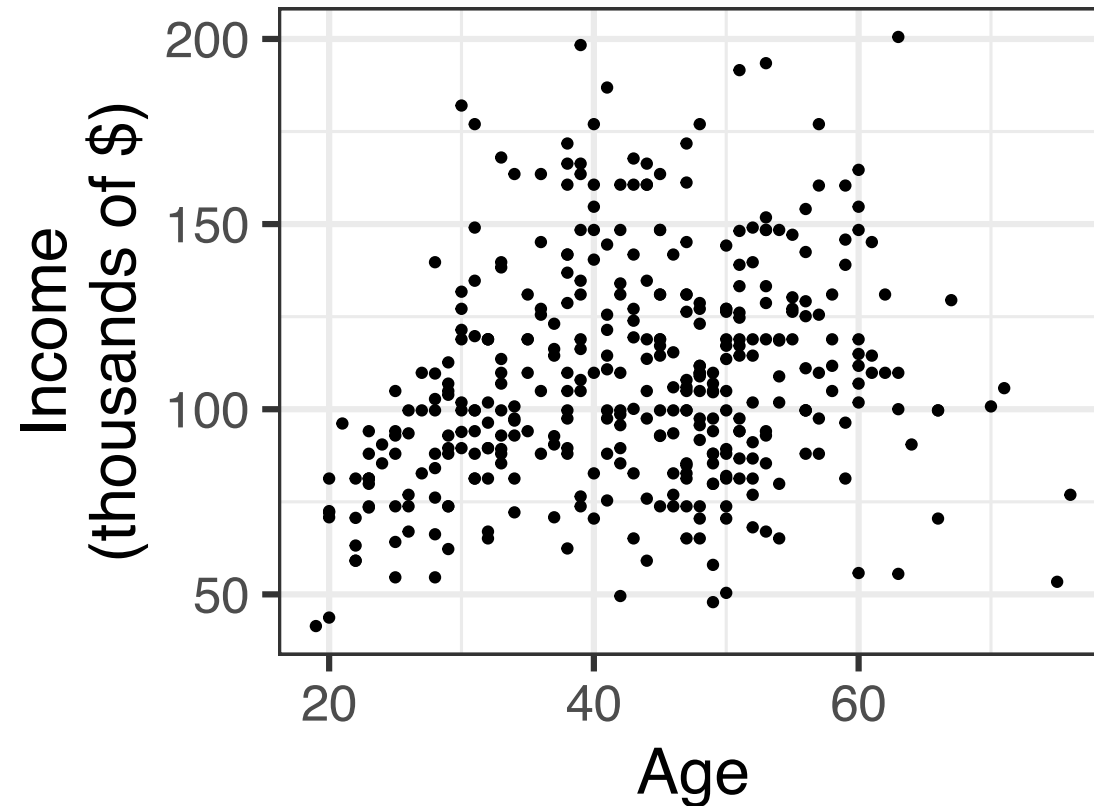
Lecture 3: Cross-validation

Lecture 4: Classification

Lecture 5: Unit review and quiz in class

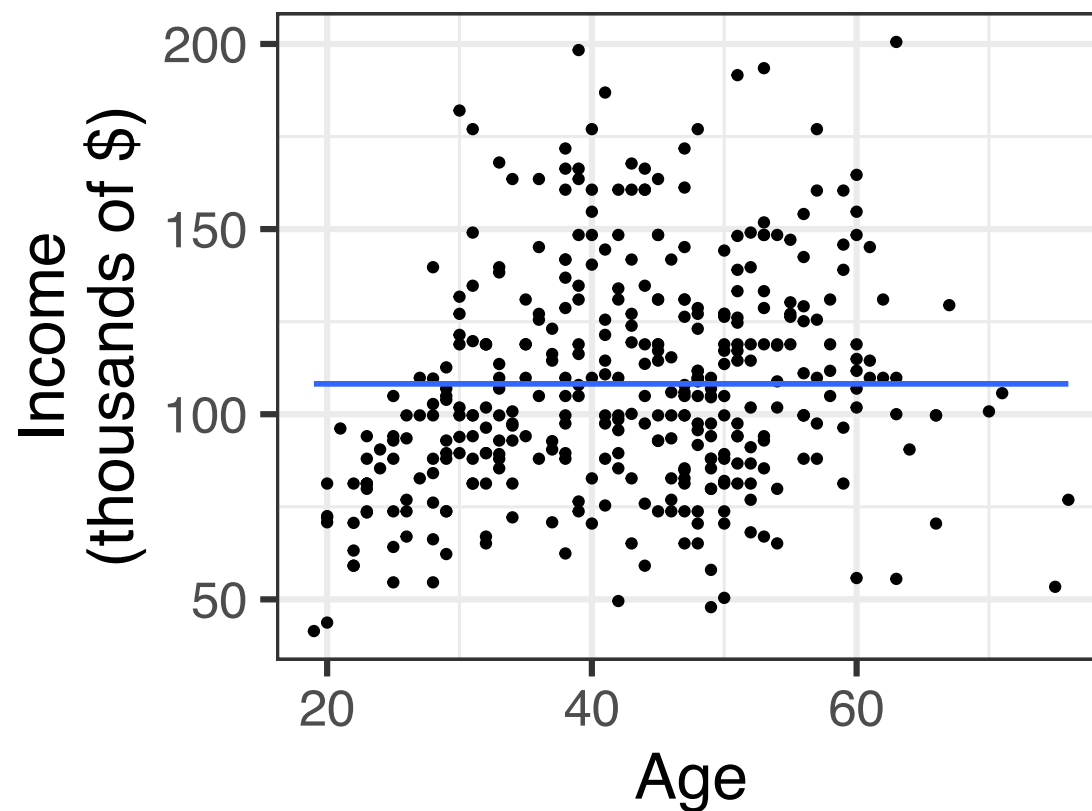
Example: Fit trend of income based on age

What does the trend look like?



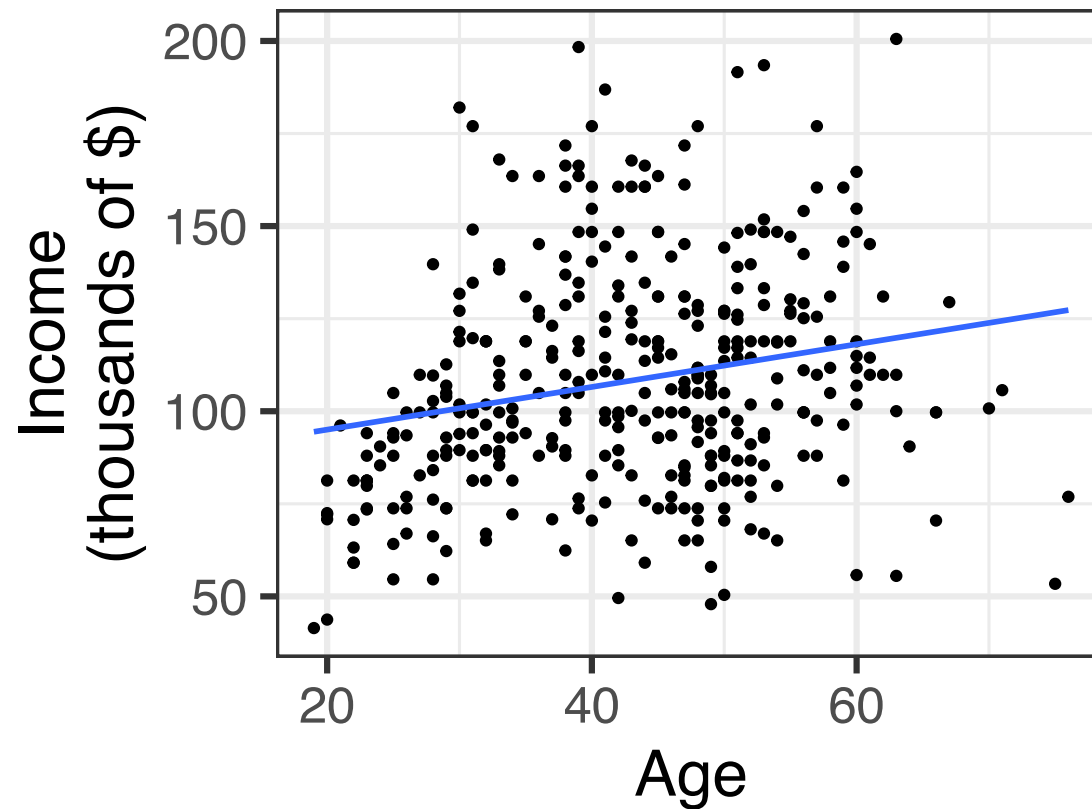
Intercept-only model (no trend)

