All Training materials are provided "as is" and without warranty and RStudio disclaims any and all express and implied warranties including without limitation the implied warranties of title, fitness for a particular purpose, merchantability and noninfringement.

Studio

Advanced Modeling

Many modeling methods for many types of data

http://www.flickr.com/photos/thedailyenglishshow/5585763492/

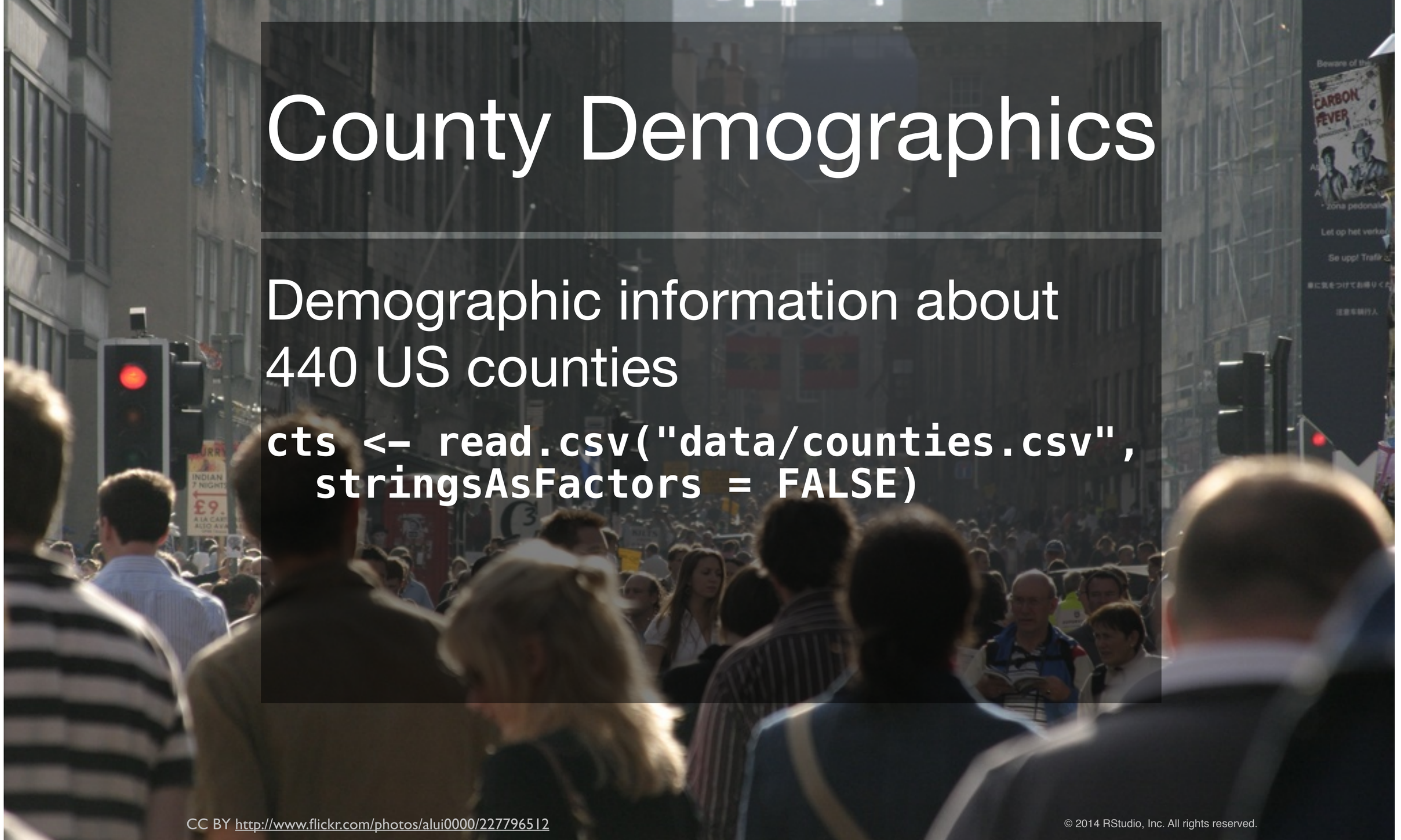
Garrett Grolemund

Master Instructor, RStudio

August 2014

1. Variable Selection
2. Non-linear models
   a. Transformations
   b. Splines
   c. Smoothing Functions
3. Logistic Regression

Variable selection



County Demographics

Demographic information about 440 US counties

```
cts <- read.csv("data/counties.csv",
  stringsAsFactors = FALSE)
```

Variable	Measures
county	Name of county
state	Name of state
crime	**Number of crimes per 1000 people**
area	Land area of county
pop	Population of county
p18_25	% between 18 and 25 years old
p65	% greater that 65 years old
nphysician	number of physicians
nhospbeds	number of hospital beds
phs	% with high school degree
pcollege	% with college degree
ppoverty	% below poverty line
punemployed	% unemployed
avg_income	Per capita income
tot_income	Total income
region	Region of countr

Which variables predict crime?

Variable selection

1. Theory driven
2. Data driven (stepwise)
3. Penalized regression (LASSO)

Theory driven

Theory driven

Let established scientific theory determine which variables should be in your model *before you look at the data.*

Data driven

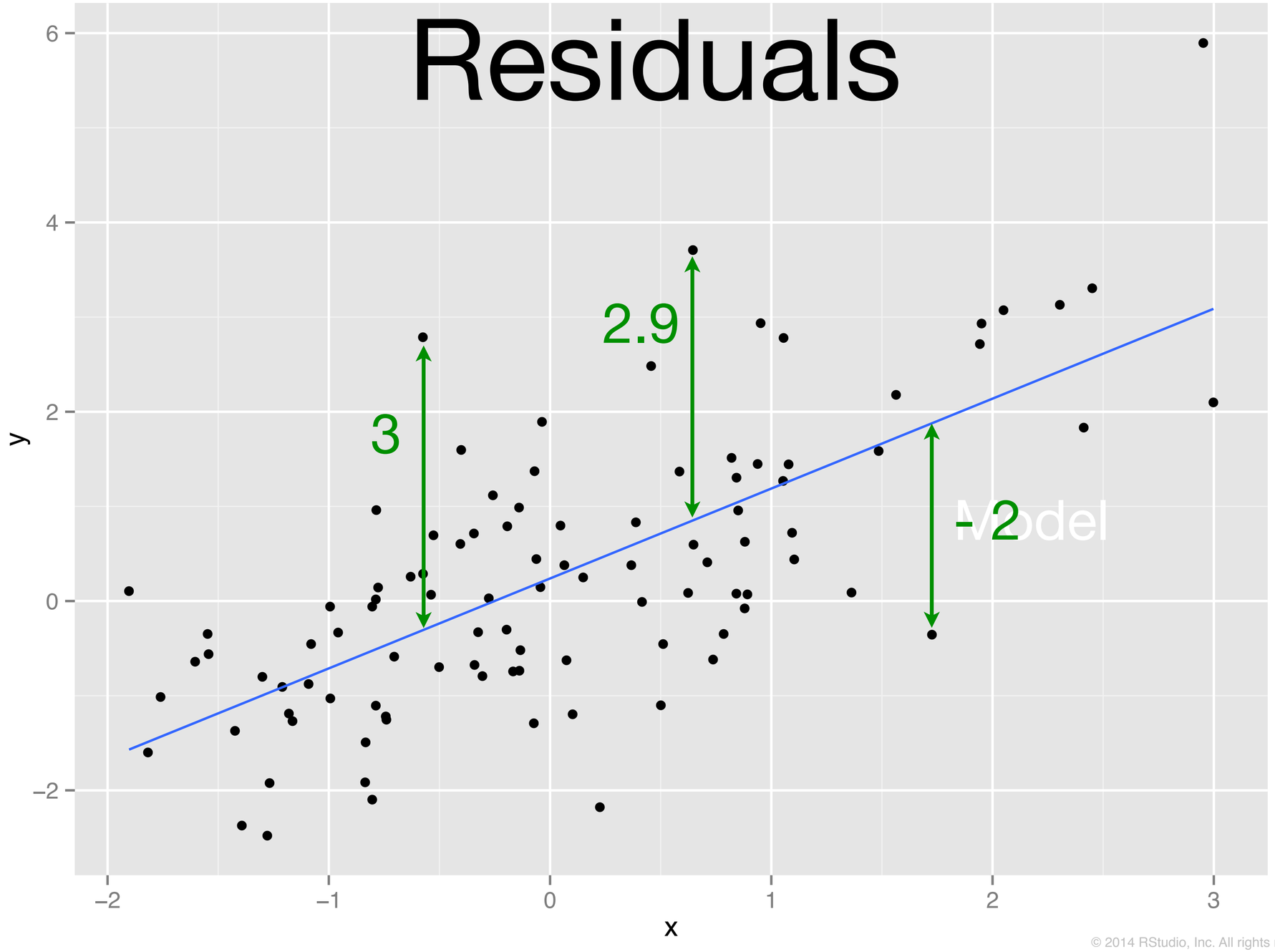
Warm up

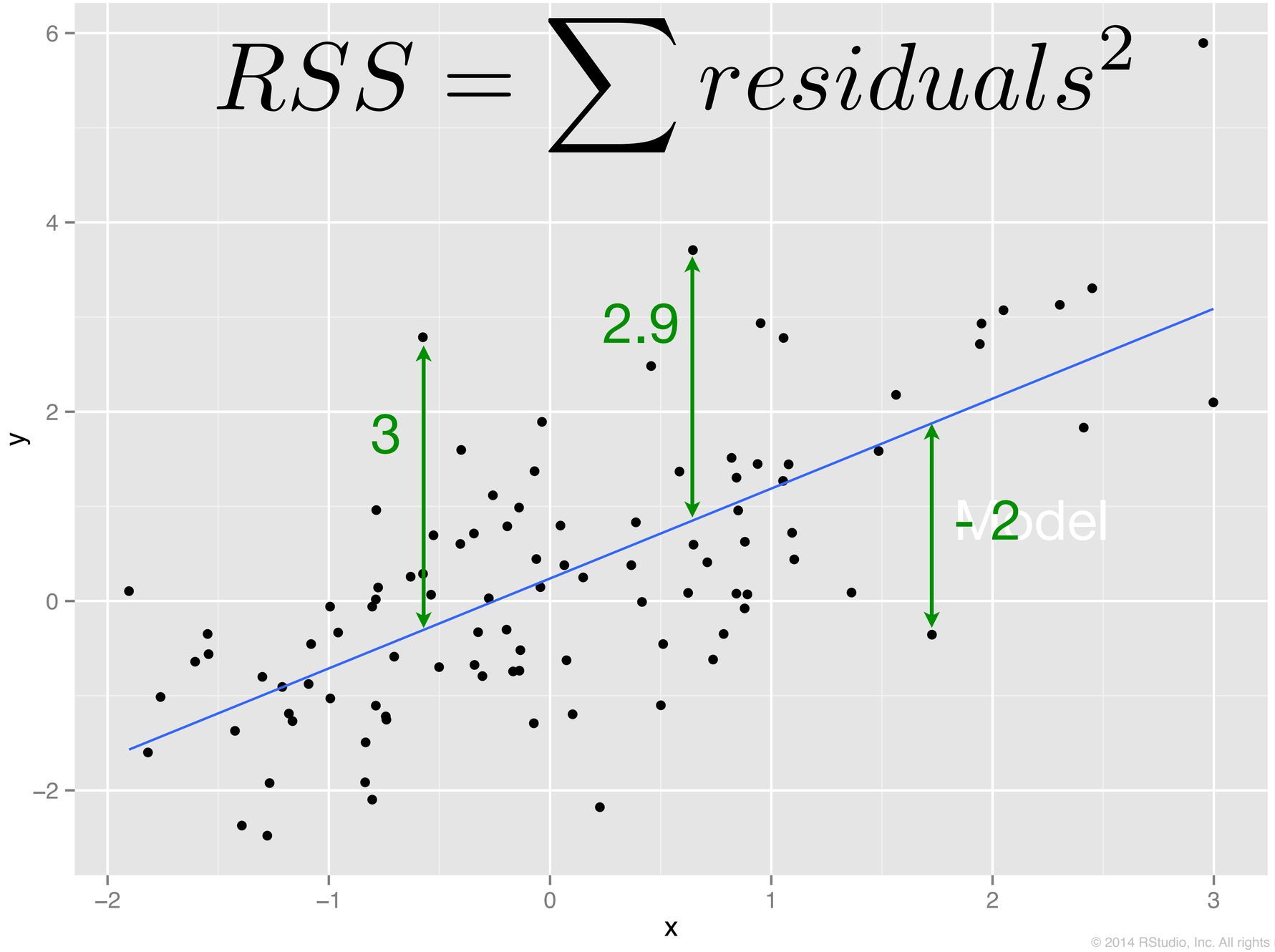
How can you identify a good model?