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



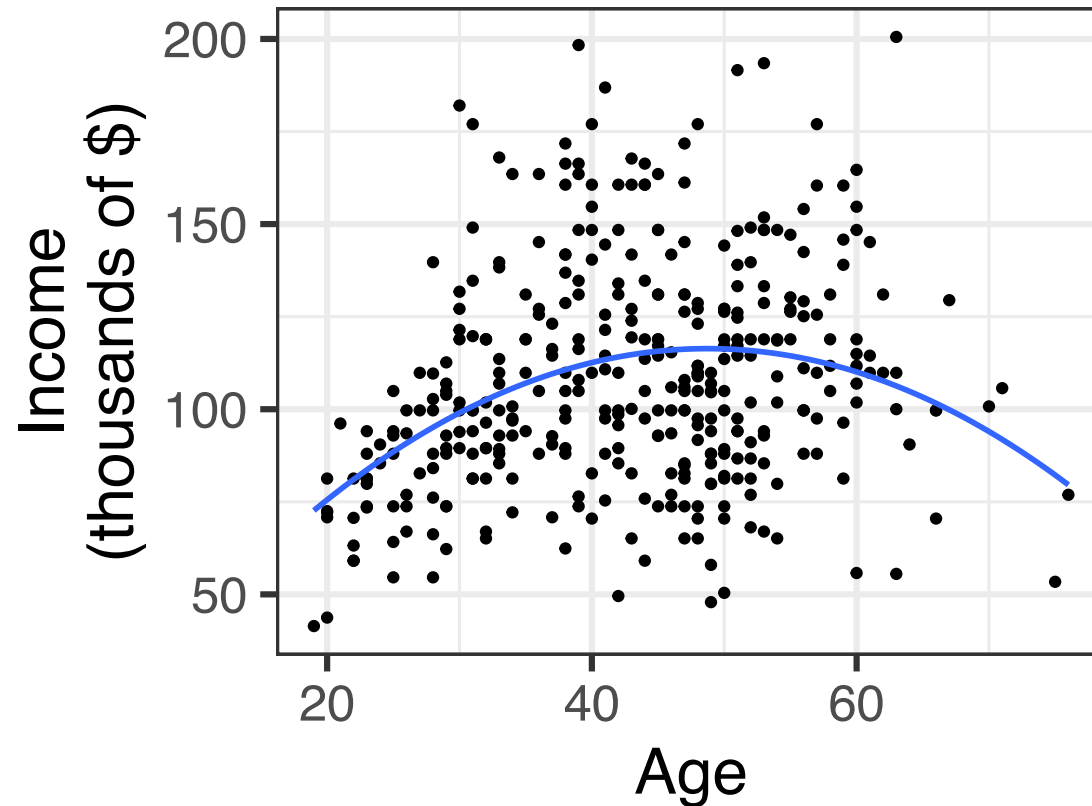
Linear model (linear trend)

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



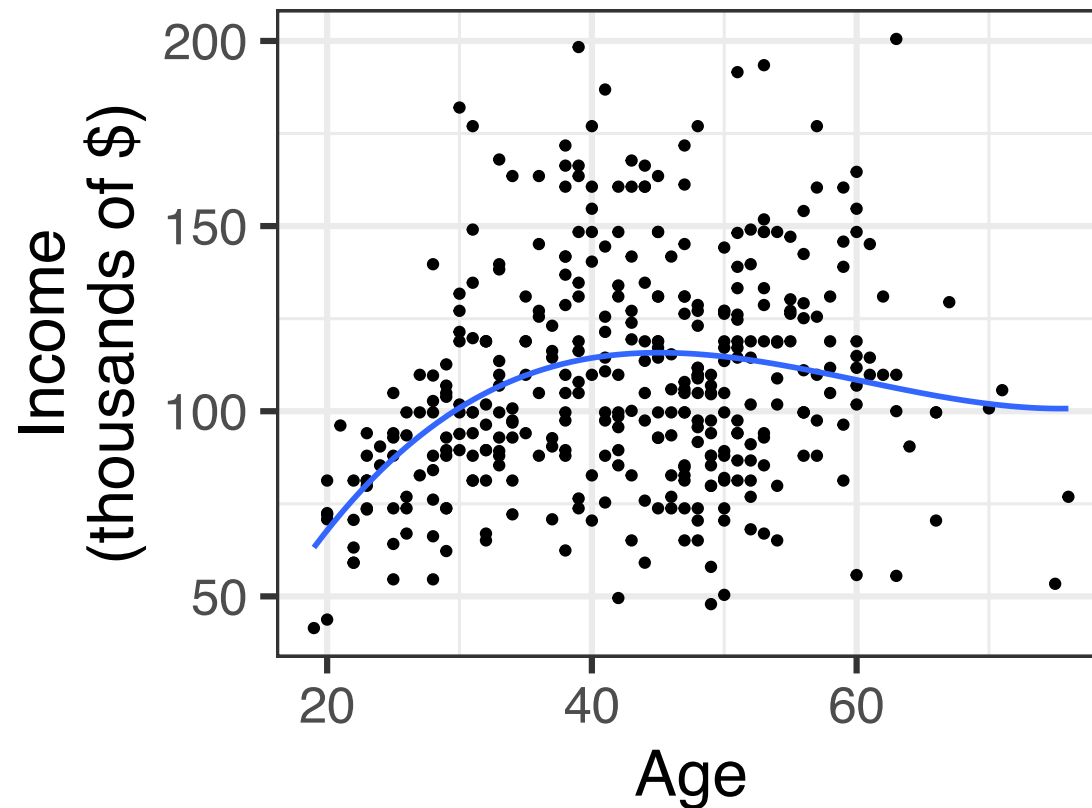
Polynomial model (quadratic trend)

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



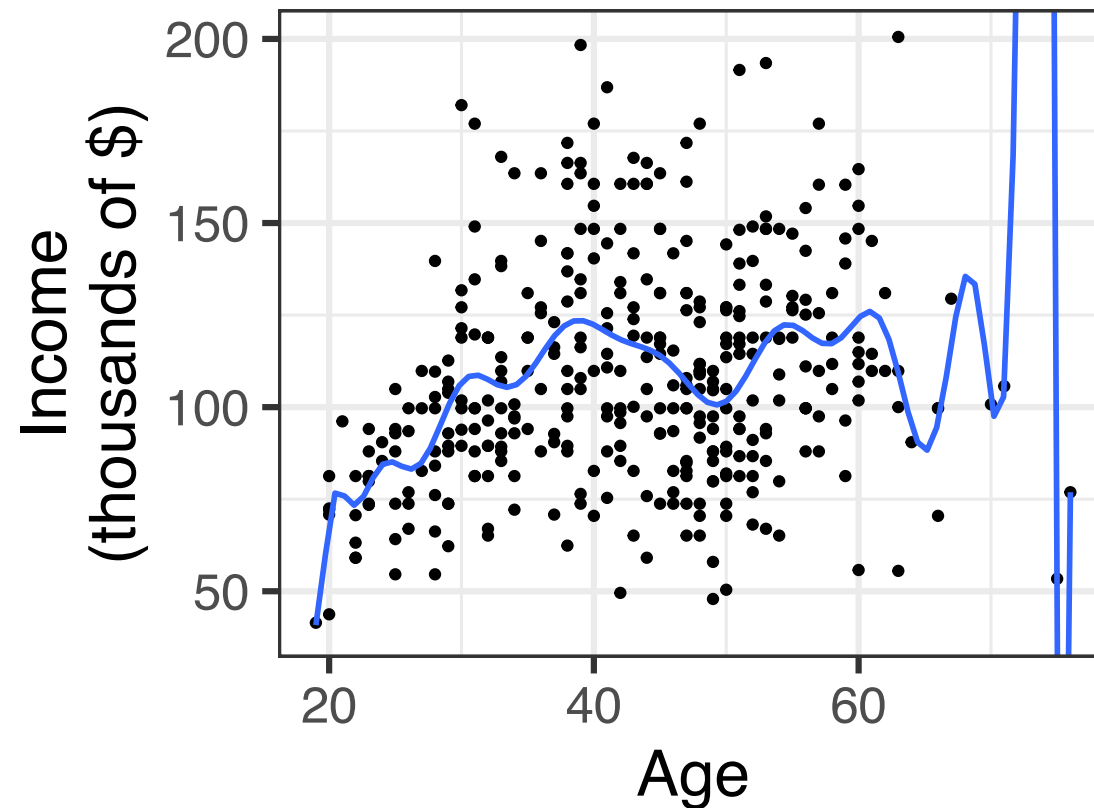
Polynomial model (cubic trend)