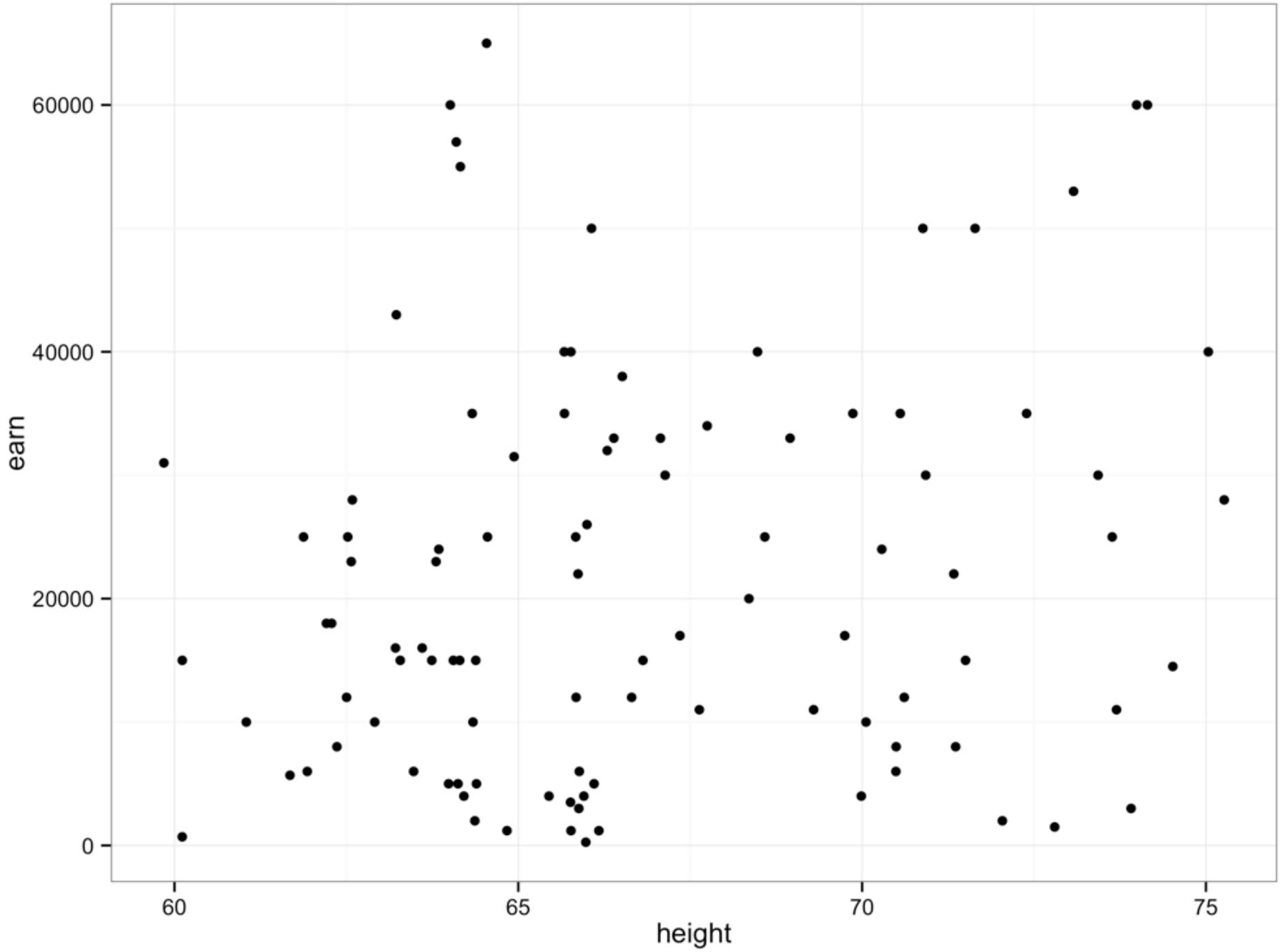
1. Small residuals*

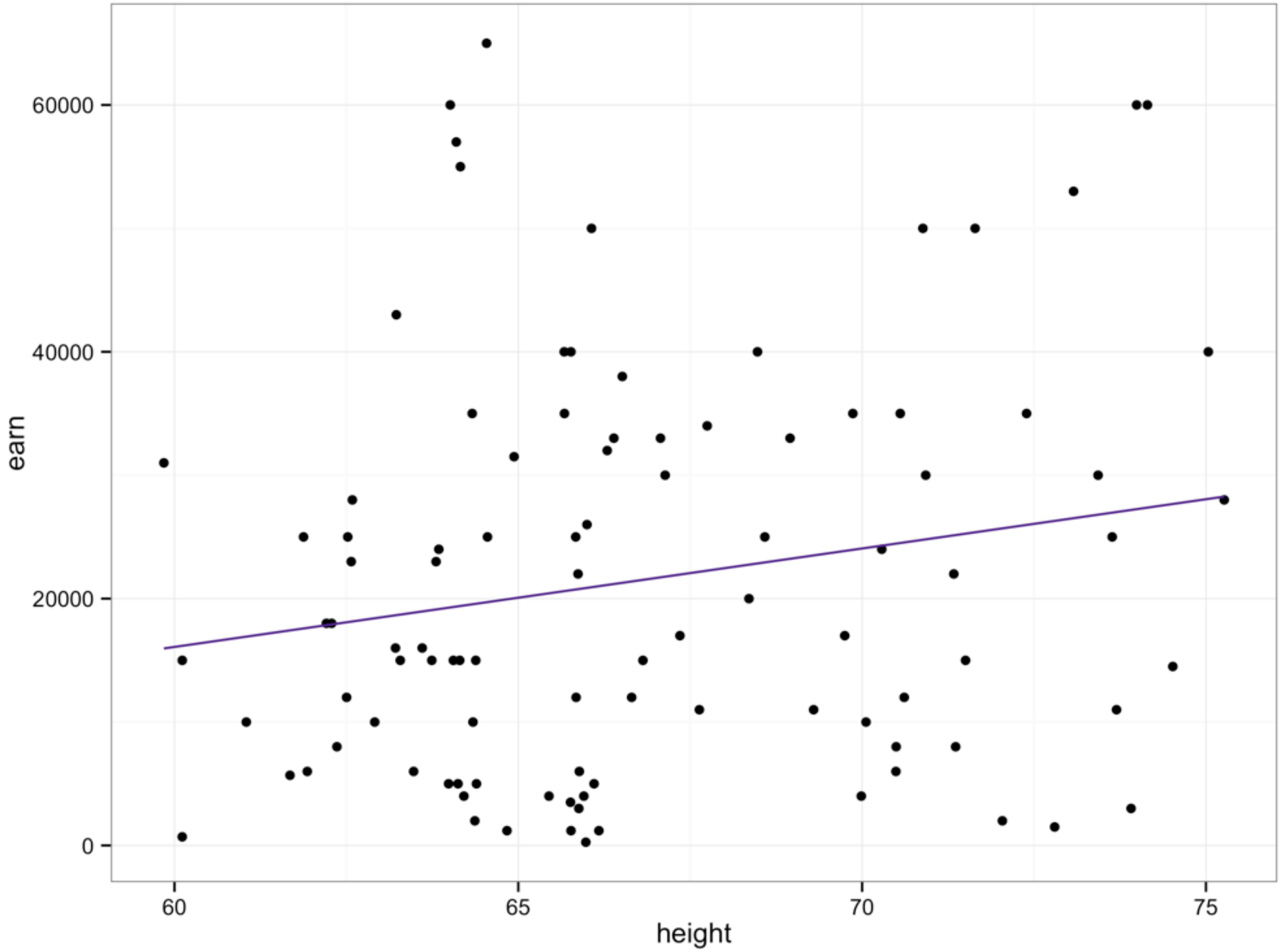
2. Accurate predictions

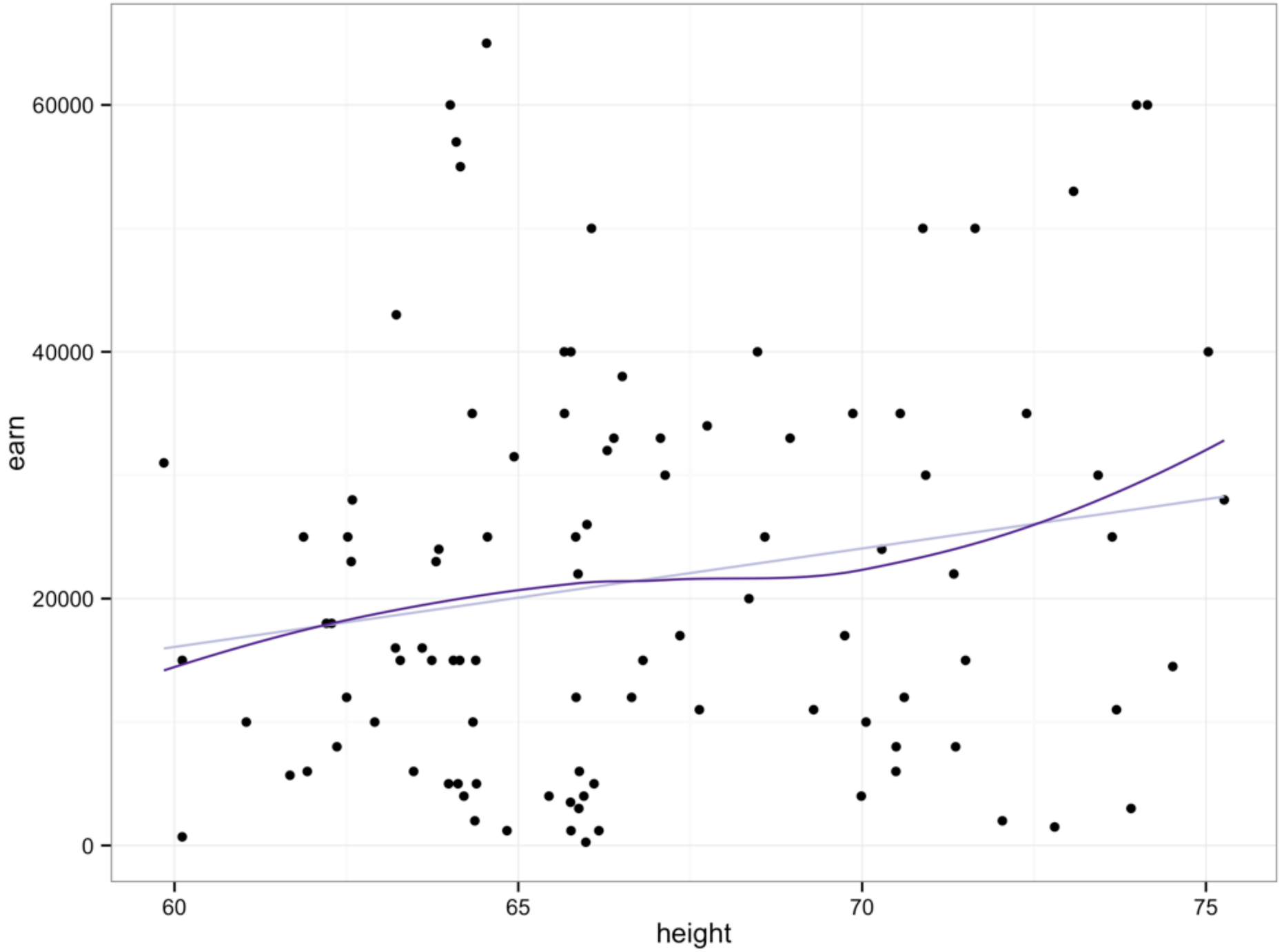
3. ...