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



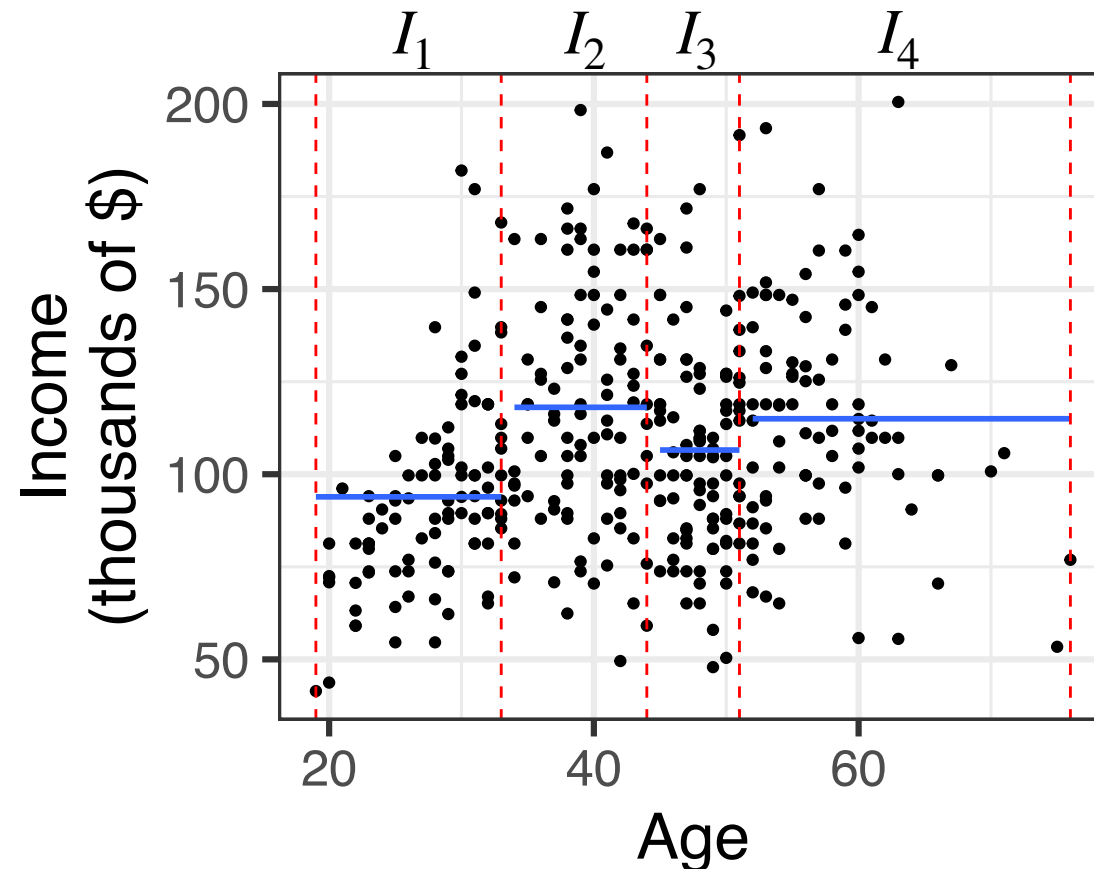
20th degree polynomial model

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



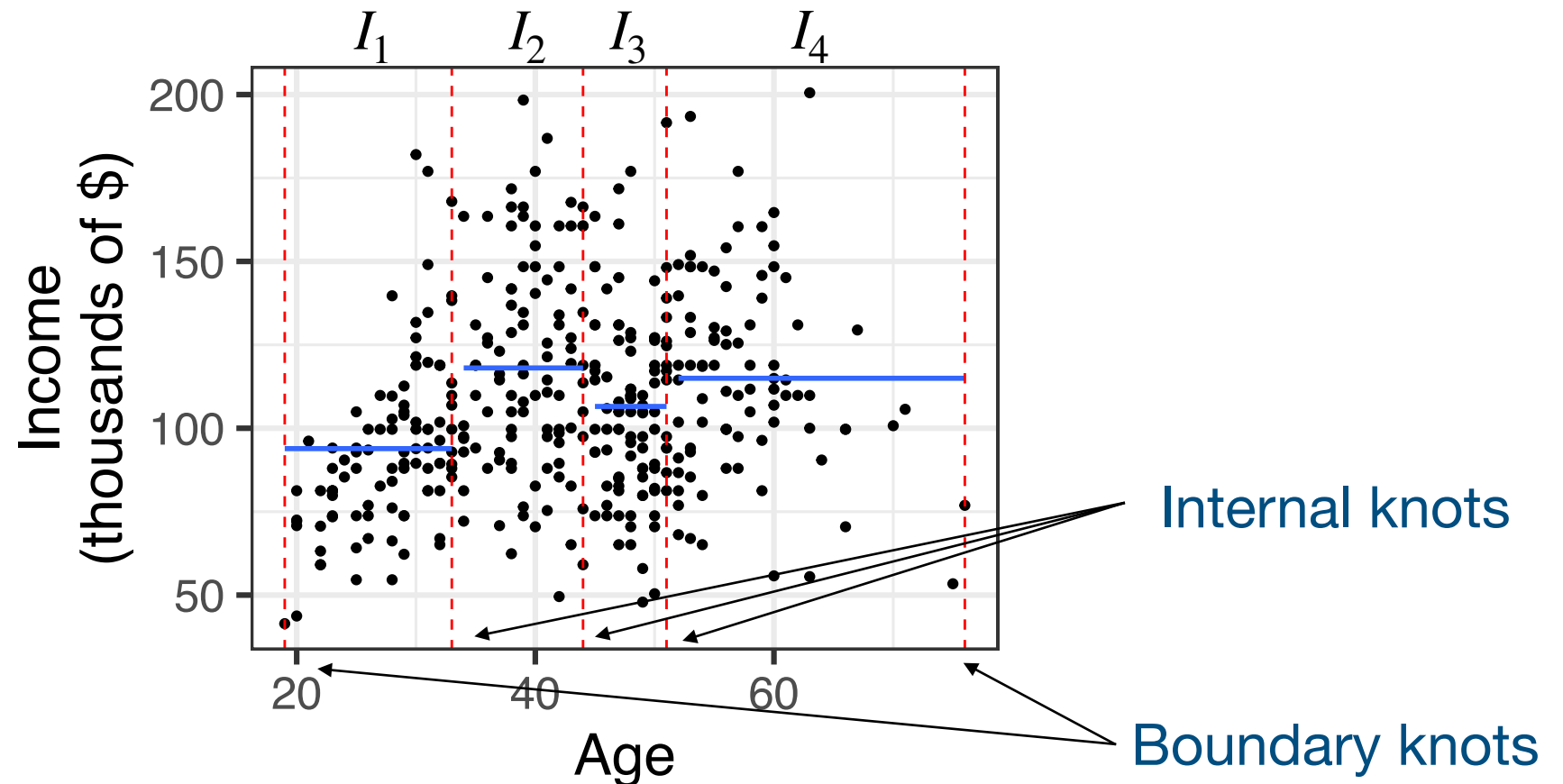
Piece-wise polynomial (piece-wise constant)

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



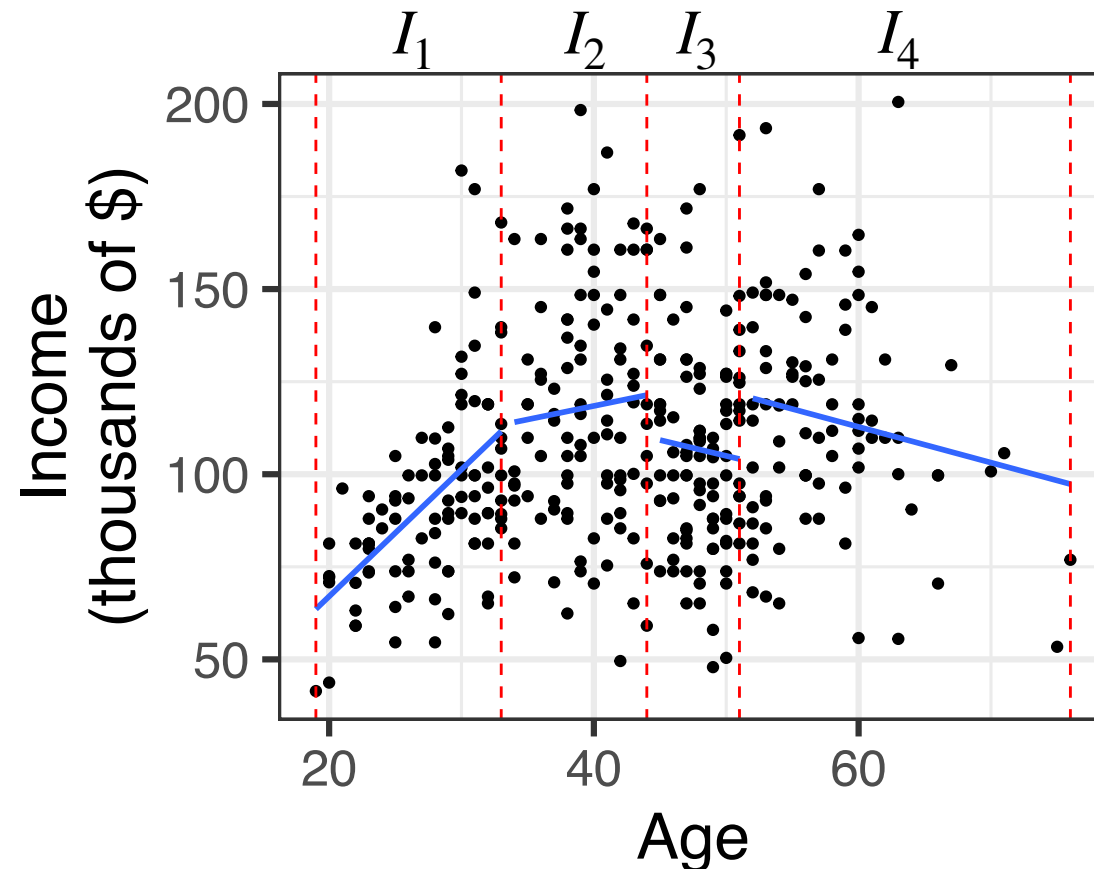
Piece-wise polynomial (piece-wise constant)