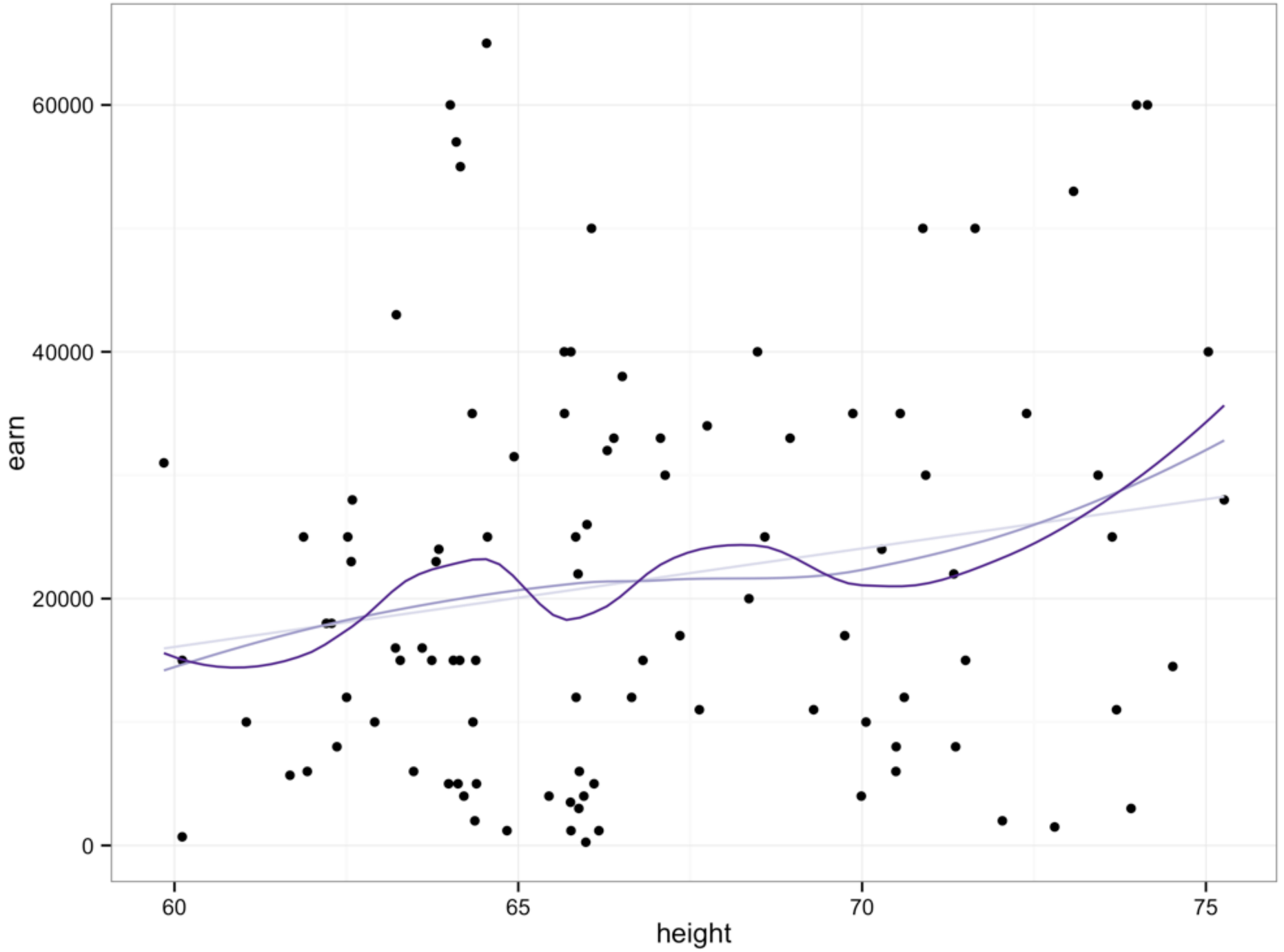


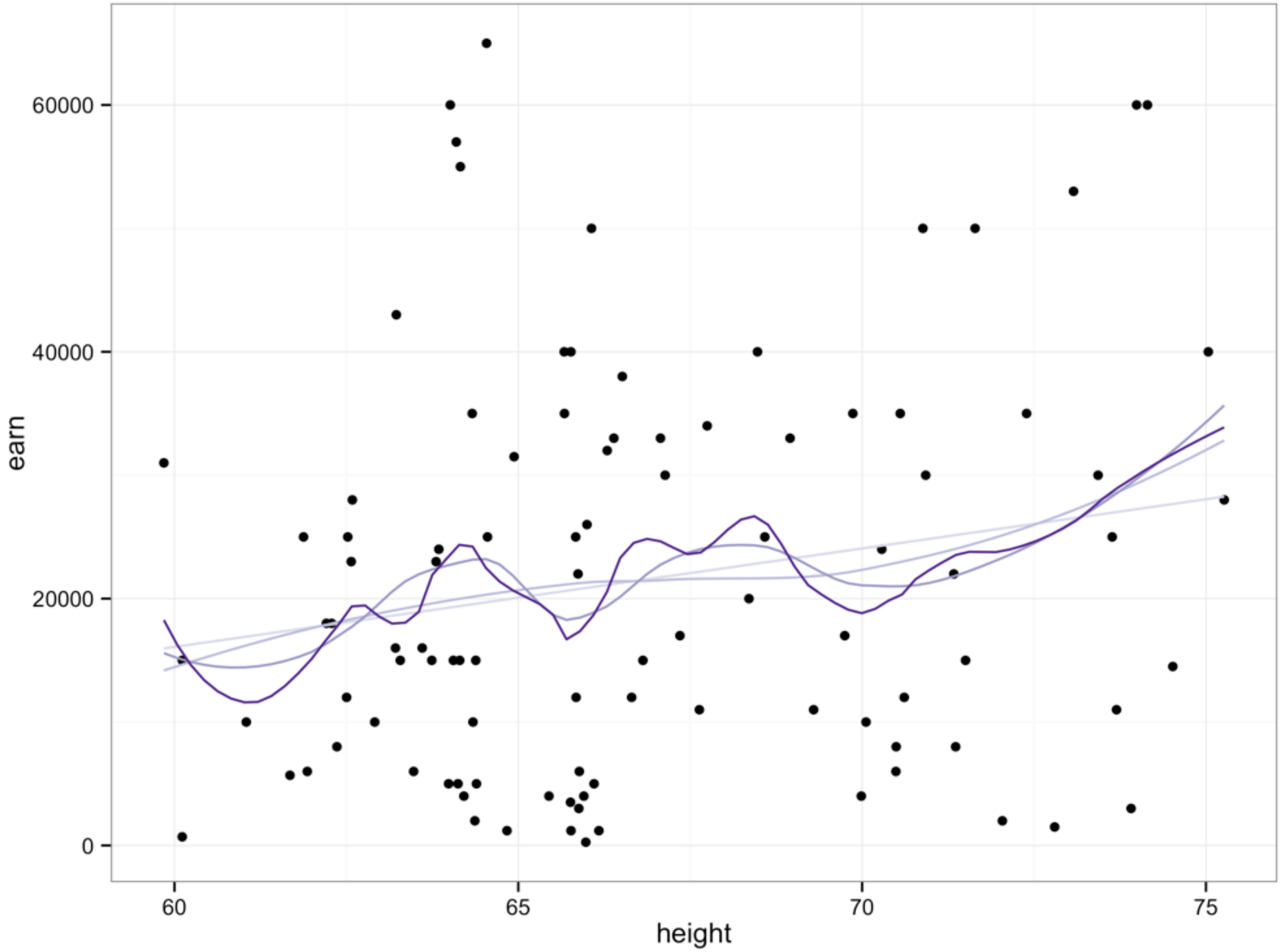


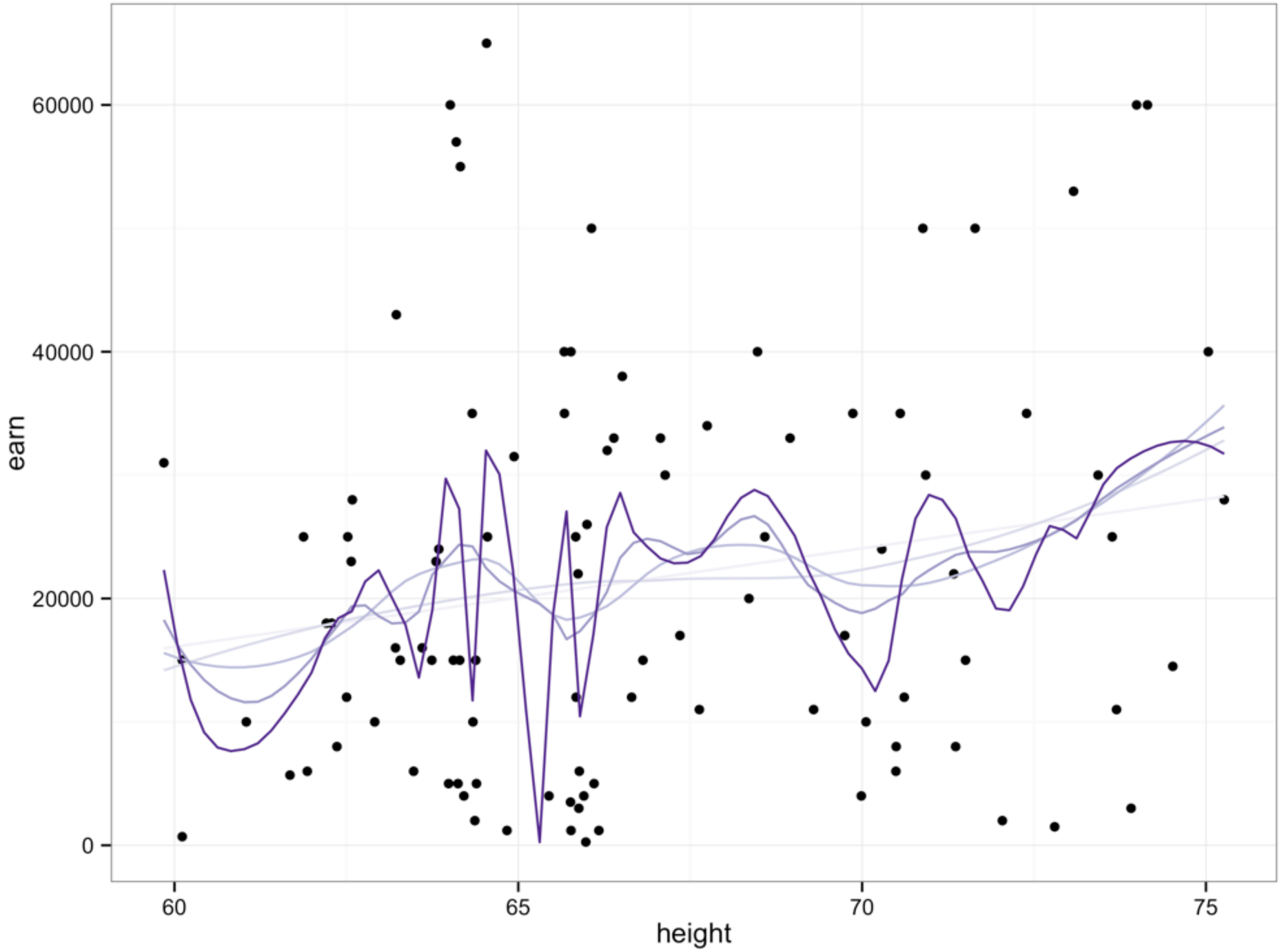

60000
40000
20000
0
earn
60
65
70
75
height

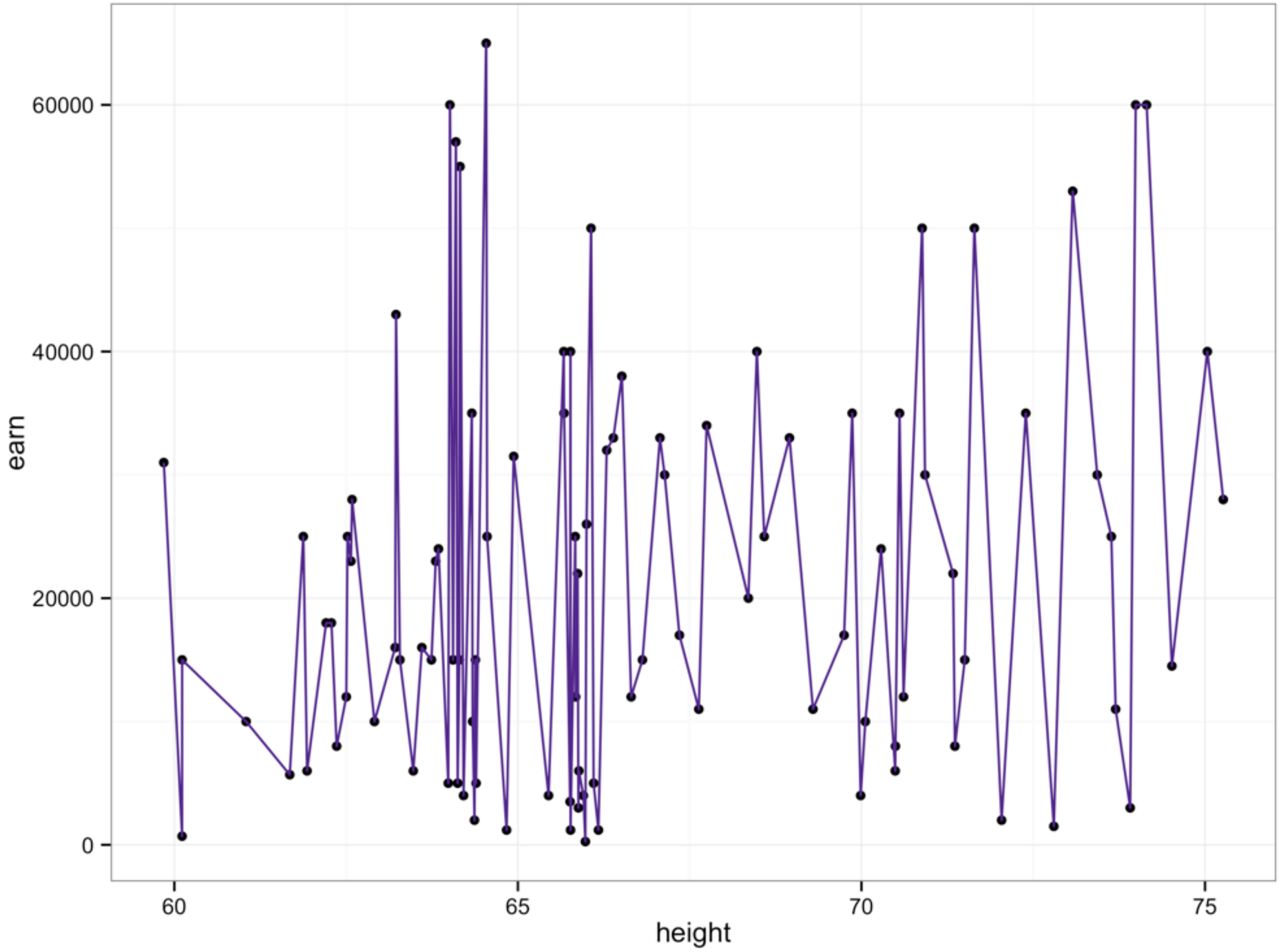

60000
40000
20000
0
earn
60
65
70
75
height


60000
40000
20000
0
earn
60
65
70
75
height

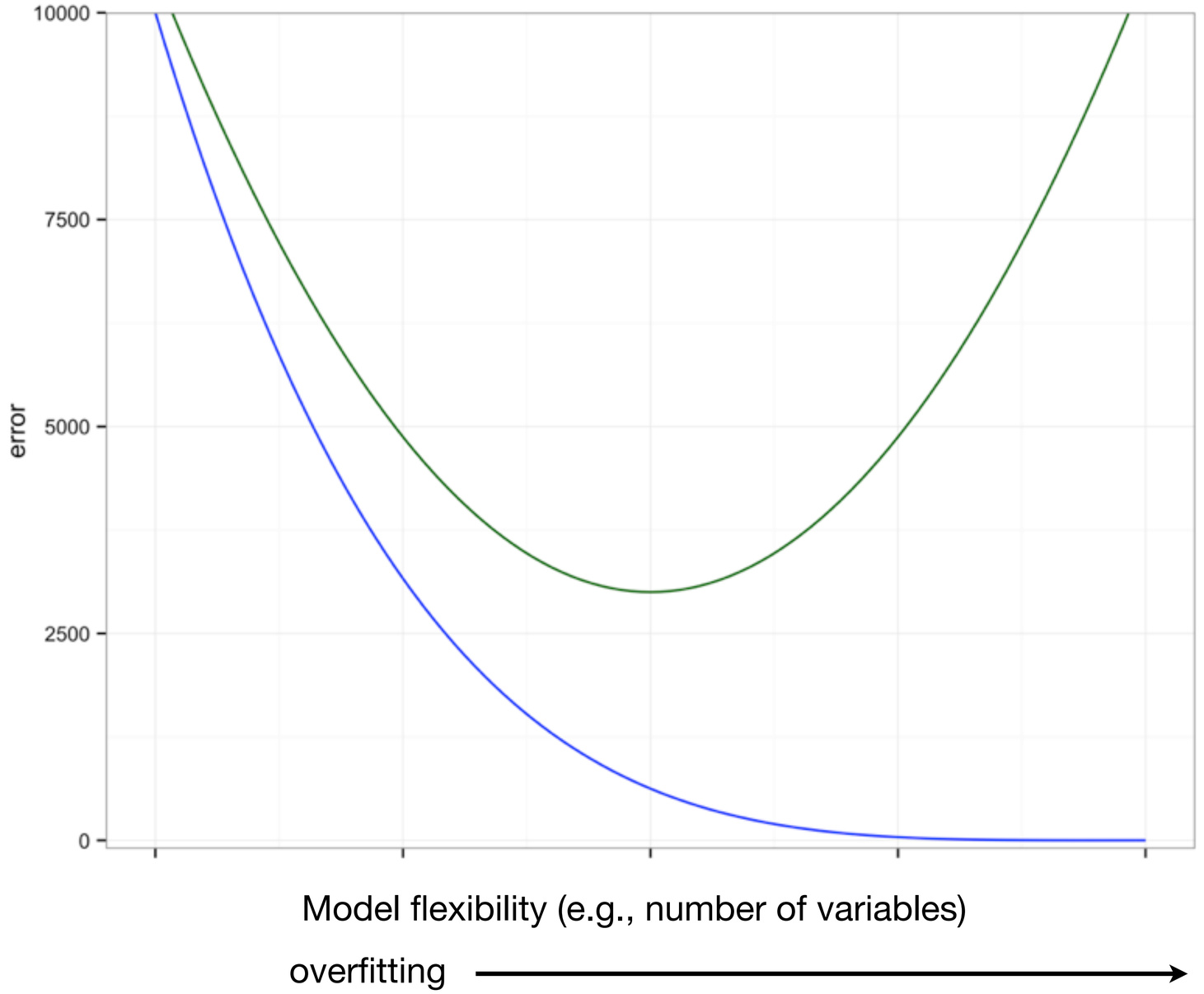

60000
40000
20000
0
earn
60
65
70
75
height


60000
40000
20000
0
earn
60
65
70
75
height


60000
40000
20000
0
earn
60
65
70
75
height


60000
40000
20000
0
earn
60
65
70
75
height

Predictive accuracy



Test error
(predictions
vs. new data)

Training error
(model vs.
available data)

Principle

A good model has low test error. This implies that it makes good predictions and has reasonably small residuals.

There are a number of ways to estimate test error without collecting new data.

Mallow's C_p

$$C_p = \frac{1}{n}(RSS + 2p\hat{\sigma}^2)$$

C_p adds a penalty as number of variables (flexibility) increases.

Choose the lowest C_p.

AIC

$$AIC = \frac{2}{n}(p - loglik)$$

Similar to C_p, but can be generalized to non-linear models.

Choose the lowest AIC.

```
# heights <- read.csv("data/heights.csv")
# hmod <- lm(earn ~ height, data = wages)
AIC(hmod)
# 32341.19
```

BIC

$$BIC = -2 \cdot loglik + p \cdot log(n)$$

Proportional to AIC, but penalizes flexibility more heavily.

Choose the lowest BIC.

```
BIC(hmod)
# 32356.88
```

Adjusted R^2

$$Adj\,R^2 = 1 - \frac{RSS/(n-p-1)}{TSS/(n-1)}$$

Adds penalty to R^2 for unnecessary variables.
Popular, but not as statistically well-founded as C_p, AIC and BIC.

Choose the *highest* Adjusted R^2.

```
summary(hmod)$adj.r.squared
# 0.08436624
```

Statistic	best when	useful cases
adjusted R	high ↑	
Mallow's C	low ↓	linear, least squares regression
AIC	low ↓	true relationship is complex
BIC	low ↓	true relationship is simple

Statistic	best when	useful cases
adjusted R	high ↑	
Mallow's C	low ↓	linear, least squares regression
AIC	low ↓	true relationship is complex
BIC	low ↓	true relationship is simple

Your turn

Determine each possible combination of variables in `wages` that you could use to model `earn` (ignore interaction effects).

Devise a strategy for testing how well one combination does.

```
earn ~ height
earn ~ sex
earn ~ ed
earn ~ age
earn ~ race
```

```
earn ~ height + sex
earn ~ height + ed
earn ~ height + age
earn ~ height + race
earn ~ sex + ed
earn ~ sex + age
earn ~ sex + race
earn ~ ed + age
earn ~ ed + race
earn ~ age + race
```

```
earn ~ height + sex + ed
earn ~ height + sex + age
earn ~ height + sex + race
earn ~ height + ed + age
earn ~ height + ed + race
earn ~ height + age + race
earn ~ sex + ed + age
earn ~ sex + ed + race
earn ~ sex + age + race
earn ~ ed + age + race
```

```
earn ~ height + sex + ed + age
earn ~ height + sex + ed + race
earn ~ height + sex + age + race
earn ~ height + ed + age + race
earn ~ sex + ed + age + race
```

```
earn ~ height + sex + ed + age + race
```

Data driven

Use whichever combination of variables provides the best prediction error. Also known as:

- best subset selection
- stepwise selection

Best Subset selection

Find the prediction error for every possible subset of variables. Use the subset with the lowest test error.

But that's **A LOT** of cross-validation

```
earn ~ height
earn ~ sex
earn ~ ed
earn ~ age
earn ~ race
```

```
earn ~ height + sex
earn ~ height + ed
earn ~ height + age
earn ~ height + race
earn ~ sex + ed
earn ~ sex + age
earn ~ sex + race
earn ~ ed + age
earn ~ ed + race
earn ~ age + race
```

```
earn ~ height + sex + ed
earn ~ height + sex + age
earn ~ height + sex + race
earn ~ height + ed + age
earn ~ height + ed + race
earn ~ height + age + race
earn ~ sex + ed + age
earn ~ sex + ed + race
earn ~ sex + age + race
earn ~ ed + age + race
```

```
earn ~ height + sex + ed + age
earn ~ height + sex + ed + race
earn ~ height + sex + age + race
earn ~ height + ed + age + race
earn ~ sex + ed + age + race
```

```
earn ~ height + sex + ed + age + race
```

Best Subset selection in R

1. Find best subsets of each size with `regsubsets` in the `leaps` package

2. Compare different sizes with Adjusted R^2, C_p or BIC.

regsubsets syntax

in the leaps package

regsubsets

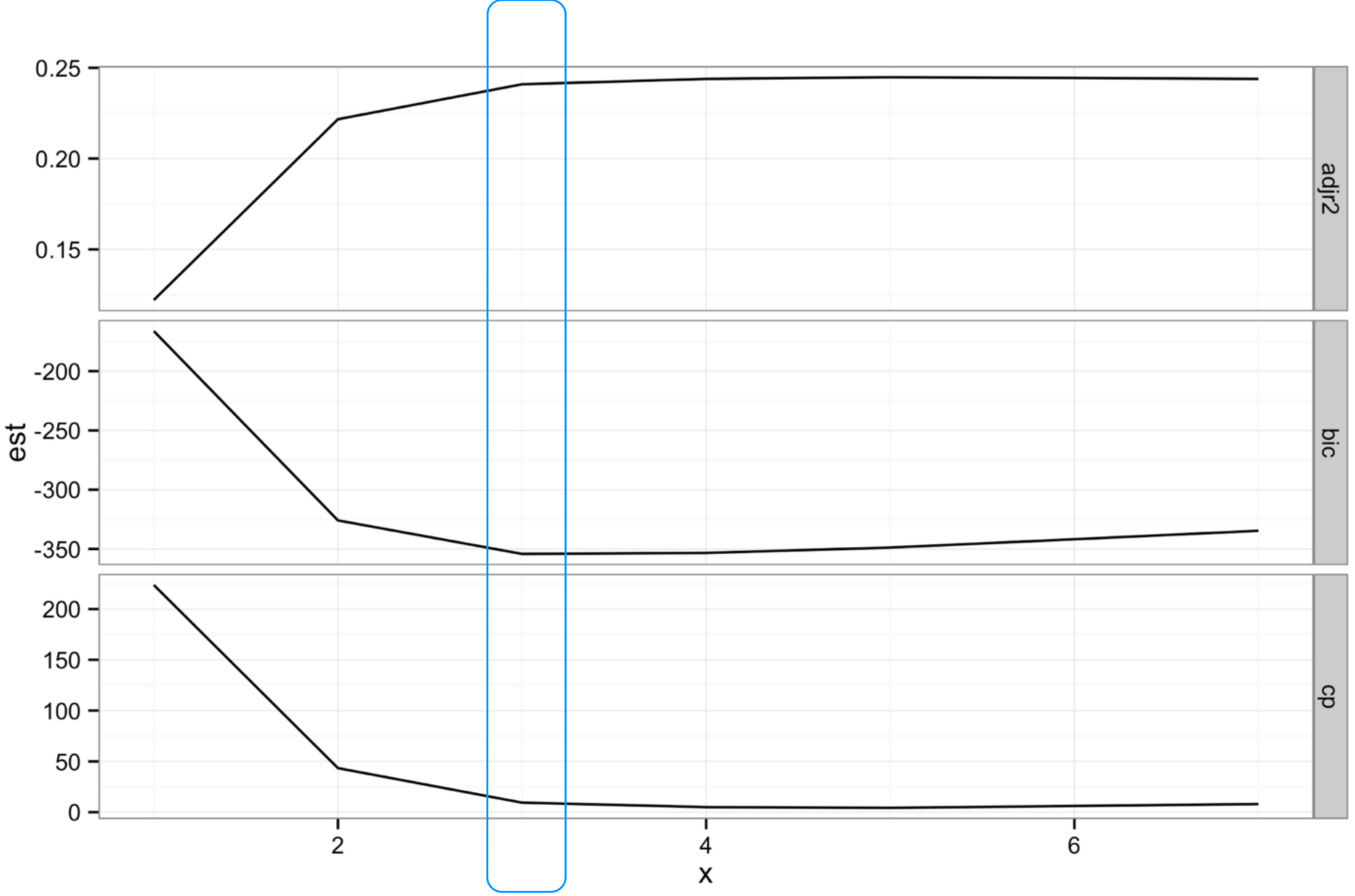
same arguments as loess, lm, etc...

```
library(leaps)
subs <- regsubsets(earn ~ ., data = wages)
```

Put all the variables you want to consider in the formula

```
library(leaps)
subs <- regsubsets(earn ~ ., data = heights)
summary(subs)
# Subset selection object
# Call: regsubsets.formula(earn ~ ., data = wages)
# ...
# Selection Algorithm: exhaustive
#          height sexfemale racewhite raceblack raceother ed  age
# 1  ( 1 ) " "    " "       " "       " "       " "       "*" " "
# 2  ( 1 ) " "    "*"       " "       " "       " "       "*" " "
# 3  ( 1 ) " "    "*"       " "       " "       " "       "*" "*"
# 4  ( 1 ) "*"    "*"       " "       " "       " "       "*" "*"
# 5  ( 1 ) "*"    "*"       "*"       " "       " "       "*" "*"
# 6  ( 1 ) "*"    "*"       "*"       "*"       " "       "*" "*"
# 7  ( 1 ) "*"    "*"       "*"       "*"       "*"       "*" "*"
```

```
summary(subs)$adjr2
# [1] 0.1221245 0.2216757 0.2409214 0.2438843 0.2448159
# [6] 0.2443771 0.2438716
summary(subs)$cp
# [1] 223.716019  43.392152   9.365024   4.977063
4.285389
# [6]   6.082783   8.000000
summary(subs)$bic
# [1] -166.1583 -325.9095 -354.2102 -353.3775 -348.8526
# [6] -341.8272 -334.6814

df <- data.frame(
  est = c(summary(subs)$adjr2, summary(subs)$cp,
    summary(subs)$bic),
  x = rep(1:7, 3),
  type = rep(c("adjr2", "cp", "bic"), each = 7)
)
```




```
qplot(x, est, data = df, geom = "line") +
  theme_bw() + facet_grid(type ~ ., scales = "free_y")
```

the model with 3 variables

```
coef(subs, 3)
# (Intercept)   sexfemale          ed         age
# -26906.3434 -21000.8844   4469.2720    281.2589
```

Your turn

Use `regsubsets` to find the best variables to predict `crime ~ .` in the `cts` data.

```
_________(_____ ~ _____, data = ______)
```

```
regsubsets(crime ~ ., data = cts)
```

```
regsubsets(crime ~ ., data = cts)
# Error in leaps.exhaustive(a, really.big) :
#  Exhaustive search will be S L O W, must
#  specify really.big=T
```

Best subsets is impractical for many variables

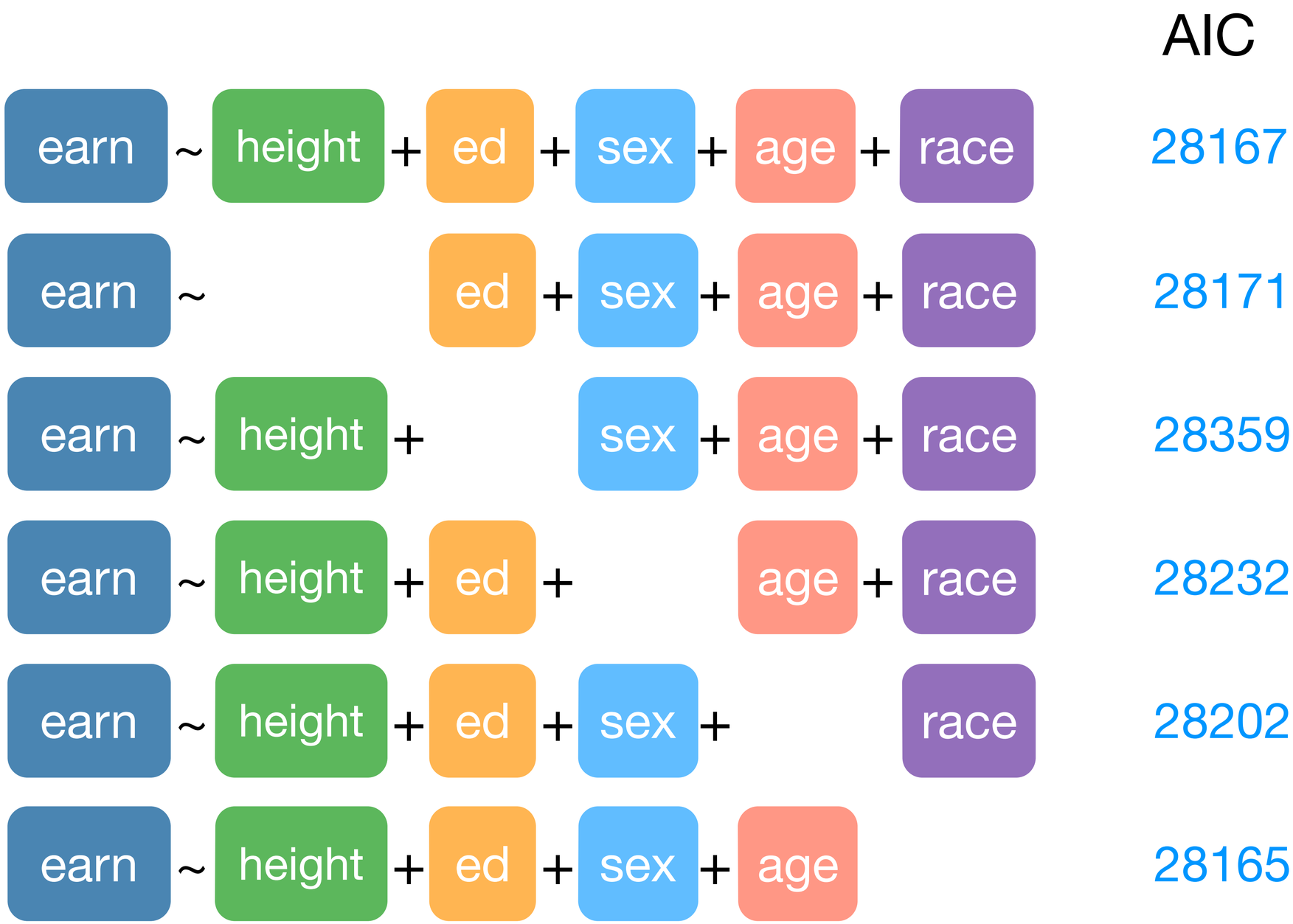
Stepwise selection

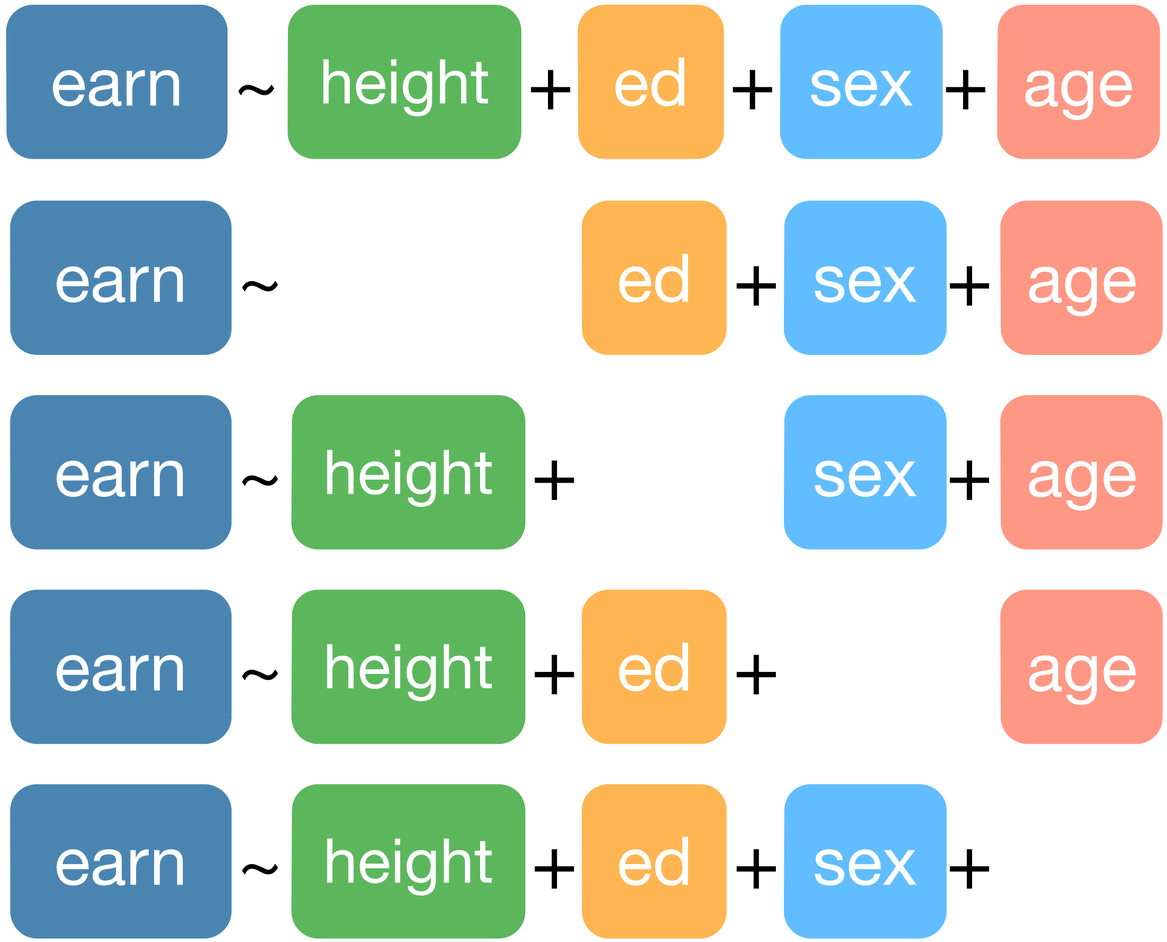
Backwards selection

Start with every variable in the model. Then delete the variable who's omission improves the test error the most. Continue until you cannot improve the test error.

Forward selection

Start with no variables in the model. Then add the variable that improves the test error the most. Continue until you cannot improve the test error.





AIC

28165

28169

28358

28227

28201

AIC

earn ~

28368

AIC

earn ~ ed +

step

step

A model object to start with

"forward", "backward", or "both"

```
step(model, scope, direction = "backward")
```

A named list of two model objects:
list(upper = the largest model you are willing to consider,
lower = the smallest model you are willing to consider)

```
start.mod <- lm(earn ~ height, data = wages)
empty.mod <- lm(earn ~ 1, data = wages)
full.mod <- lm(earn ~ ., data = wages)

step(start.mod,
  scope = list(upper = full.mod,
               lower = empty.mod),
  direction = "forward")
```

What does the output look like?

Your turn

```
cts2 <- select(cts, -state, -county)
```

Use `step` to find the best way to predict `crime` in the `cts2` data. Start by assuming that only `pop` matters.

Try using `"forward"`, `"backward"`, and `"both"`. Do you get the same results in each case?

Hint: use

```
full <- lm(crime ~ ., data = cts2)
```

```
start <- lm(crime ~ pop, data = cts2)
full <- lm(crime ~ ., data = cts2)
empty <- lm(crime ~ 1, data = cts2)
bounds <- list(upper = full, lower = empty)

step(start, bounds,  direction = "forward")
step(start, bounds,  direction = "backward")
step(start, bounds,  direction = "both")
```

What if we had a different sample?

```
cts3 <- cts2[1:110, ]
start <- full <- lm(crime ~ pop, data = cts3)
full <- lm(crime ~ ., data = cts3)
empty <- lm(crime ~ 1, data = cts3)
bounds <- list(upper = full, lower = empty)

step(start, bounds,  direction = "forward")
step(start, bounds,  direction = "backward")
step(start, bounds,  direction = "both")
```

step with cts2	step with cts3
area	area
pop	
p18_25	p18_25
p65	
nphysician	
nhospbeds	
pcollege	
ppoverty	ppoverty
punemployed	
avg_income	
tot_income	
region	region

Subset selection still tends to overfit the data

Using the data to guide the analysis is almost as dangerous as not doing so.

– Frank Harrell

Penalized regression (feature selection)

LASSO

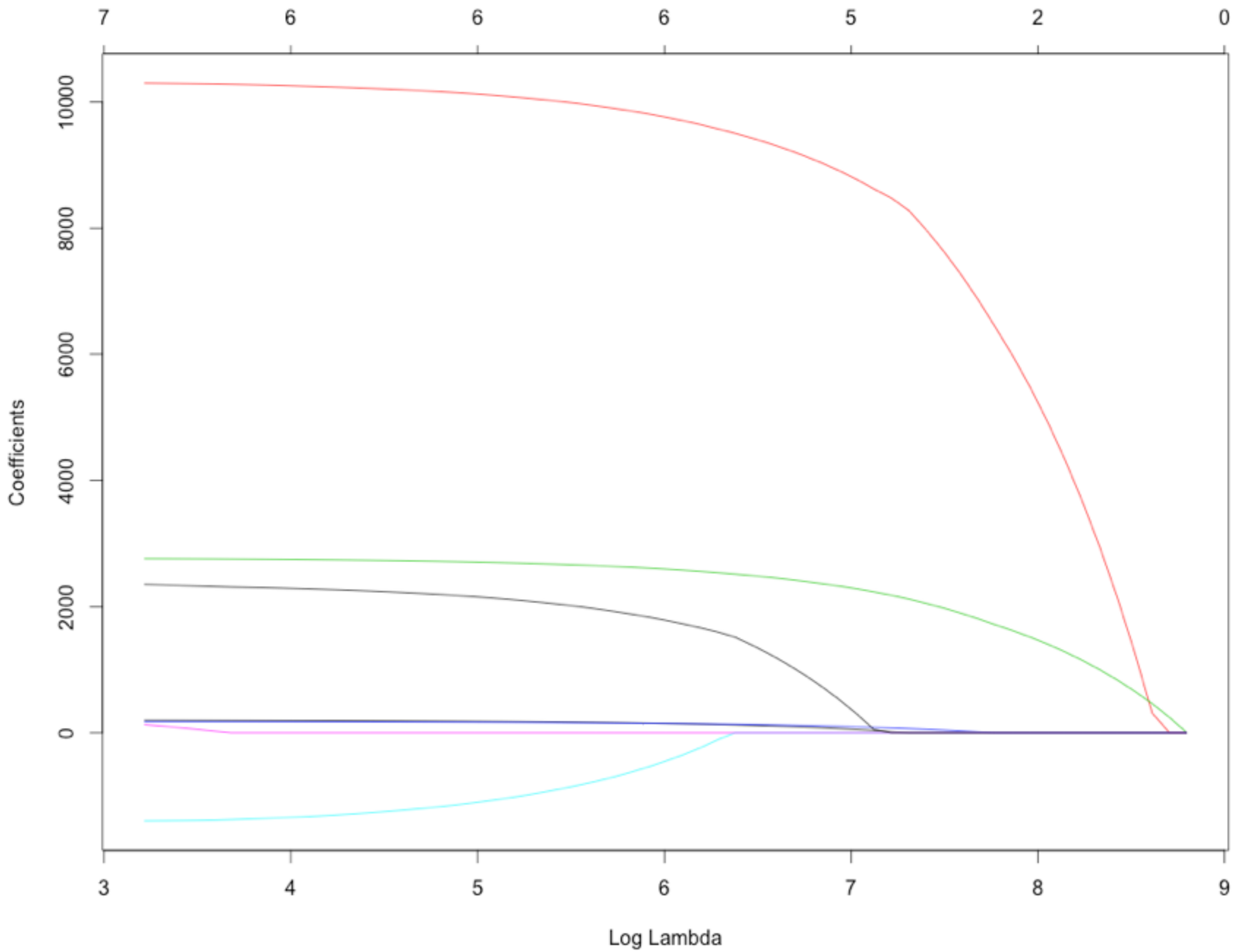
Linear regression finds the α and β that minimize

$$f(x) - (\alpha + \beta x)$$

LASSO finds the line that minimizes

$$f(x) - \left(\alpha + \beta x + \lambda \sum |\beta|\right)$$

forces model to set $\beta_j = 0$ for x_j's that contribute little



LASSO removes variables sequentially as λ increases

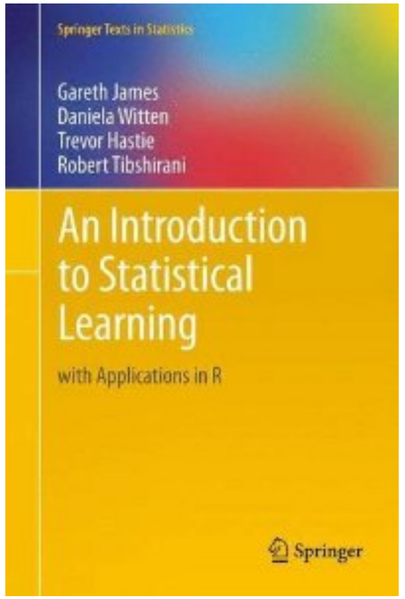
Advantages of LASSO

1. Performs better than stepwise selection
2. Performs variable selection to any number of variables
3. Can create models where p > n (simple linear regression cannot)
4. Statistically well founded
5. Usually fast

LASSO can be performed in R with the `glmnet` function in the `glmnet` package.

"An Introduction to Statistical Learning" provides a good guide to the process.

http://amzn.com/1461471370



Variable Selection Recap

1. Let theory dictate variables
2. Let data dictate variables
   A. Choose an optimization criteria:
      i. Mallow's Cp
      ii. AIC - `AIC`
      iii. BIC - `BIC`
      iv. Adjusted R^2 - `summary`
   B. Search for optimal subset
      i. Best subsets - `leaps:regsubsets`
      ii. stepwise - `step`
3. Let model algorithm dictate variables
   A. Penalized regression (LASSO) - `glmnet:glmnet`

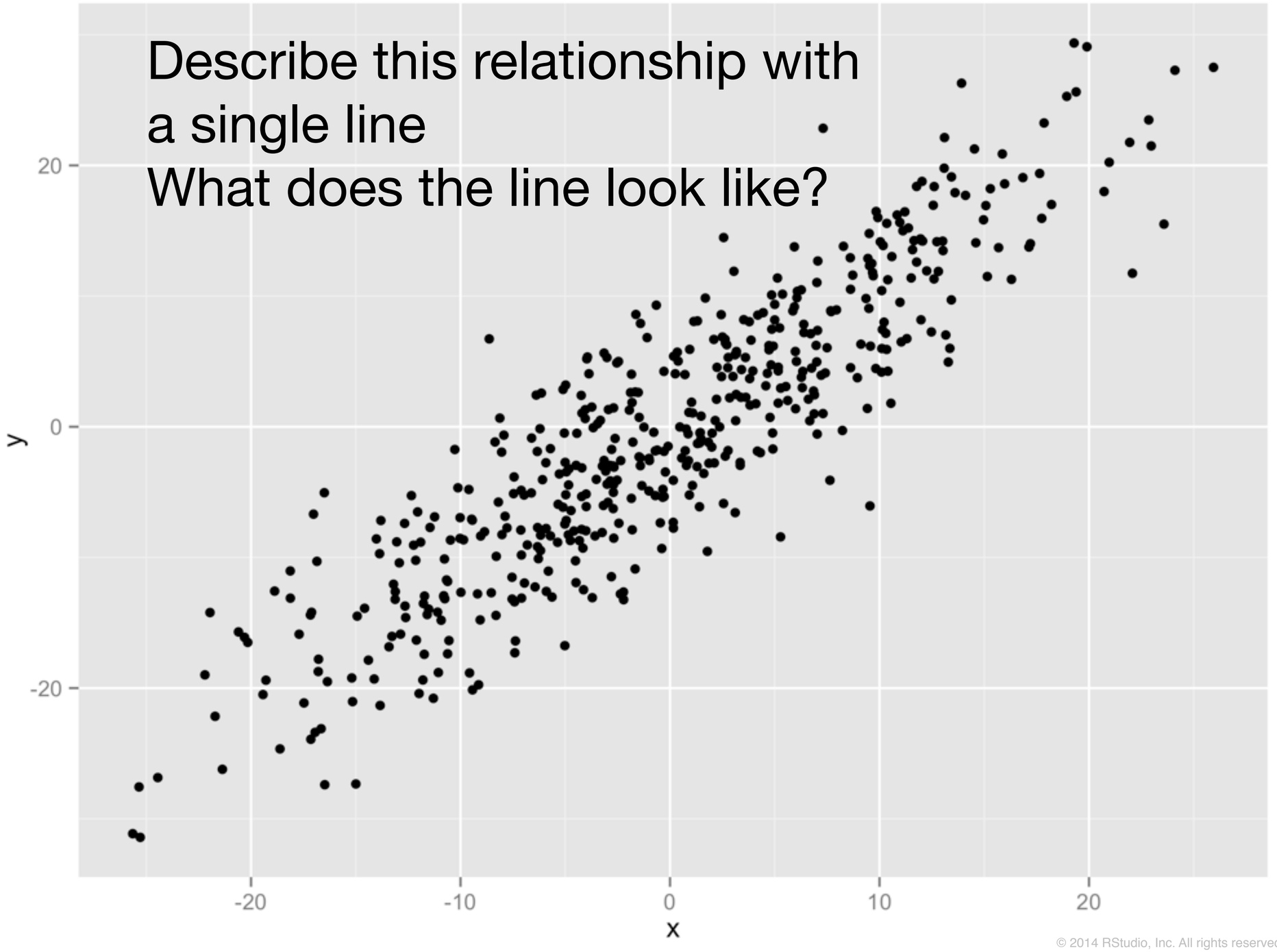
Non-linear relationships

Your turn

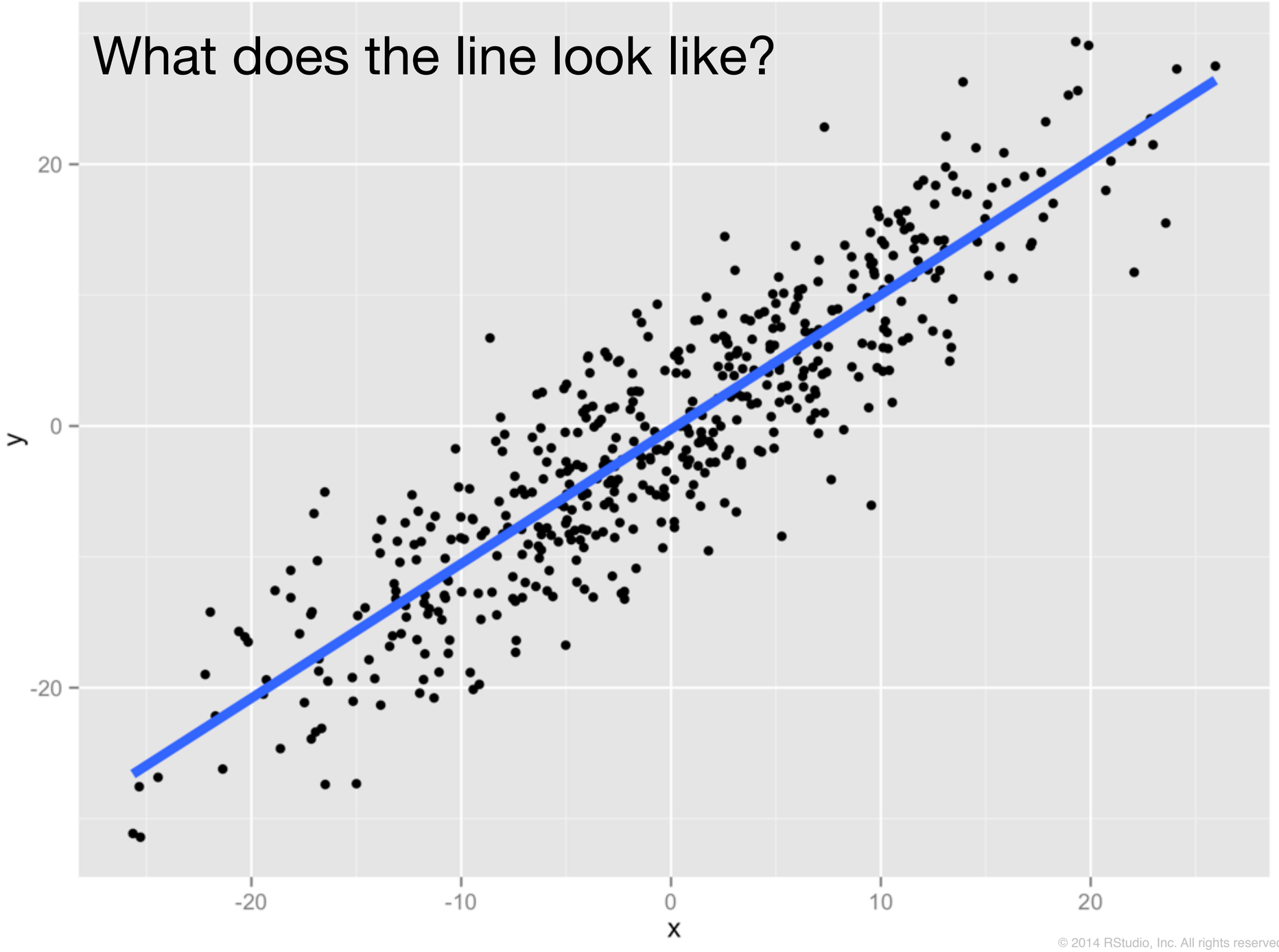
For each of the following slides, you'll get several seconds to think about each question. Then you'll see the answer.