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



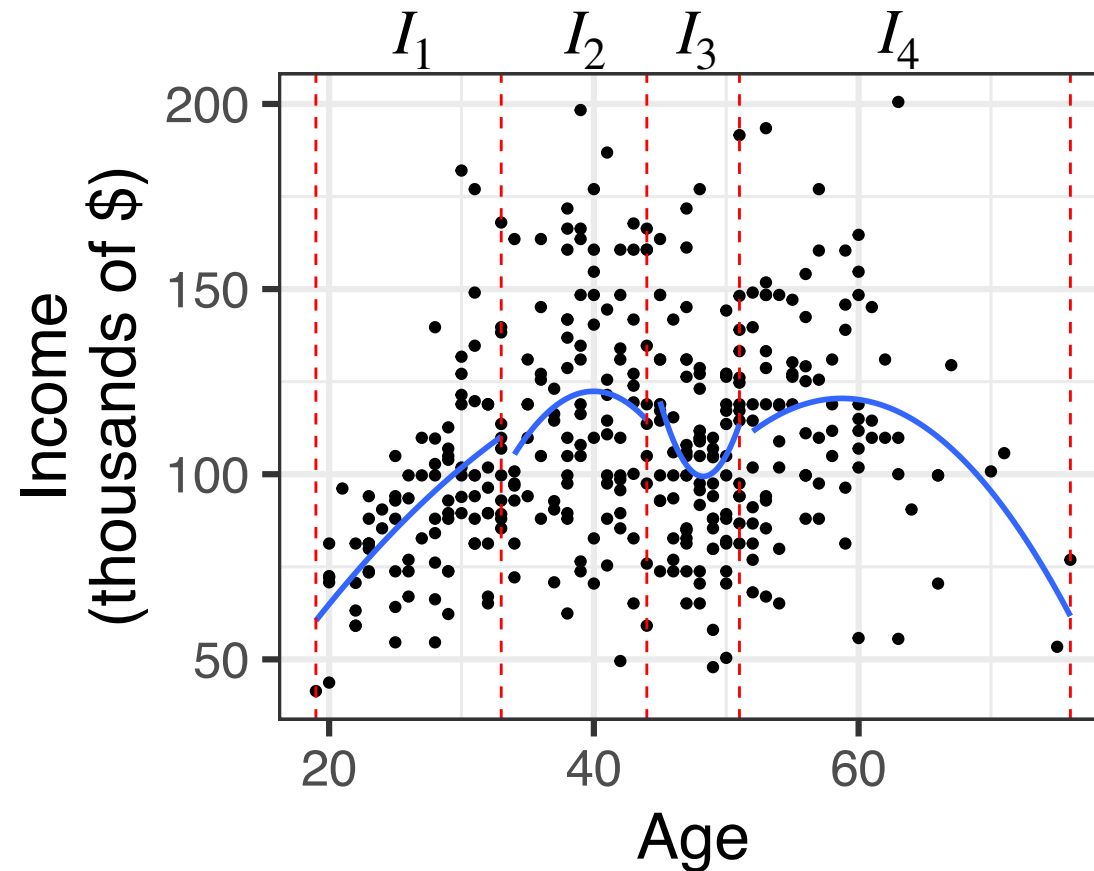
Piece-wise polynomial (piece-wise linear)

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



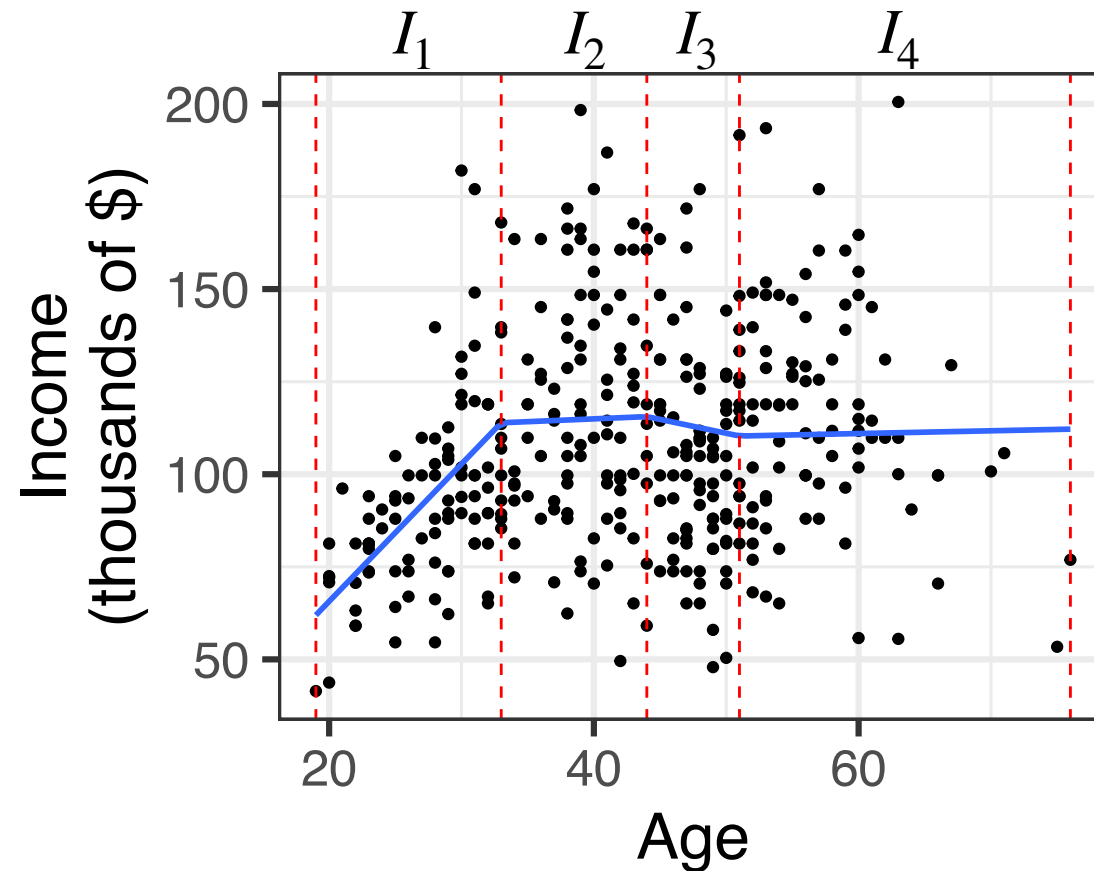
Piece-wise polynomial (piece-wise quadratic)