Describe this relationship with a single line

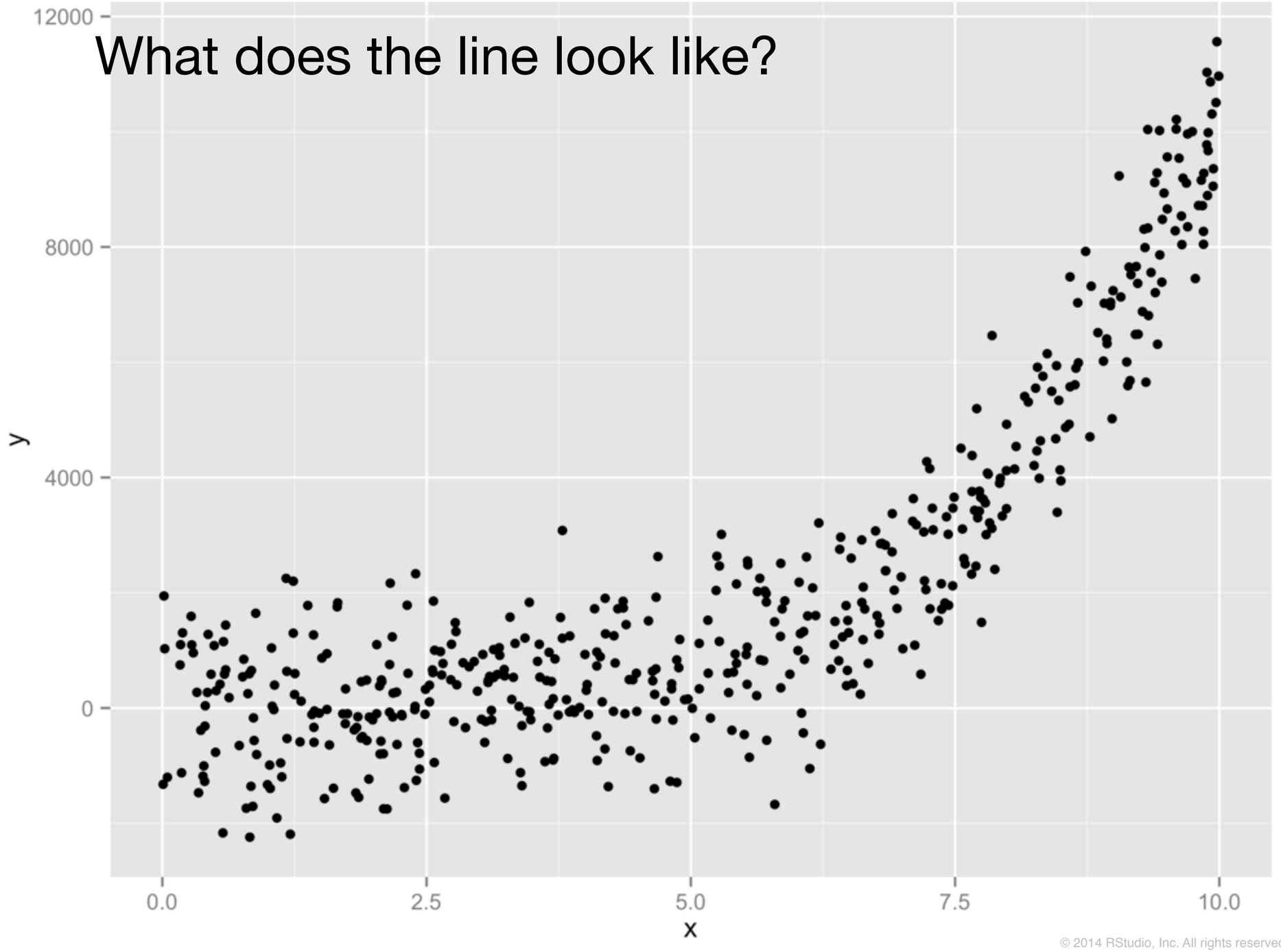
What does the line look like?



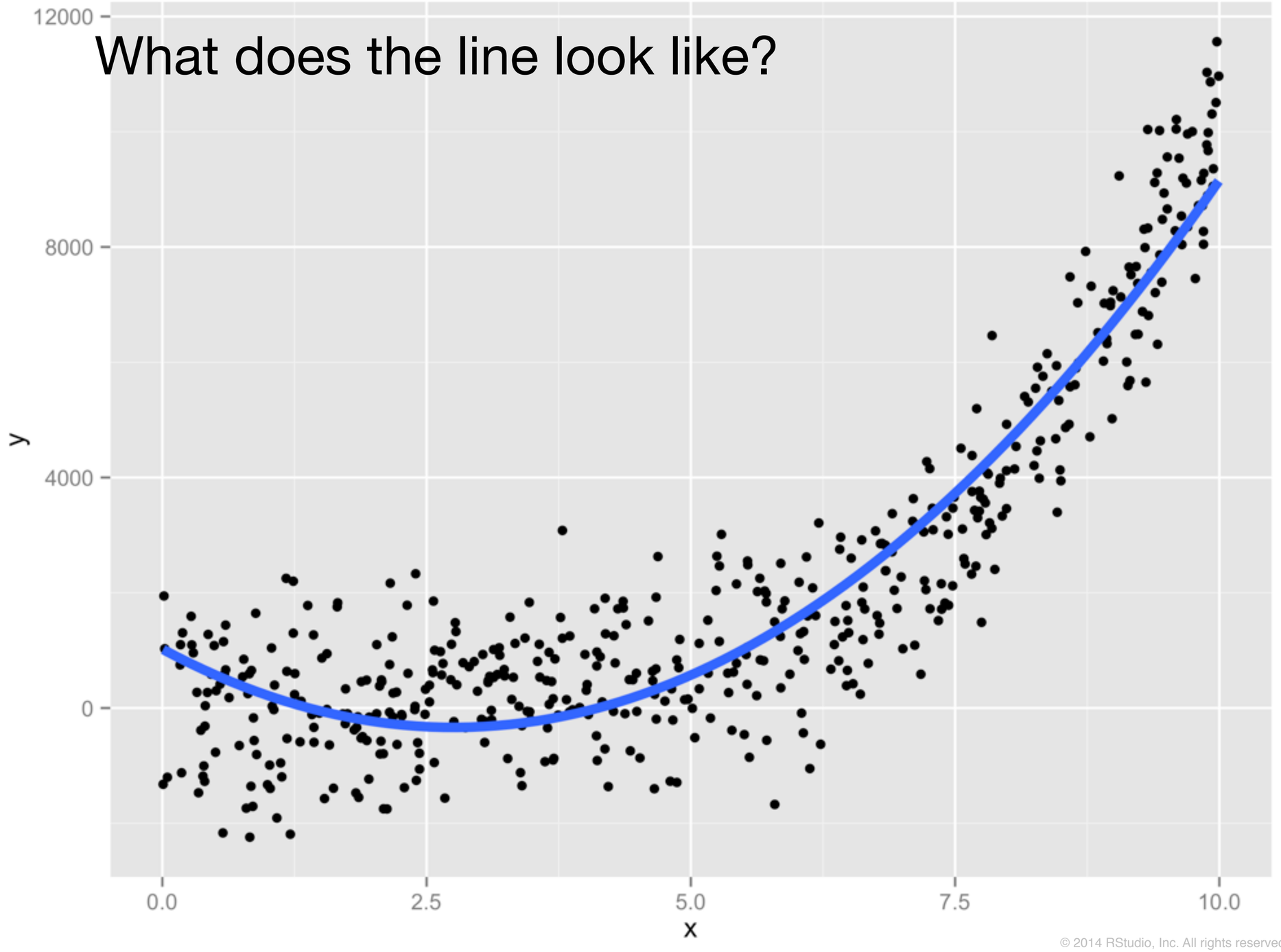
What does the line look like?

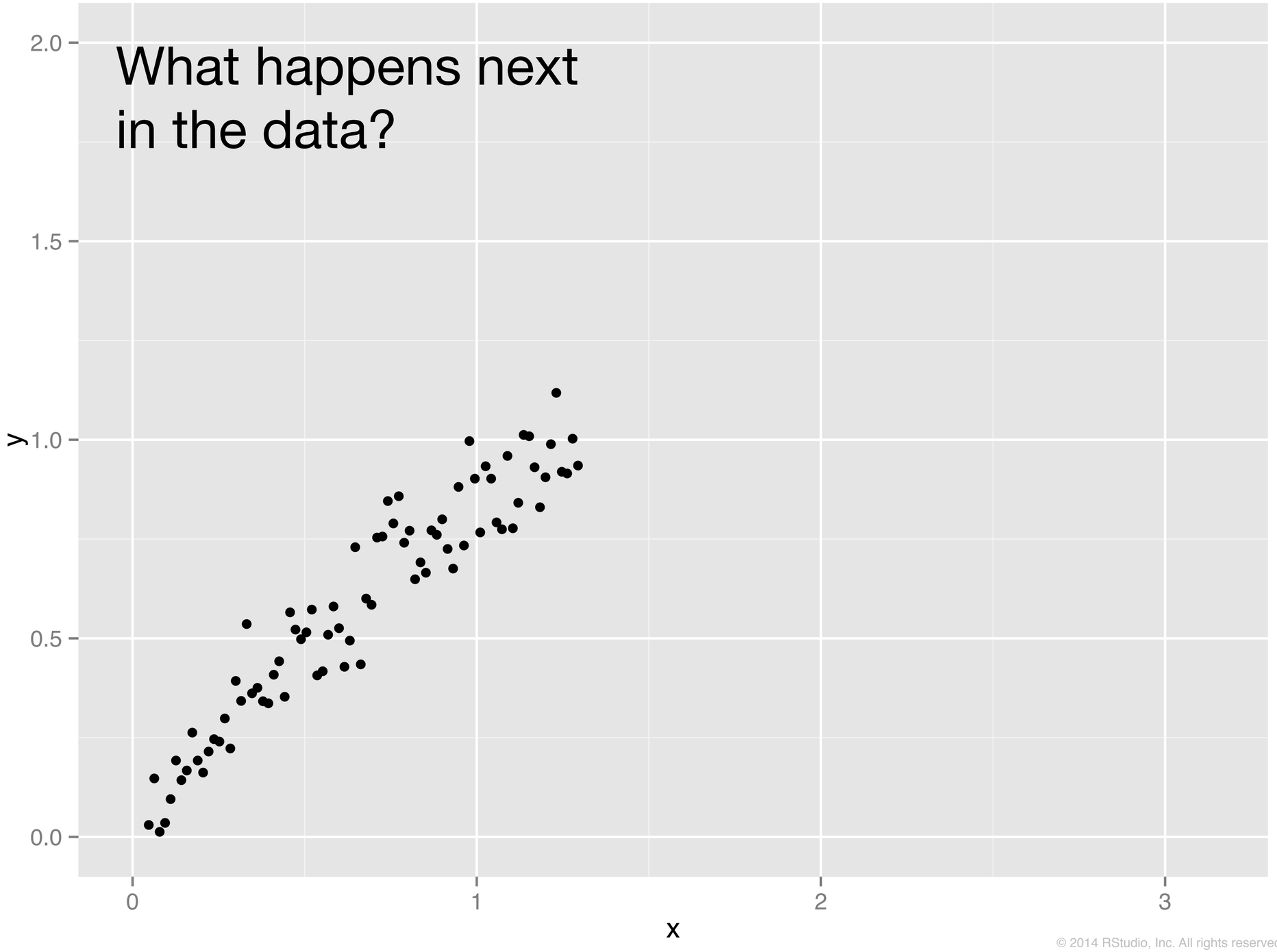


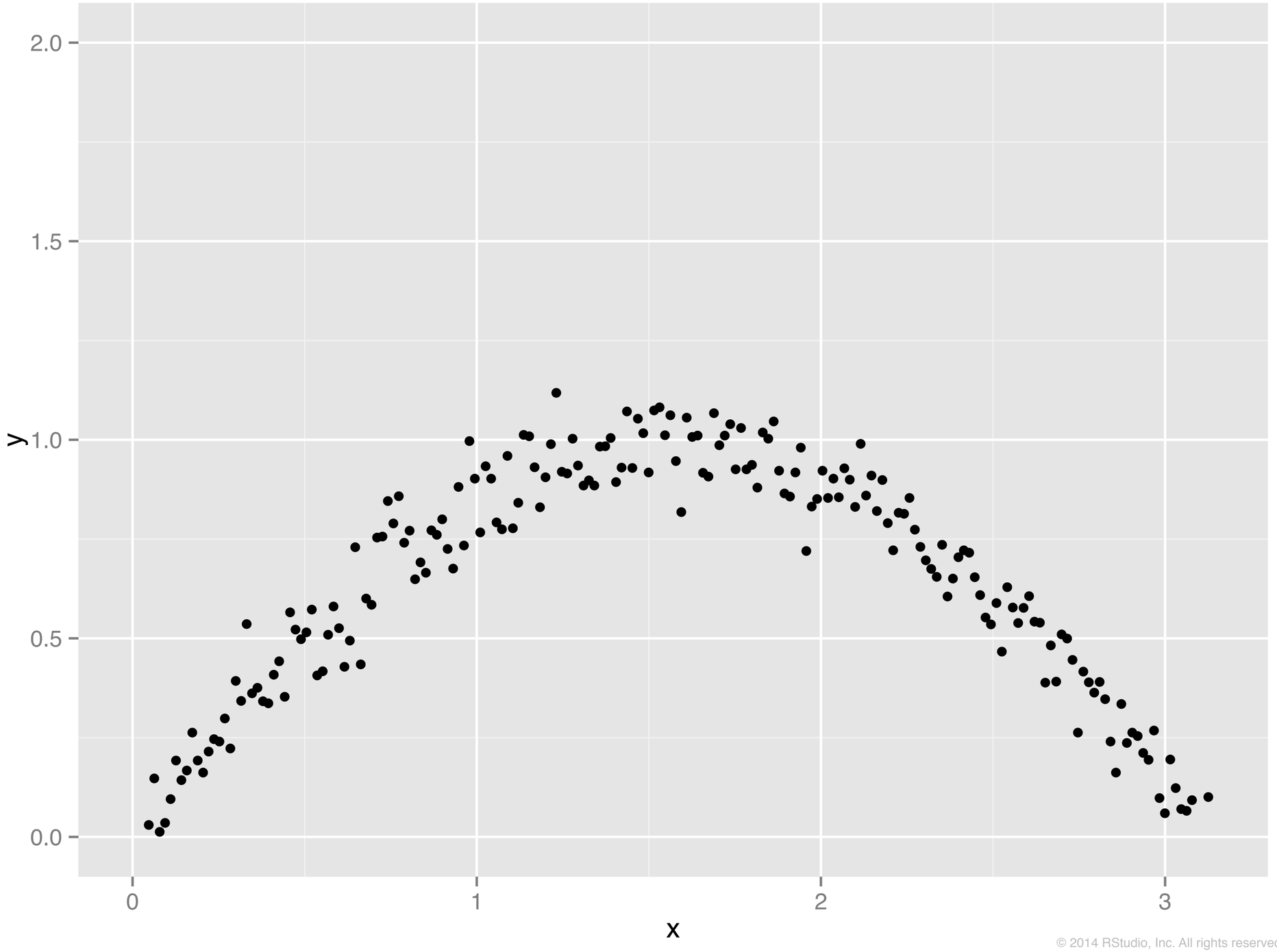
What does the line look like?



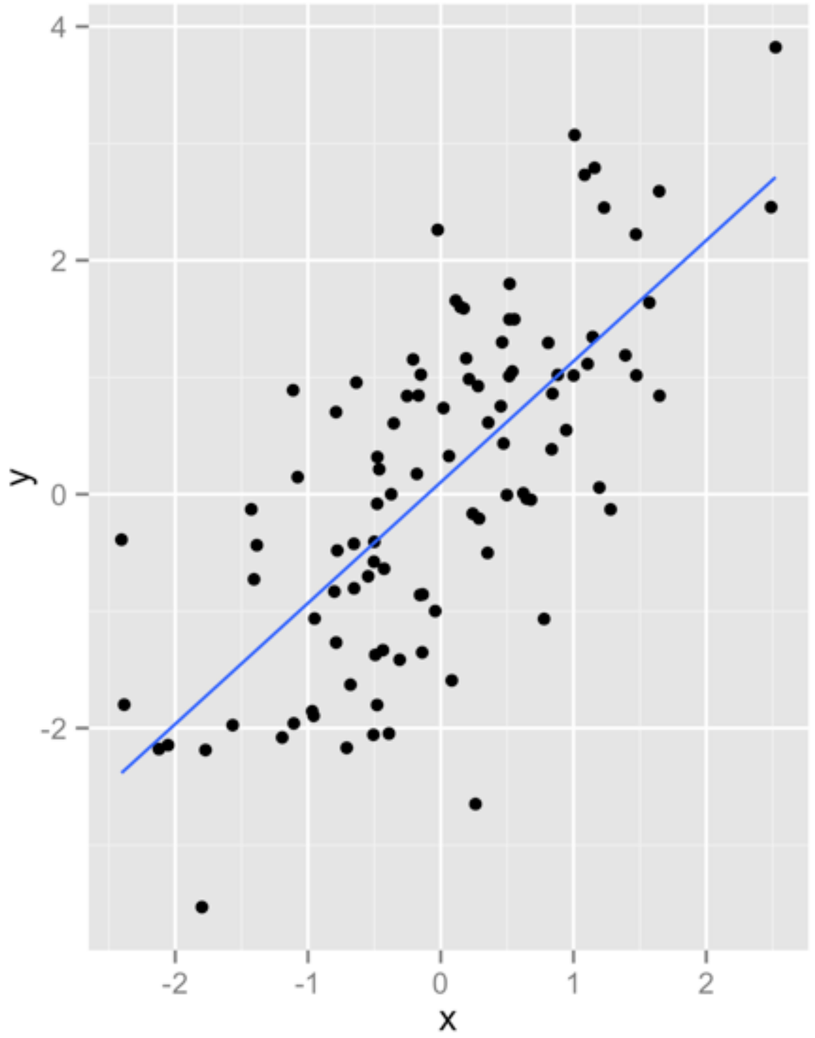
What does the line look like?





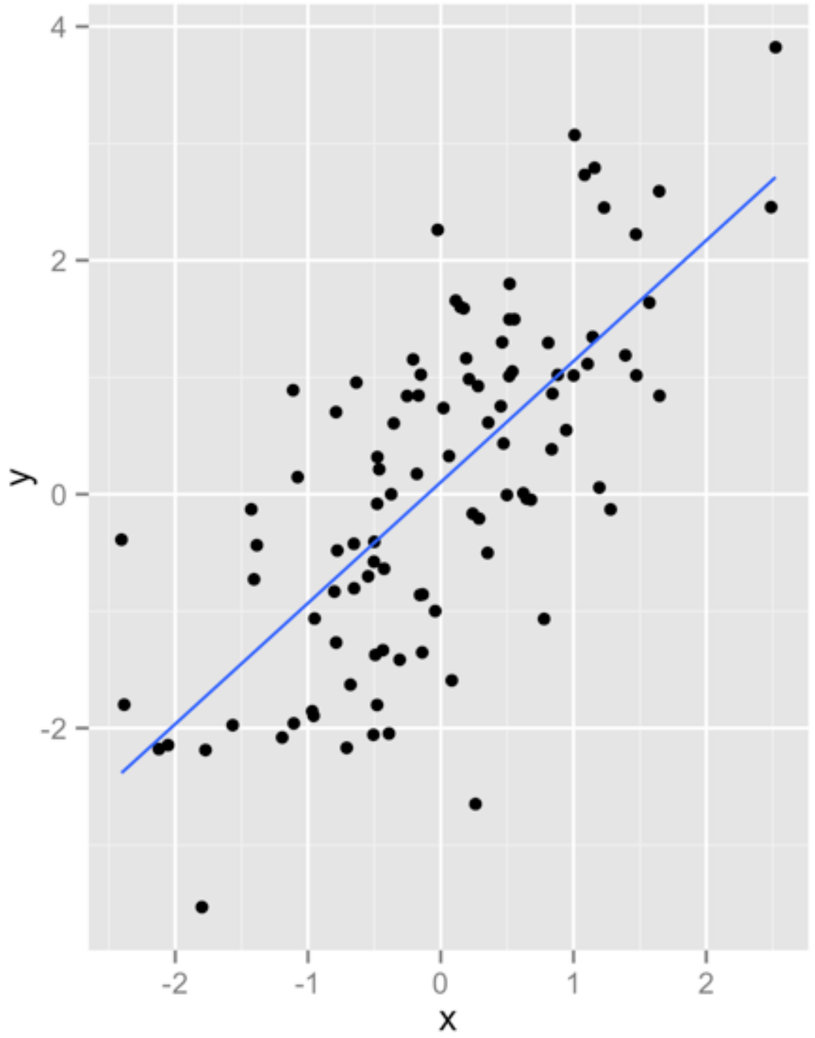


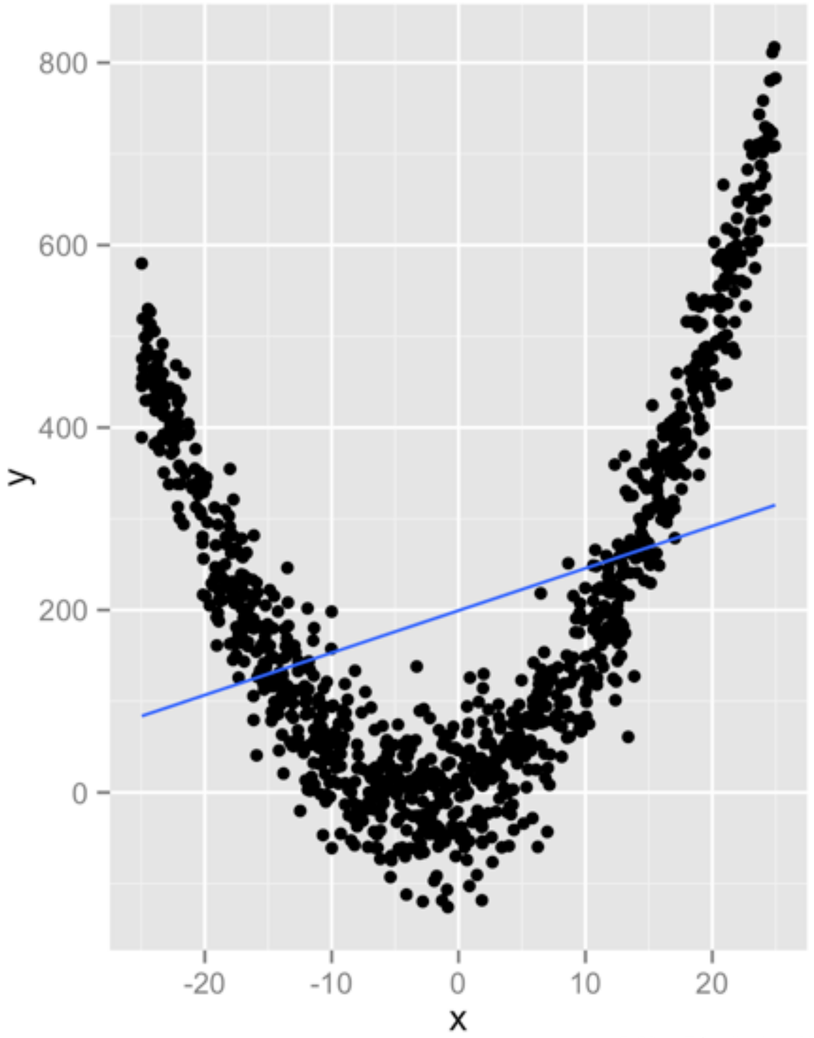
Linear models



- easy to interpret
- easy to calculate
- reasonable approximation over small areas

Linear models





Options

more flexible

$$y = \alpha + \beta x + \epsilon$$ straight lines

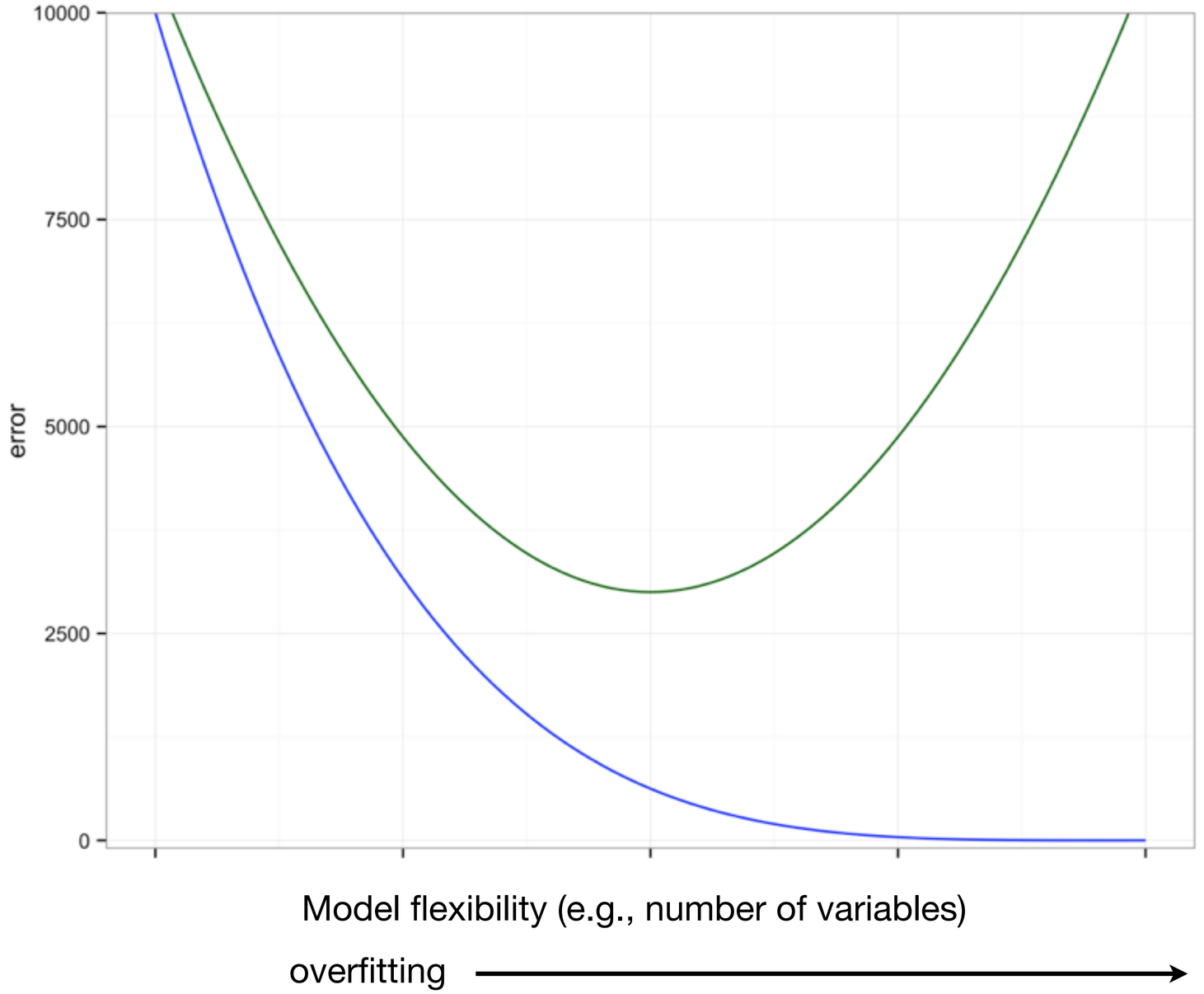
$$y = \alpha + \beta f(x) + \epsilon$$ transformations

$$y = \alpha + \beta_1 x + \beta_2 x^2 + \ldots + \epsilon$$ polynomials

$$y = \alpha + \beta ns(x) + \epsilon$$ splines

$$y = \alpha + \beta s(x) + \epsilon$$ smoothing functions

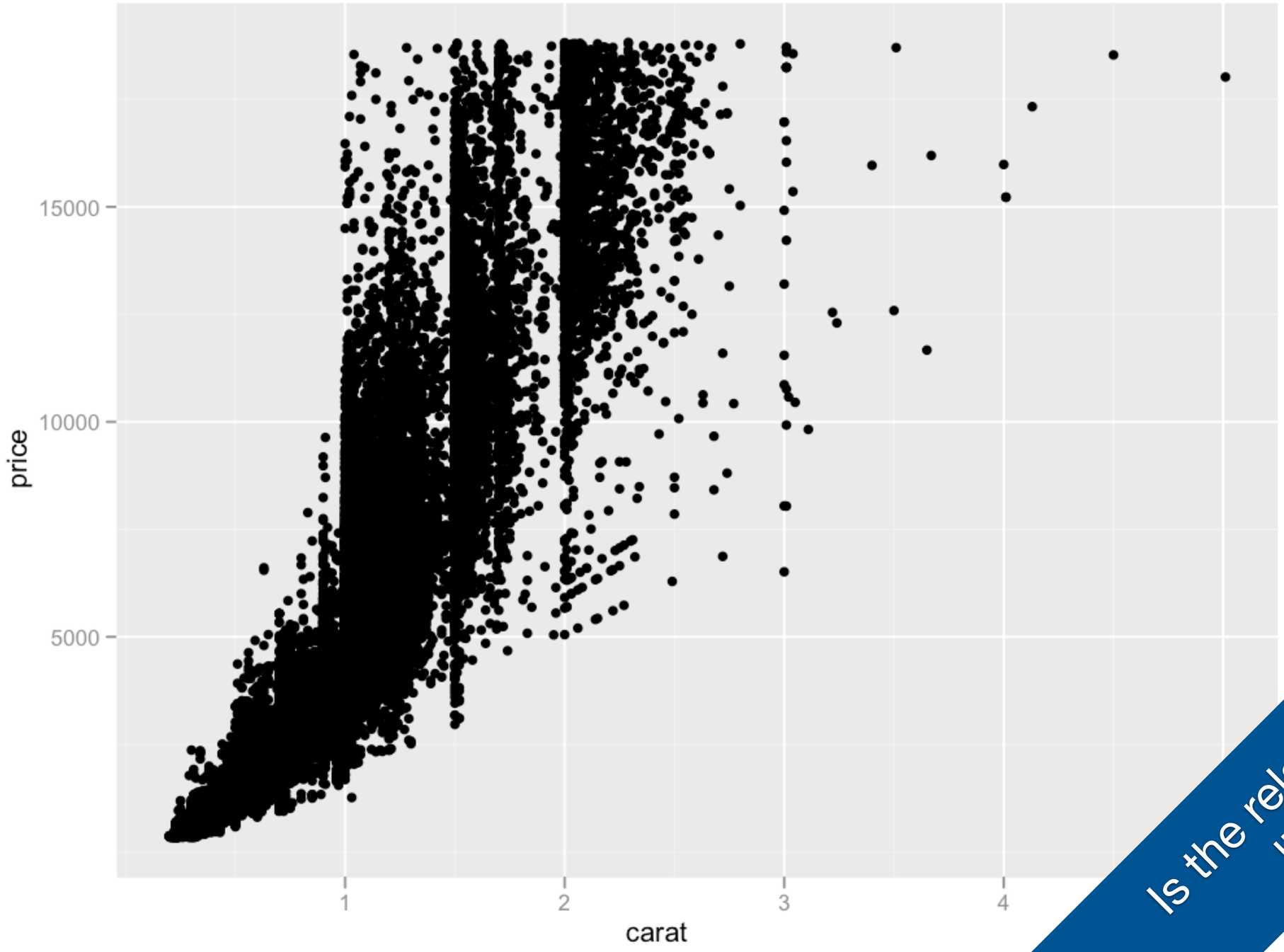
Predictive accuracy



Test error
(predictions
vs. new data)

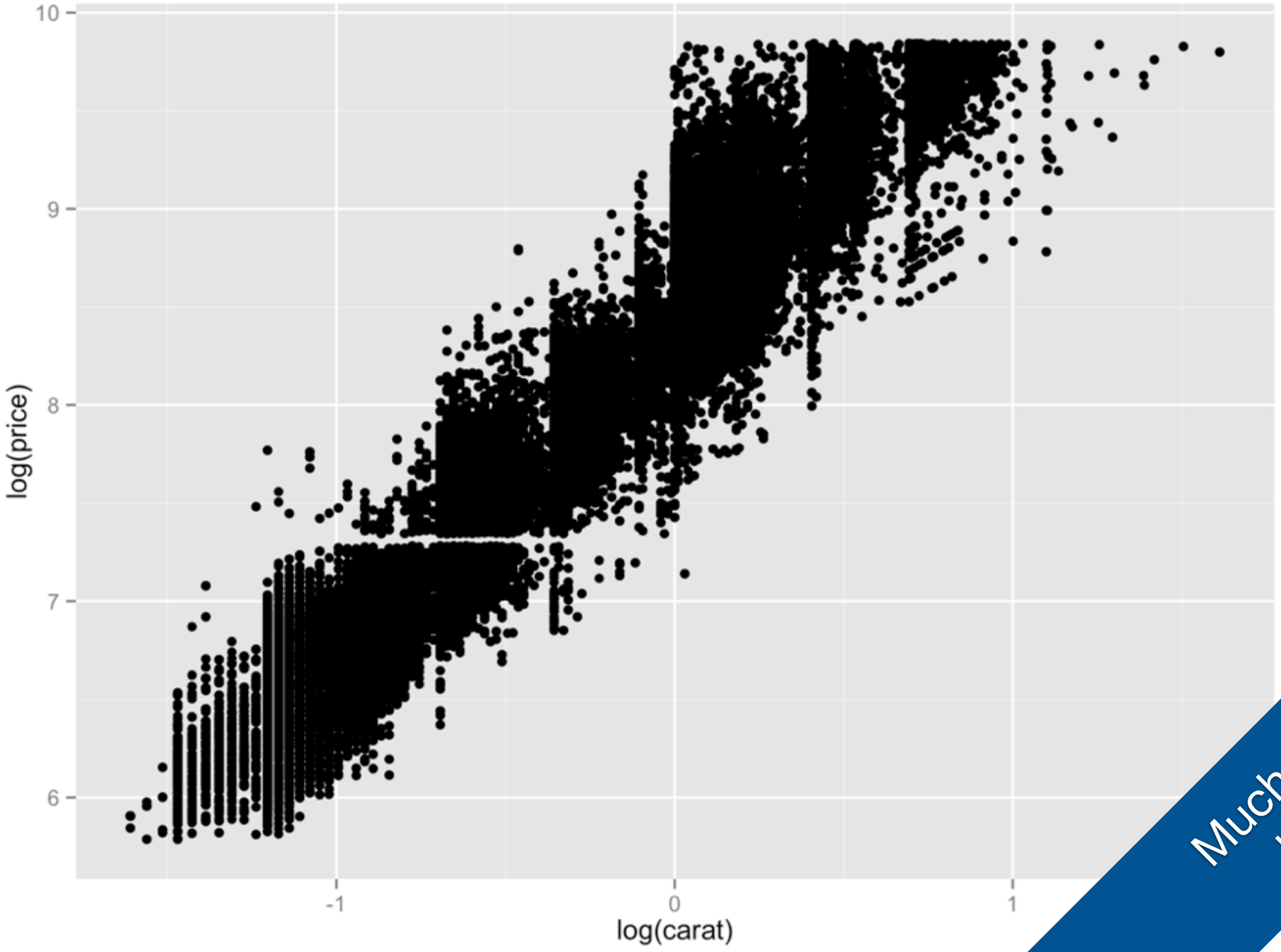
Training error
(model vs.
available data)

Transformations




```
qplot(carat, price, data = diamonds)
```

Is the relationship linear?




```
qplot(log(carat), log(price), data = diamond
```

Much more linear

Your turn

How would you fit `log(price)` to `log(carat)` with a linear model in R?