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



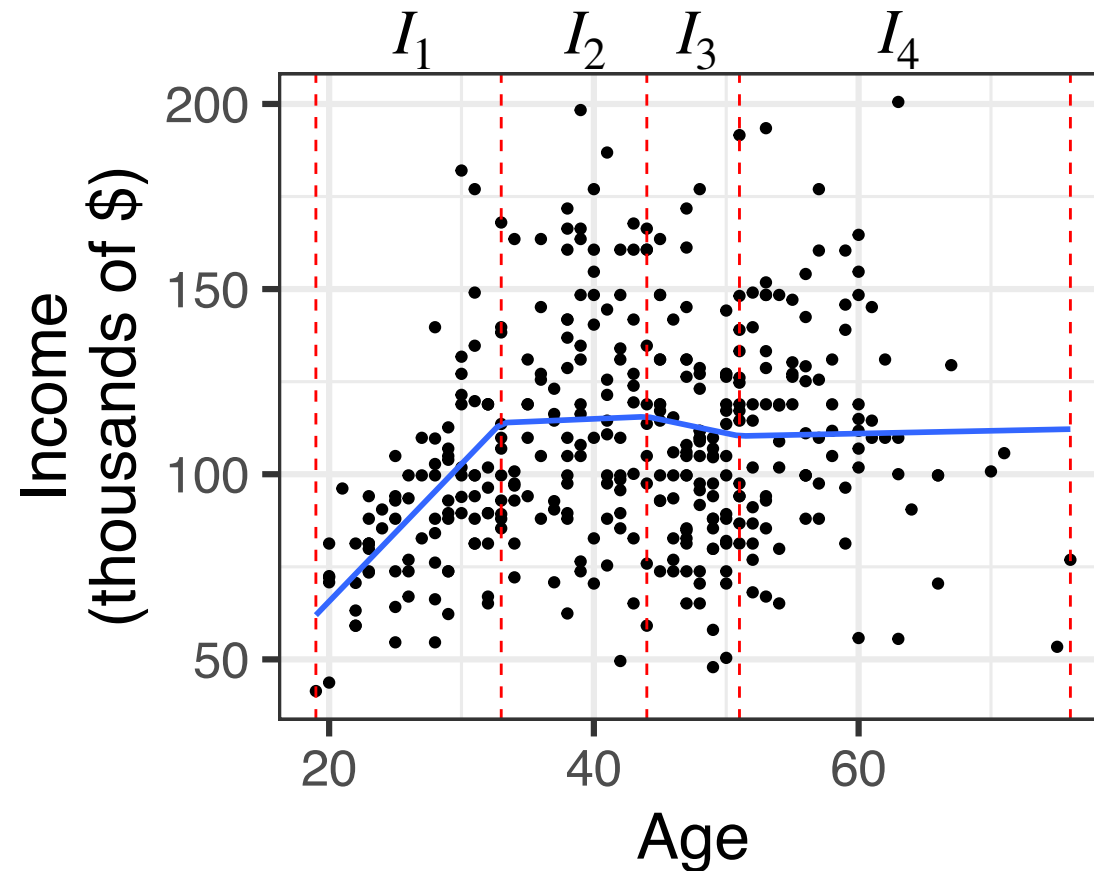
Spline (piece-wise linear)

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



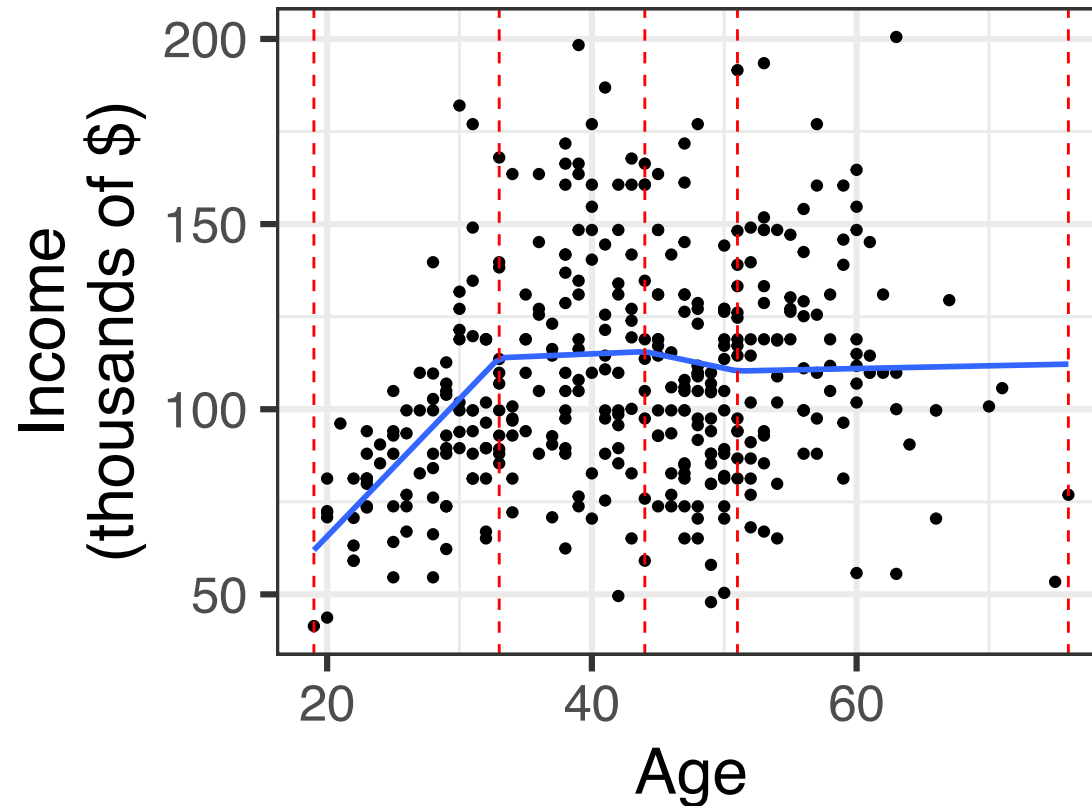
Spline (piece-wise linear)

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



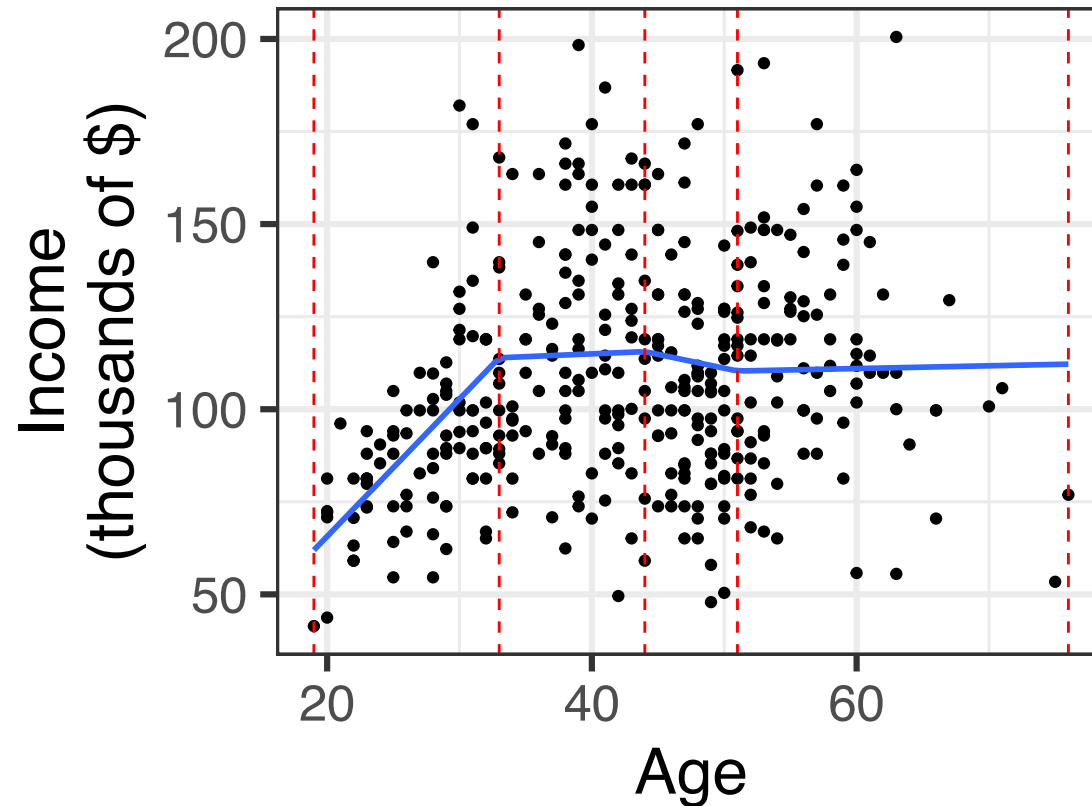
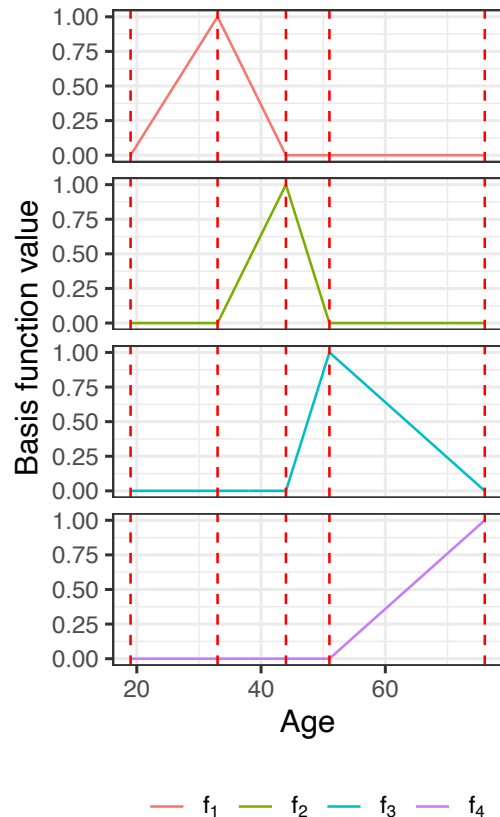
Spline (piece-wise linear)

$$\text{income} = \beta_0 + \beta_1 \cdot f_1(\text{age}) + \cdots + \beta_{p-1}f_{p-1}(\text{age}) + \epsilon$$



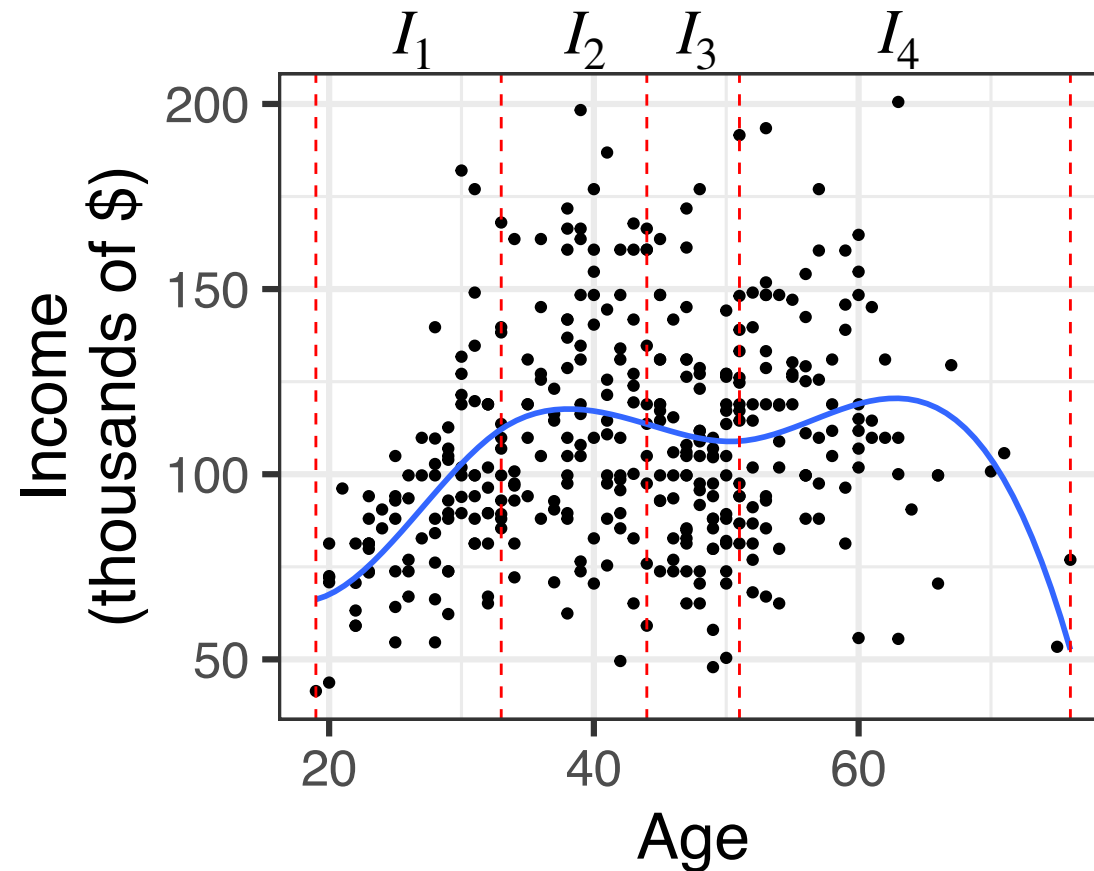
Spline (piece-wise linear)

$$\text{income} = \beta_0 + \beta_1 \cdot f_1(\text{age}) + \cdots + \beta_{p-1} f_{p-1}(\text{age}) + \epsilon$$



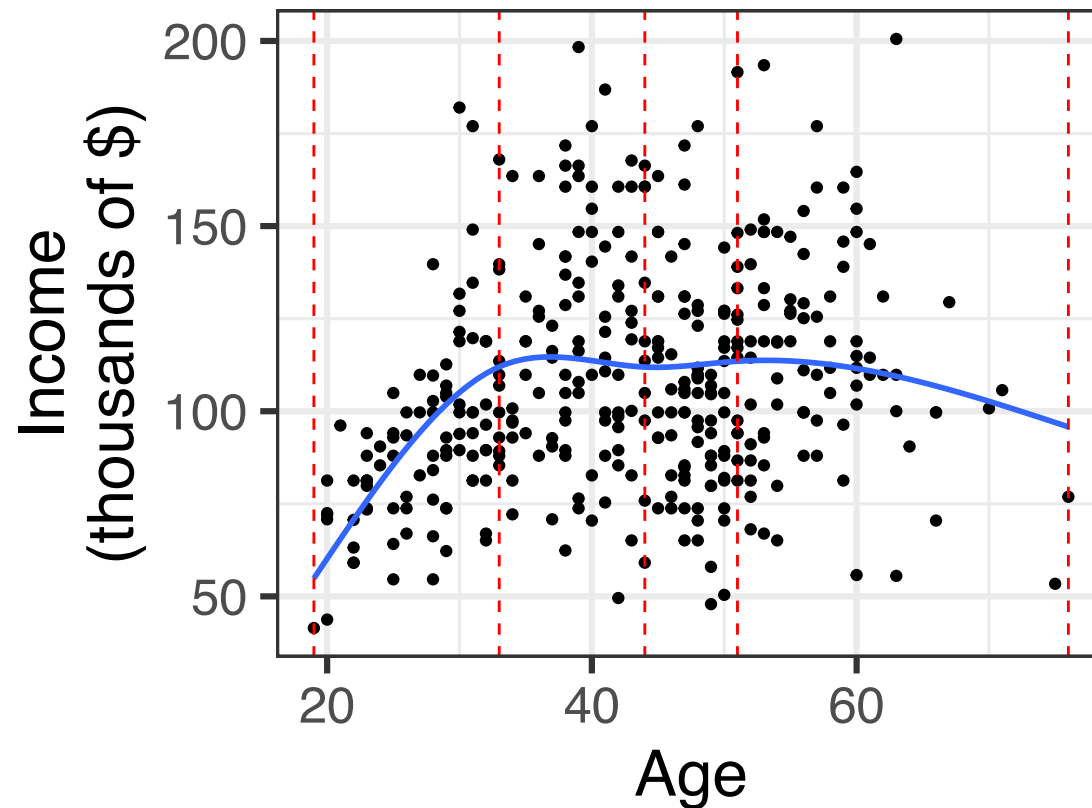
Spline (piece-wise cubic)

$$\text{income} = \beta_0 + \beta_1 \cdot f_1(\text{age}) + \cdots + \beta_{p-1}f_{p-1}(\text{age}) + \epsilon$$