Give it a try. Did it work?

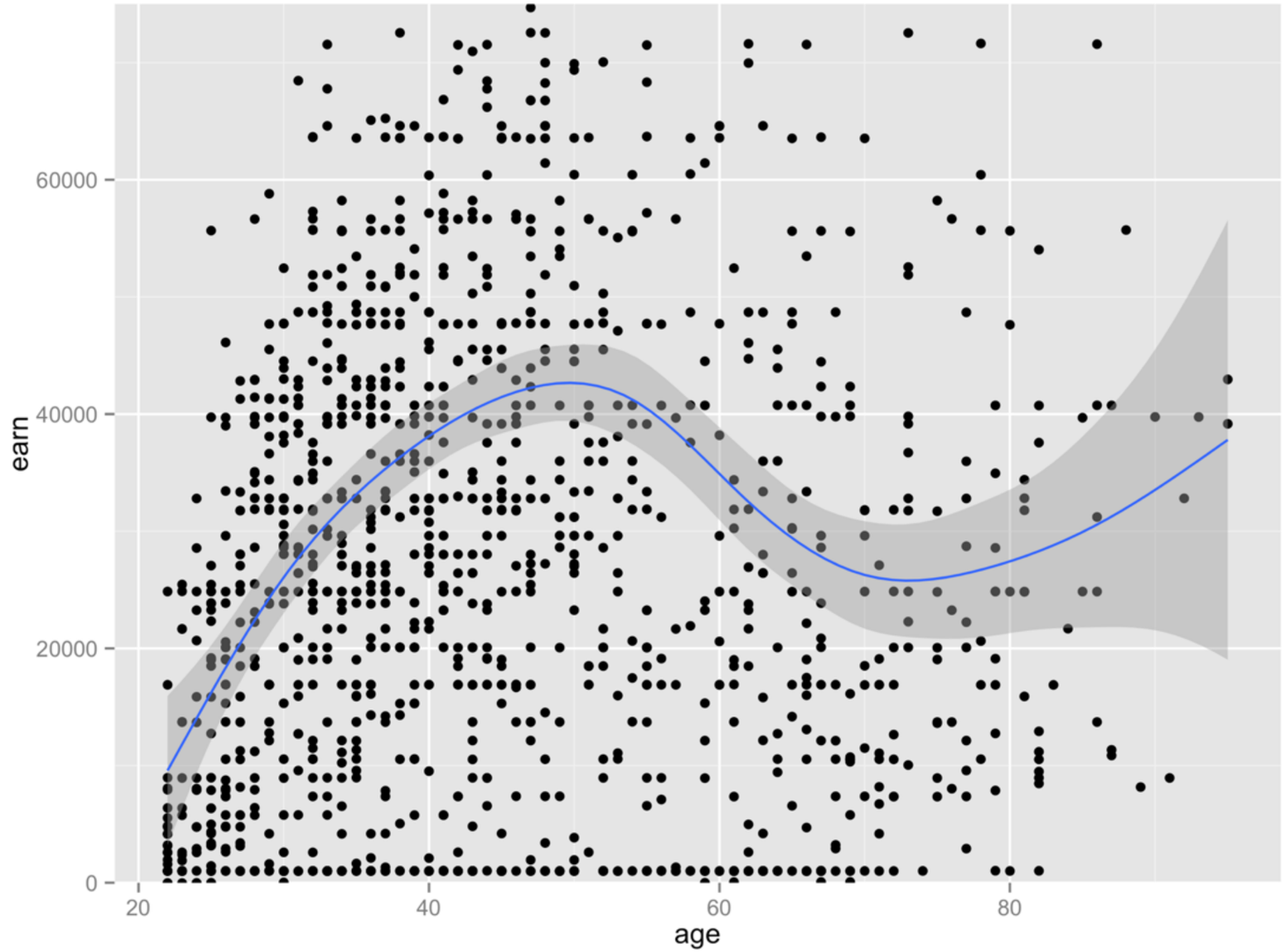
```
lm(_____ ~ _____, data = diamonds)
```

R can evaluate functions in the formula

```
mod <- lm(log(price) ~ log(carat), data = diamonds)
summary(mod)
```

```
Call:
lm(formula = log(price) ~ log(carat), data = diamonds)

Residuals:
    Min      1Q  Median      3Q     Max
-1.5083 -0.1695 -0.0059  0.1664  1.3379

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  8.44866    0.00136    6191   <2e-16 ***
log(carat)   1.67582    0.00193     867   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.26 on 53938 degrees of freedom
Multiple R-squared:  0.933, Adjusted R-squared:  0.933
F-statistic: 7.51e+05 on 1 and 53938 DF,  p-value: <2e-16
```

What if no handy transformation exists?




```
qplot(age, earn, data = wages) + geom_smooth() + coord_cartesian(ylim = c(0, 75000))
```

Polynomials

polynomials

Approach: instead of modeling x, model a polynomial of x.

$$y = \alpha + \beta x + \epsilon$$

$$y = \alpha + \beta_1 x + + \beta_2 x^2 + \ldots + \beta_k x^k + \epsilon$$

You can approximate any continuous curve in x with the right combination of x terms.

poly

```
mod <- lm(earn ~ poly(age, 3), data = wages)
```

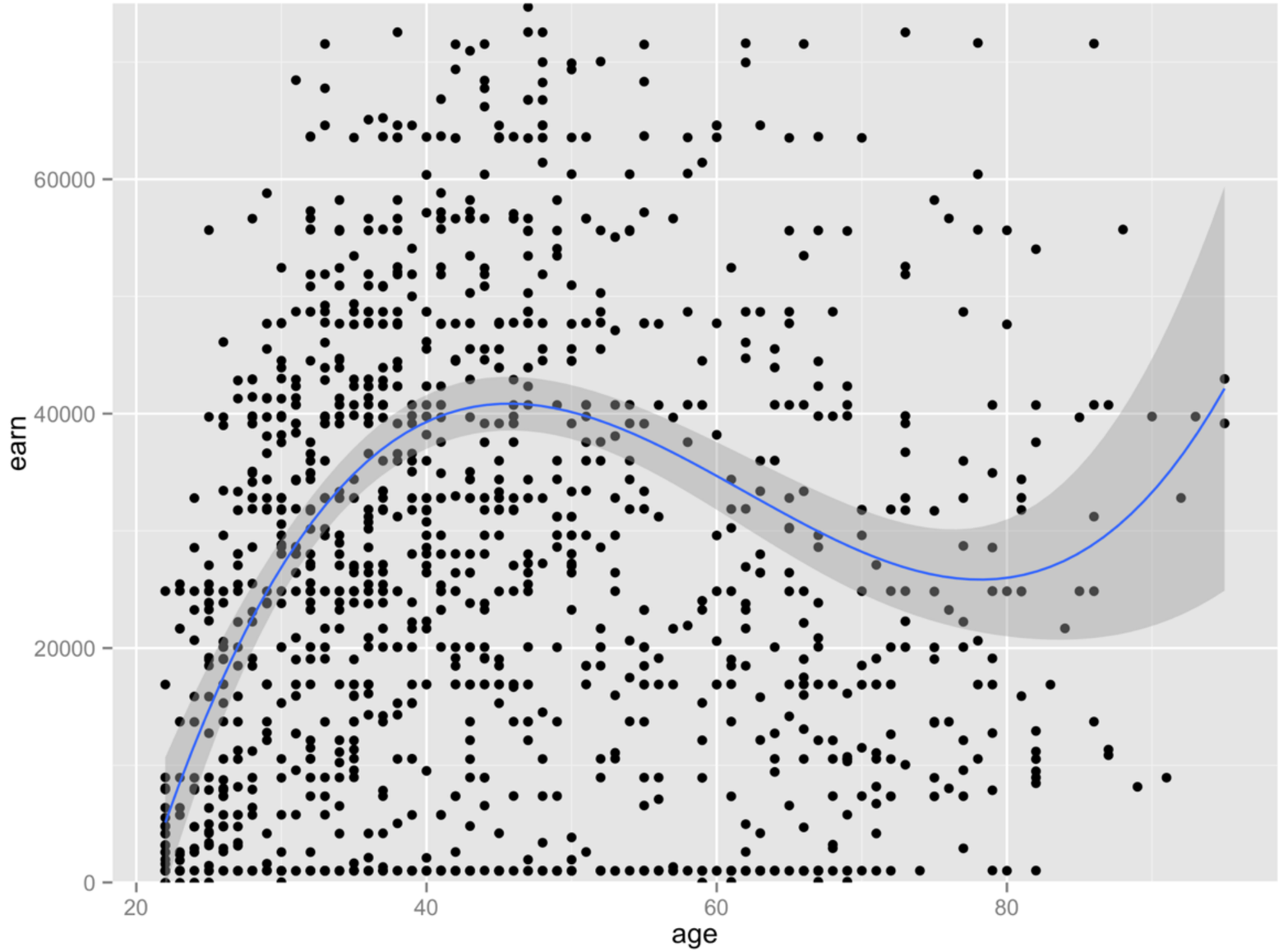
poly function to create polynomial

x variable

degree of polynomial to use

degree	polynomial
`poly(x, 1)`	$\beta_1 x$
`poly(x, 2)`	$\beta_1 x + \beta_2 x^2$
`poly(x, 3)`	$\beta_1 x + \beta_2 x^2 + \beta_3 x^3$
...	...

```
summary(mod)
# Call:
# lm(formula = earn ~ poly(age, 3), data = wages)

# Residuals:
#    Min     1Q Median     3Q    Max
# -39859 -18219  -4120  11641 280034

# Coefficients:
#               Estimate Std. Error t value Pr(>|t|)
# (Intercept)    32446.3      811.1  40.002  < 2e-16 ***
# poly(age, 3)1   85866.8    30120.5   2.851  0.00443 **
# poly(age, 3)2 -243937.9    30120.5  -8.099 1.21e-15 ***
# poly(age, 3)3  178818.6    30120.5   5.937 3.68e-09 ***
# ---
# Signif. codes:
# 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

# Residual standard error: 30120 on 1375 degrees of freedom
# Multiple R-squared:  0.07343, Adjusted R-squared:  0.0714
# F-statistic: 36.32 on 3 and 1375 DF,  p-value: < 2.2e-16
```




```
qplot(age, earn, data = heights) +
  geom_smooth(method = lm, formula = y ~ poly(x, 3)) +
  coord_cartesian(y = c(0, 75000))
```

```
+ geom_smooth(method = lm, formula = y ~ poly(x, 3))
```

always use "y" and "x"

anova

`anova` compares nested models and tests whether one is an improvement on the one before.