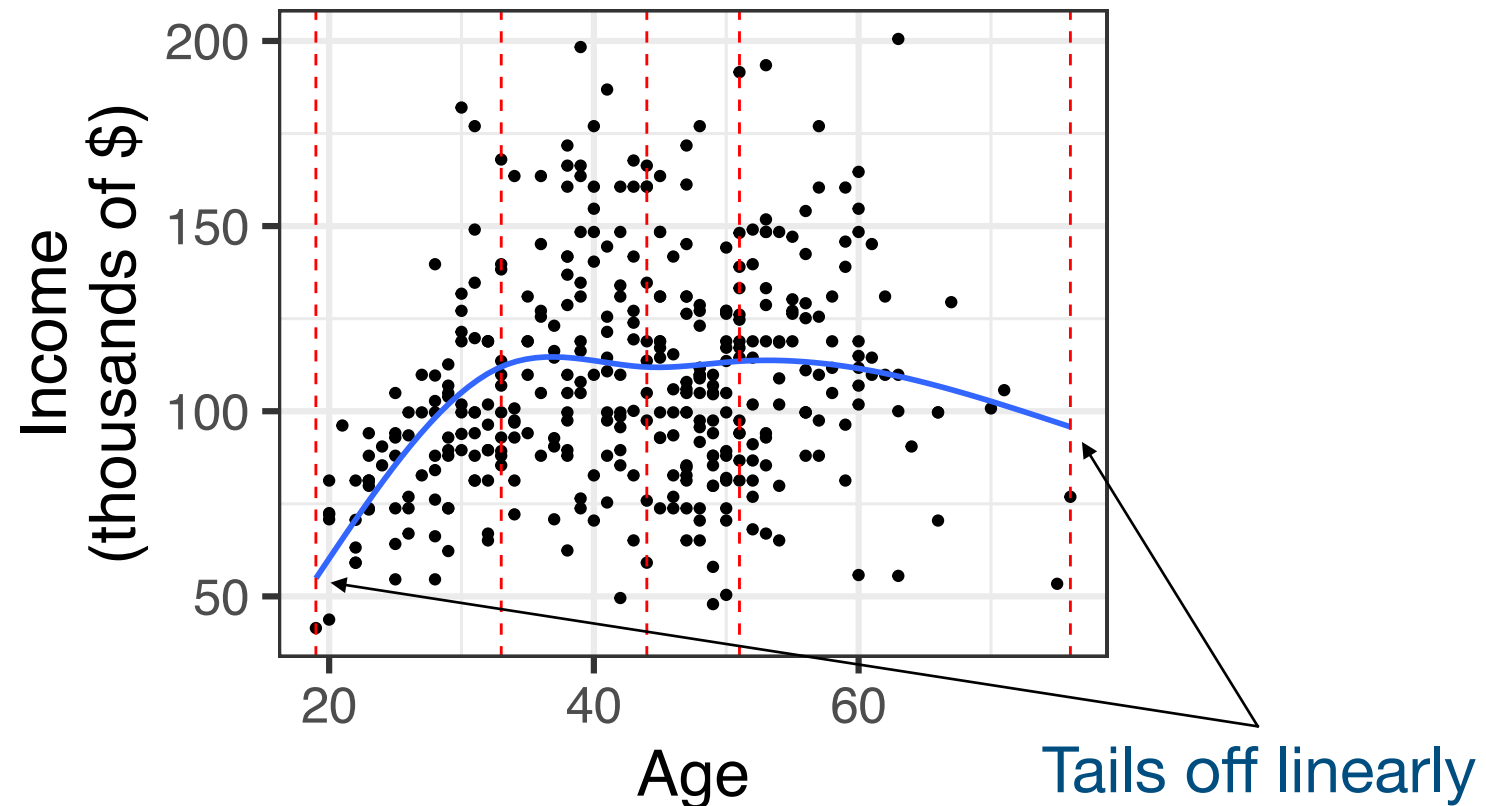
Natural cubic spline (with 5 total knots)

$$\text{income} = \beta_0 + \beta_1 \cdot f_1(\text{age}) + \cdots + \beta_{p-1}f_{p-1}(\text{age}) + \epsilon$$



Natural cubic spline (with 5 total knots)

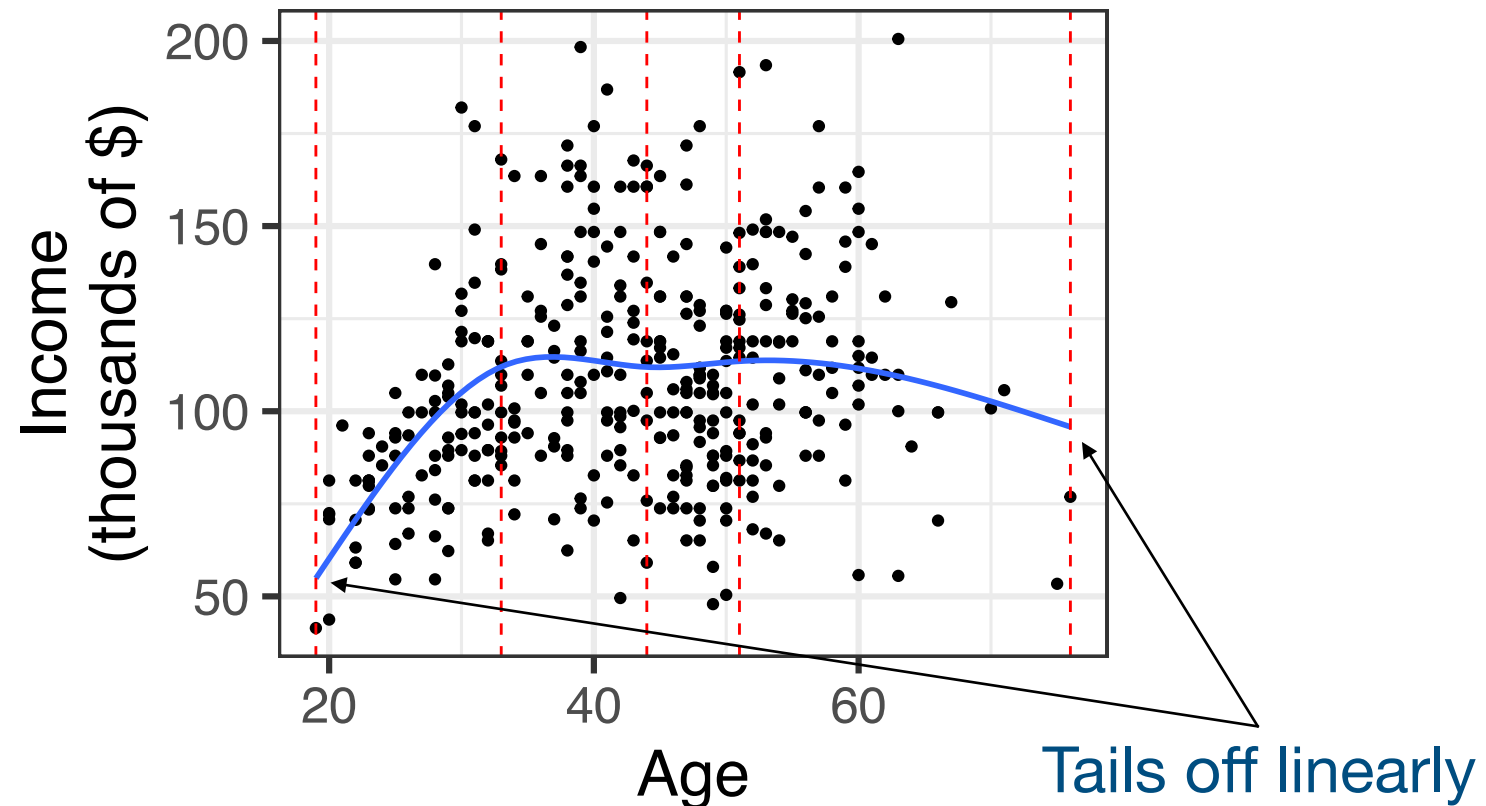
$$\text{income} = \beta_0 + \beta_1 \cdot f_1(\text{age}) + \cdots + \beta_{p-1} f_{p-1}(\text{age}) + \epsilon$$



Natural cubic spline (with 5 total knots)

$$\text{income} = \beta_0 + \beta_1 \cdot f_1(\text{age}) + \cdots + \beta_{p-1} f_{p-1}(\text{age}) + \epsilon$$

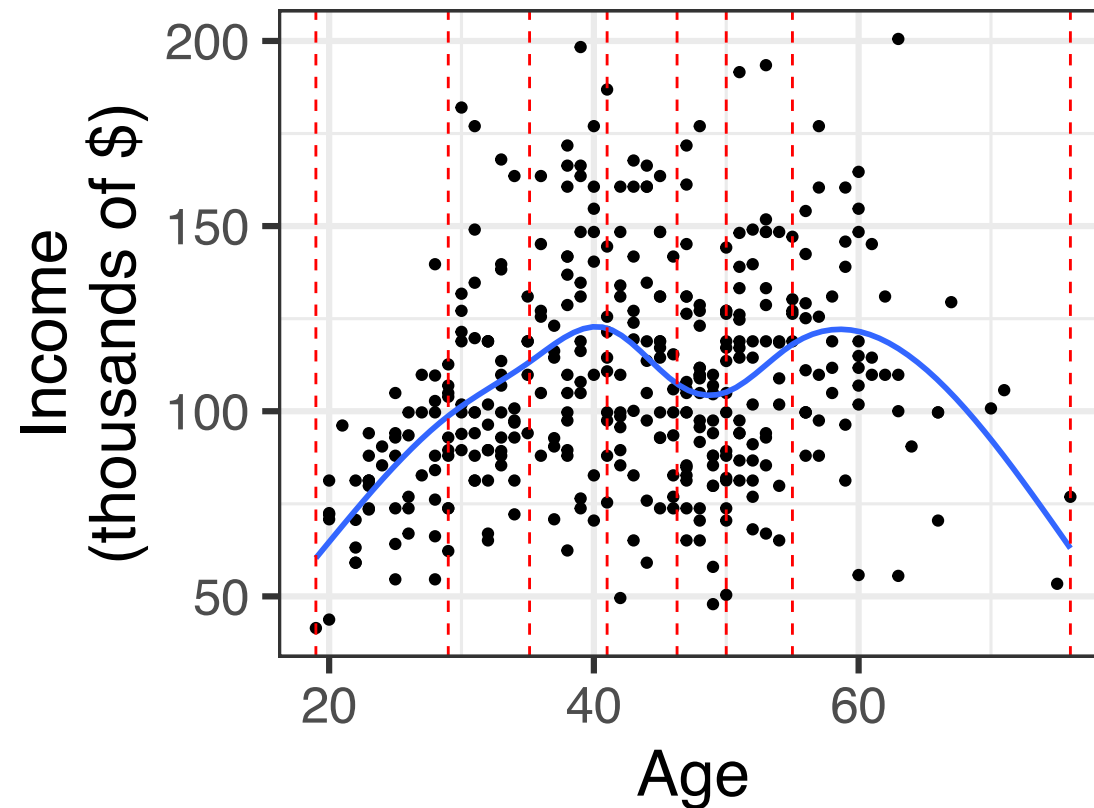
The preferred way to fit smooth curves to data.



Natural cubic spline (with 8 total knots)

$$\text{income} = \beta_0 + \beta_1 \cdot f_1(\text{age}) + \cdots + \beta_{p-1} f_{p-1}(\text{age}) + \epsilon$$

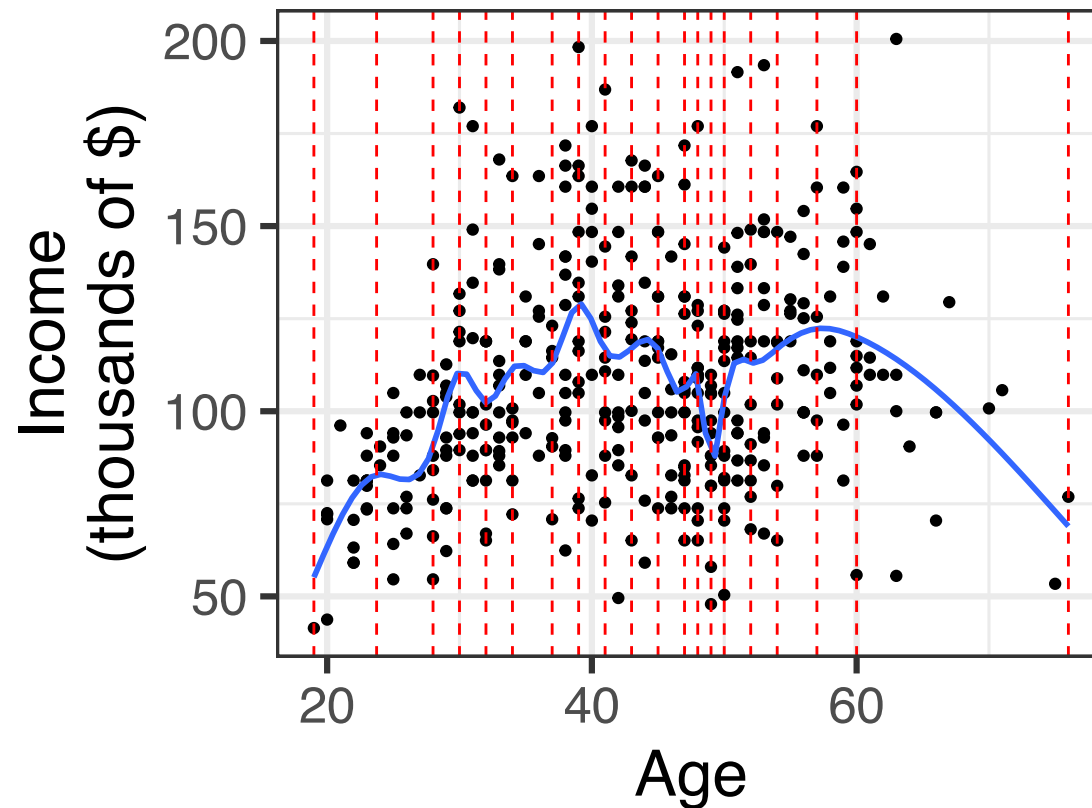
The preferred way to fit smooth curves to data.



Natural cubic spline (with 20 total knots)

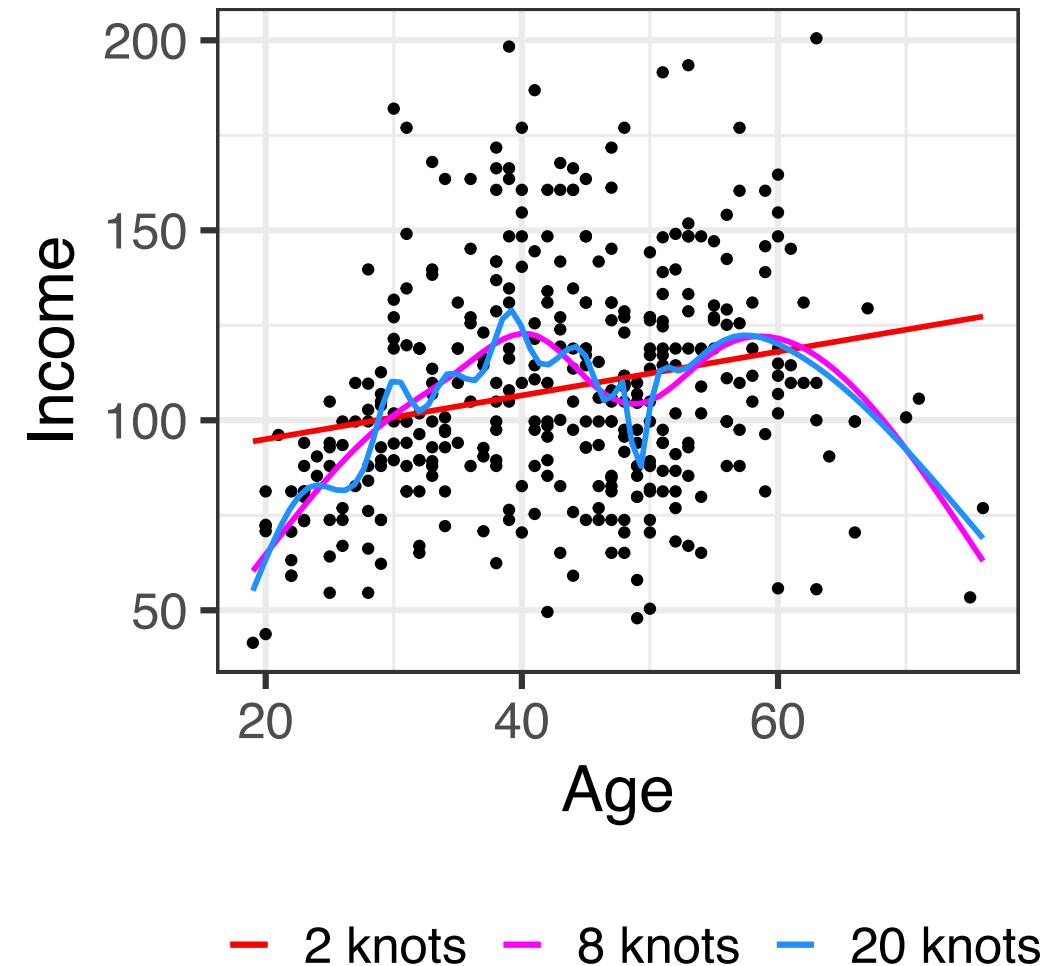
$$\text{income} = \beta_0 + \beta_1 \cdot f_1(\text{age}) + \cdots + \beta_{p-1} f_{p-1}(\text{age}) + \epsilon$$

The preferred way to fit smooth curves to data.



Model complexity

- The same data can be fit with models of varying degrees of *flexibility* or *complexity*.
- Example: Natural cubic splines with more knots are more flexible.
- 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



Quantifying model complexity for linear regression

Complexity for linear regression models of the form

$$y = \beta_0 + \beta_1 \cdot f_1(x) + \cdots + \beta_{p-1} f_{p-1}(x) + \epsilon$$

is quantified via *degrees of freedom (df)*, the number of free parameters β_j . In particular, the model above has p degrees of freedom.

Important examples:

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

Quantifying model complexity for linear regression

Complexity for linear regression models of the form

$$y = \beta_0 + \beta_1 \cdot f_1(x) + \cdots + \beta_{p-1} f_{p-1}(x) + \epsilon$$

is quantified via *degrees of freedom (df)*, the number of free parameters β_j . In particular, the model above has p degrees of freedom.

Important examples:

- Polynomials of degree p have p 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 $\text{df} = K - 1$.

Recall: Prediction performance

You have **training data** $(X_1^{\text{train}}, Y_1^{\text{train}}), \dots, (X_n^{\text{train}}, Y_n^{\text{train}})$, based on which you construct a predictive model \hat{f} such that, hopefully, $Y \approx \hat{f}(X)$.

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: extent to which $Y_i^{\text{test}} \approx \hat{Y}_i^{\text{test}}$, e.g. **mean squared test error**:

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

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**).