To use anova

1. fit each nested sub-model
2. compare with `anova`

```
mod1 <- lm(earn ~ age, data = heights)
mod2 <- lm(earn ~ poly(age, 2), data = heights)
mod3 <- lm(earn ~ poly(age, 3), data = heights)
anova(mod1, mod2, mod3)
```

```
# Analysis of Variance Table

# Model 1: earn ~ age
# Model 2: earn ~ poly(age, 2)
# Model 3: earn ~ poly(age, 3)
#   Res.Df        RSS Df  Sum of Sq      F    Pr(>F)
# 1   1377 1.3389e+12
# 2   1376 1.2794e+12  1 5.9506e+10 65.590 1.213e-15 ***
# 3   1375 1.2475e+12  1 3.1976e+10 35.245 3.675e-09 ***
# ---
# Signif. codes:
# 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Better than

Better than

polynomials

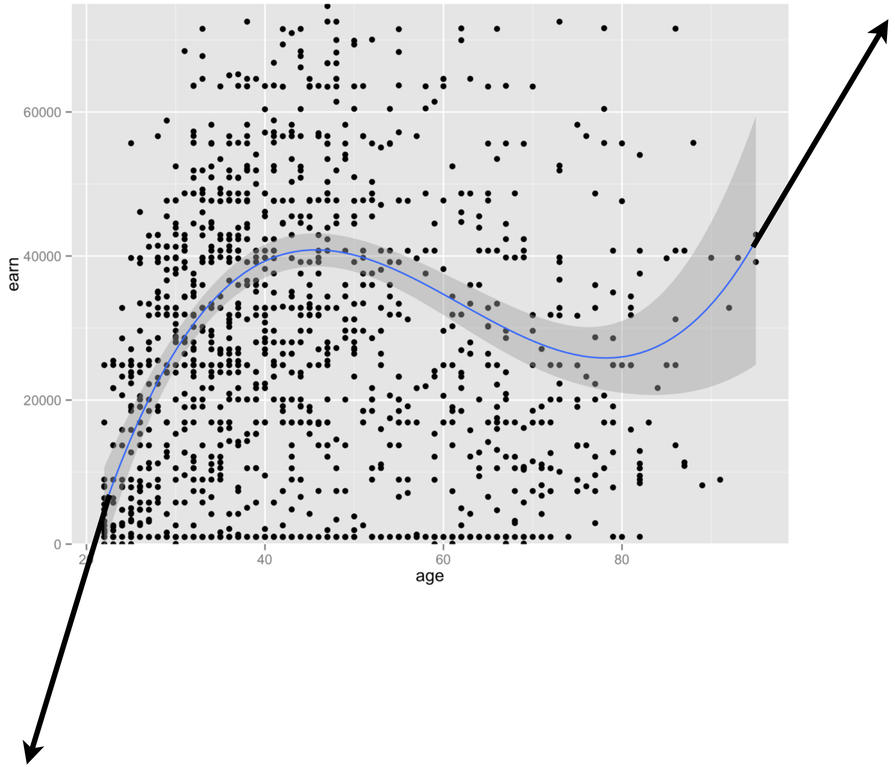
Interpret: use the cumulative effect of x coefficients on y

Visualize: plot predictions, or use `geom_smooth` with `formula` argument

Test: test with `anova`

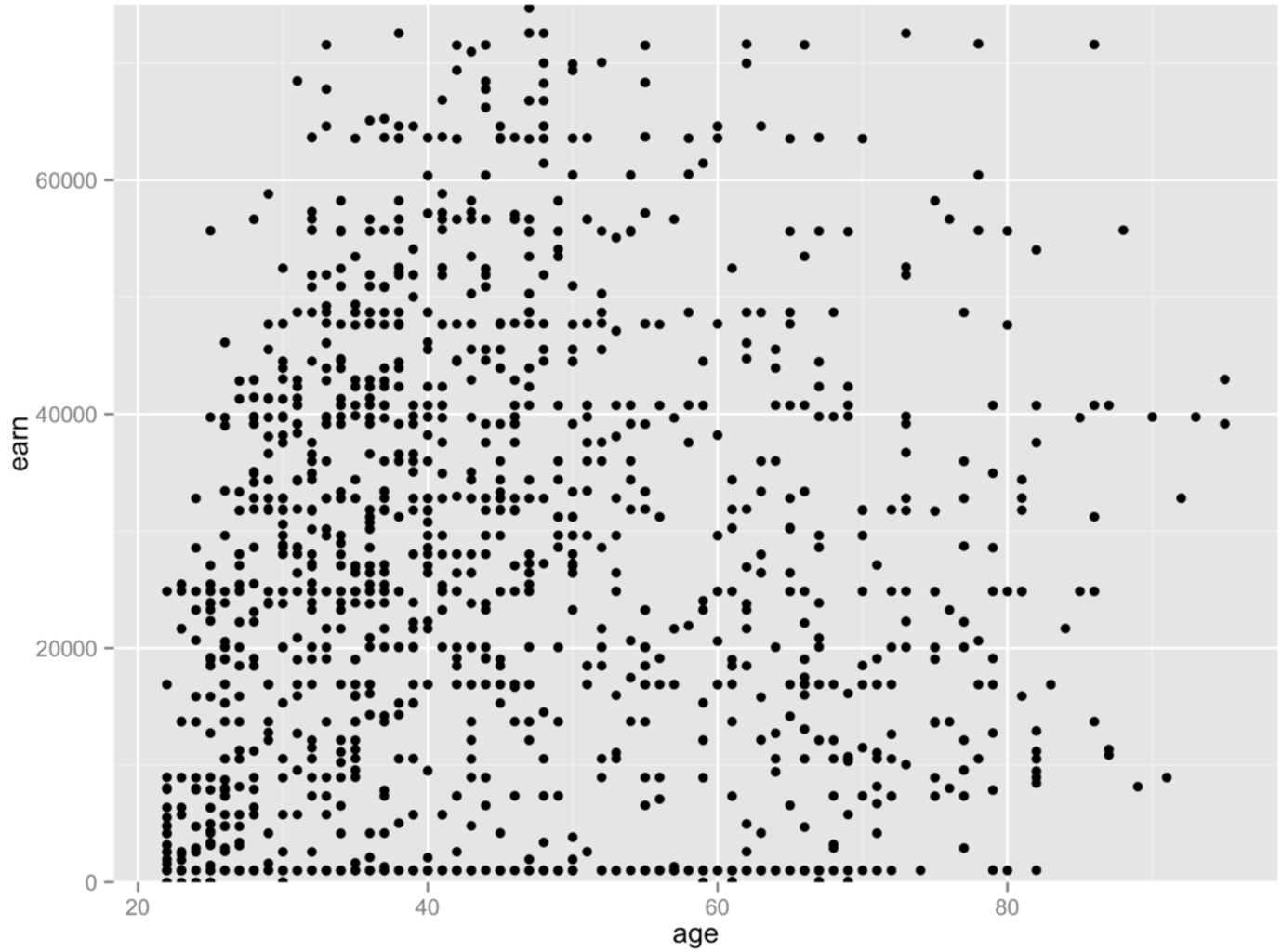
Warning

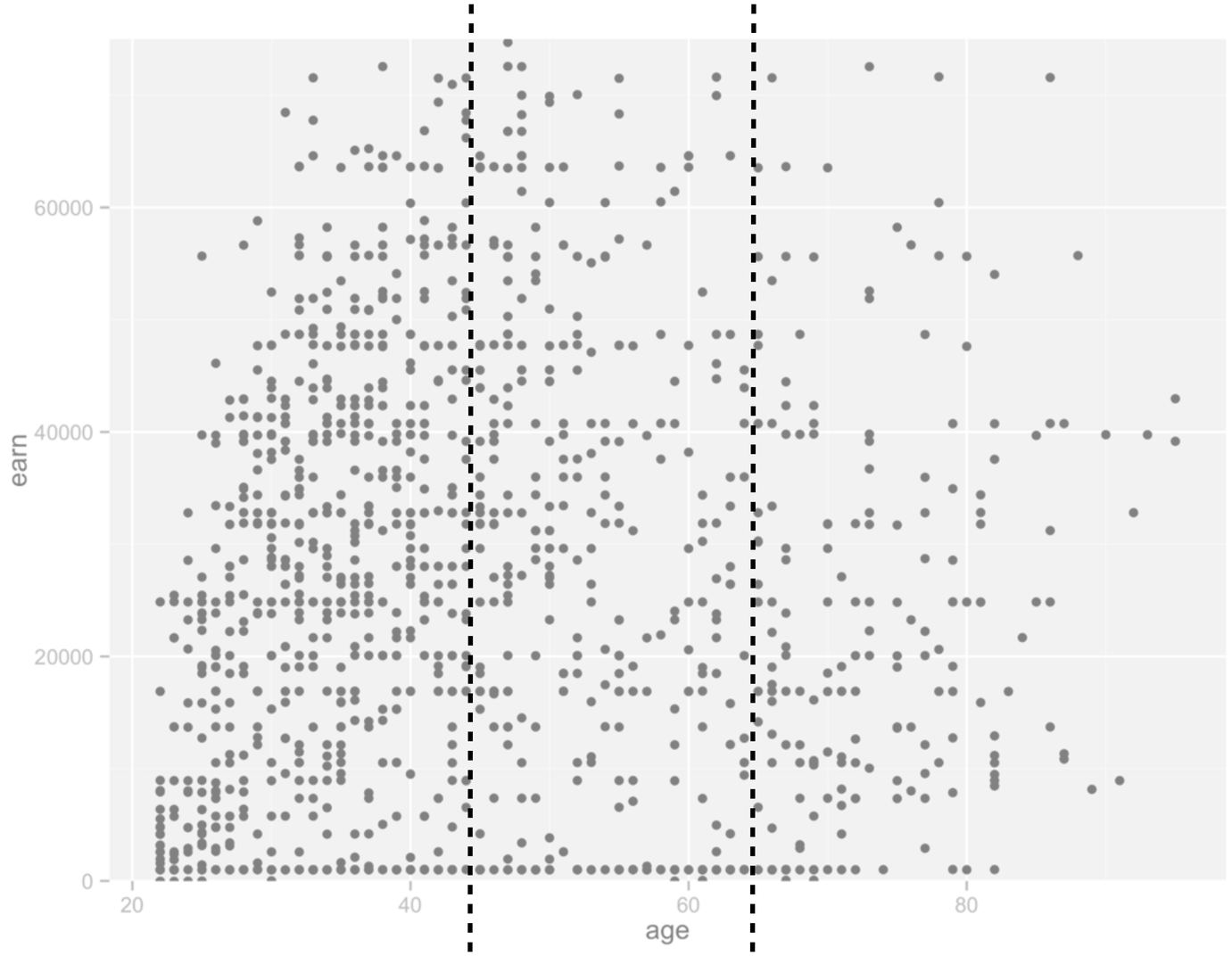
Polynomials tend to head off in extreme directions at the edges of the data

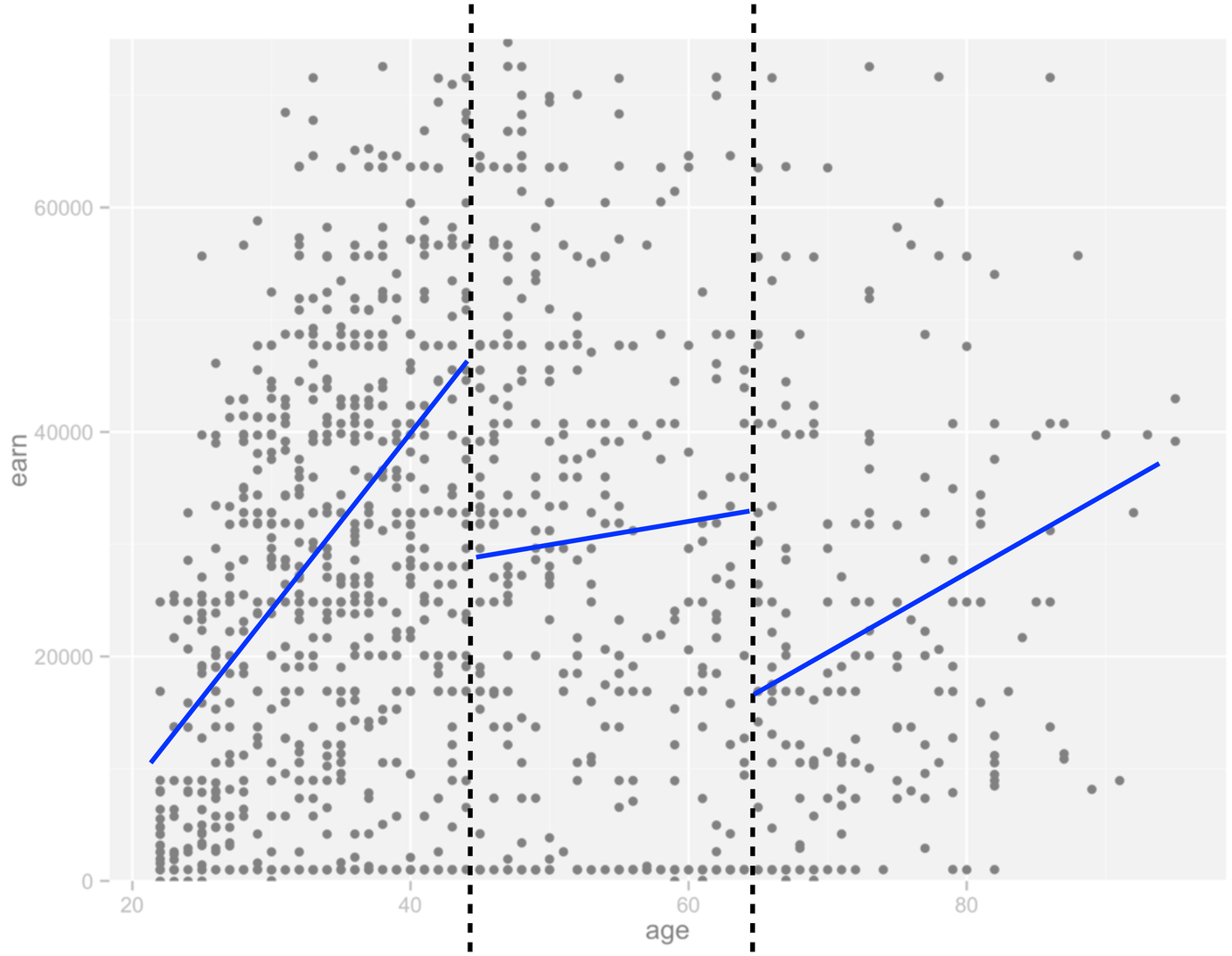


Splines

A different approach

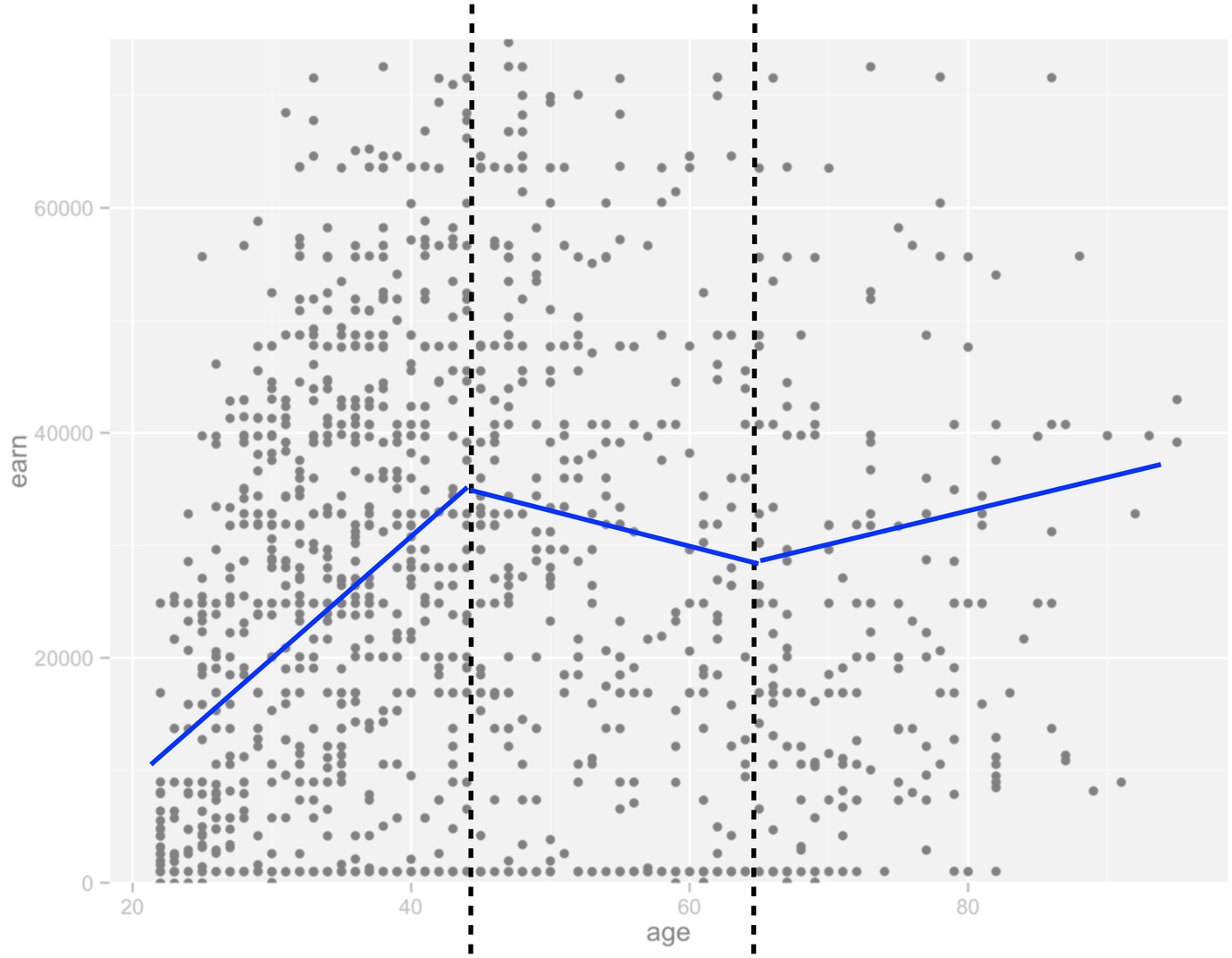






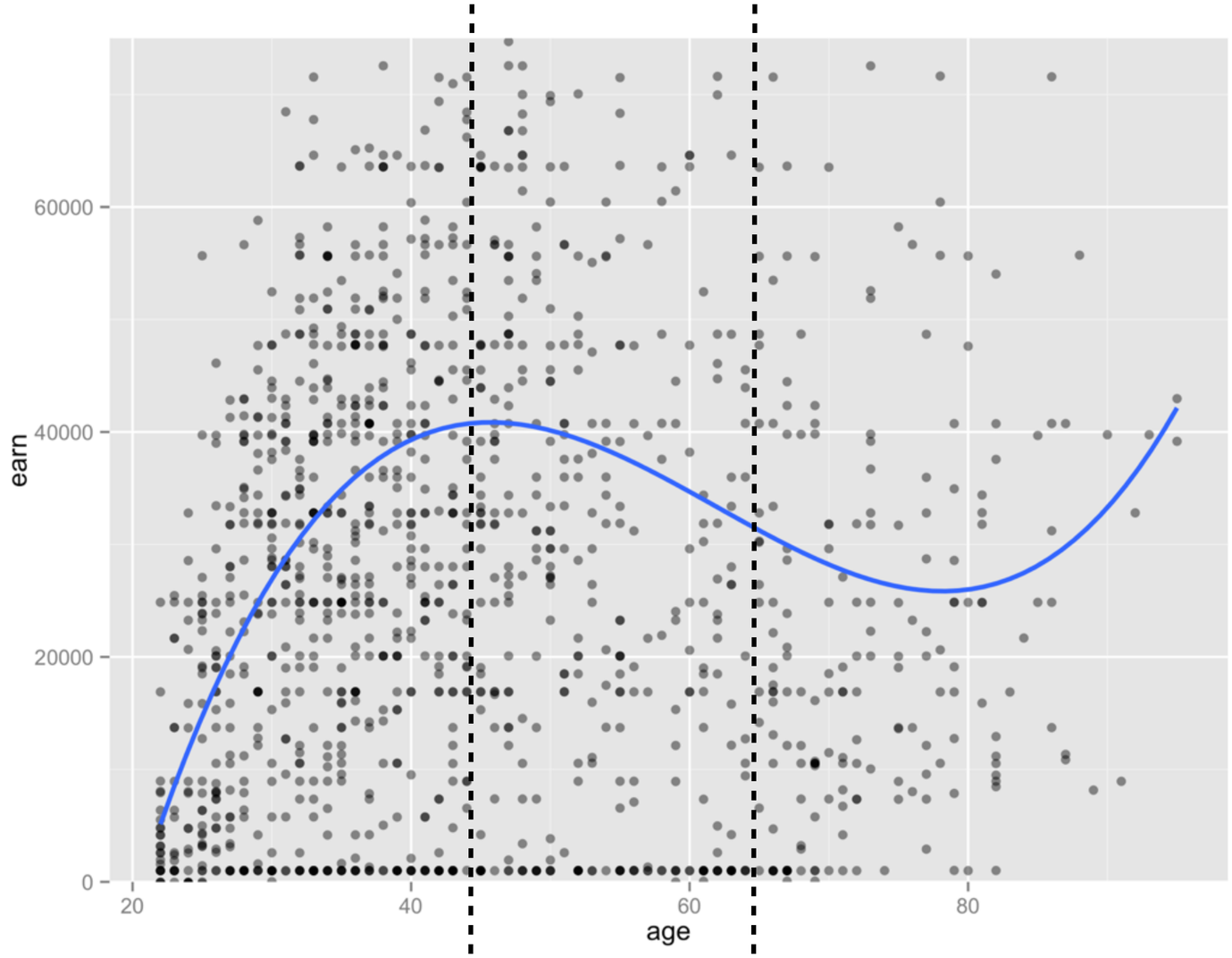
linear models separated at "knots"

Linear spline



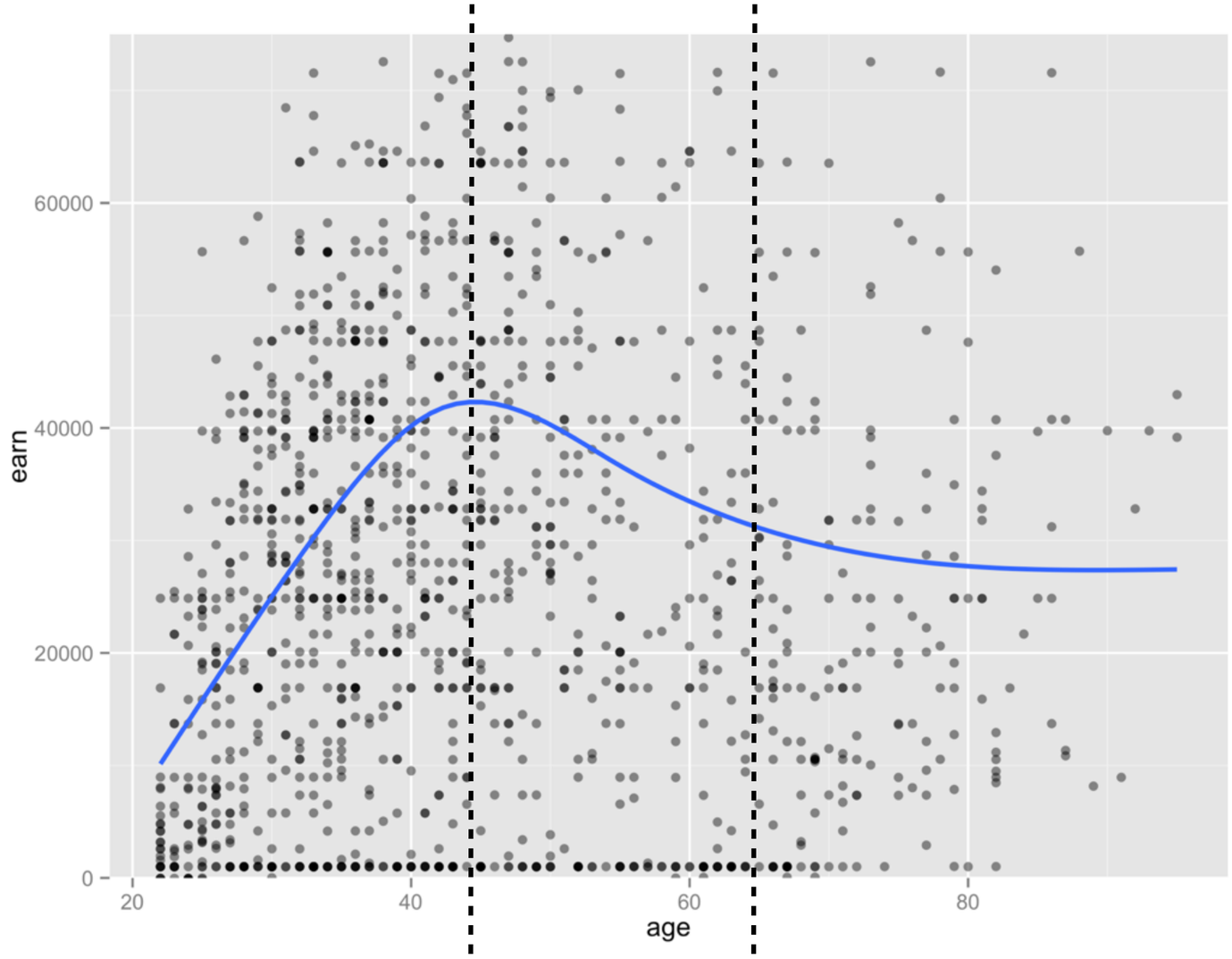
linear models that meet at knots

Cubic spline



cubic polynomial models that meet at knots

Natural spline



cubic models that meet at knots and are linear at edge of data

splines

Approach: Split data between knots. Model each section. Constrain behavior of at knots.

The `splines` library provides spline functions that work similar to `poly`

`bs(degree = 1)` - linear splines
`bs()` - cubic splines
`ns()` - natural splines

ns and bs syntax

```
library(splines)
mod <- lm(earn ~ ns(age, knots = c(40, 60)),
   data = wages)
```

ns for natural splines

x variable

where to place knots

ns and bs syntax

```
mod <- lm(earn ~ ns(age, df = 4),
   data = wages)
```

Alternatively, specify the desired degrees of freedom for your model. R will determine how many knots to use and will place them at evenly spaced quantiles within the data.

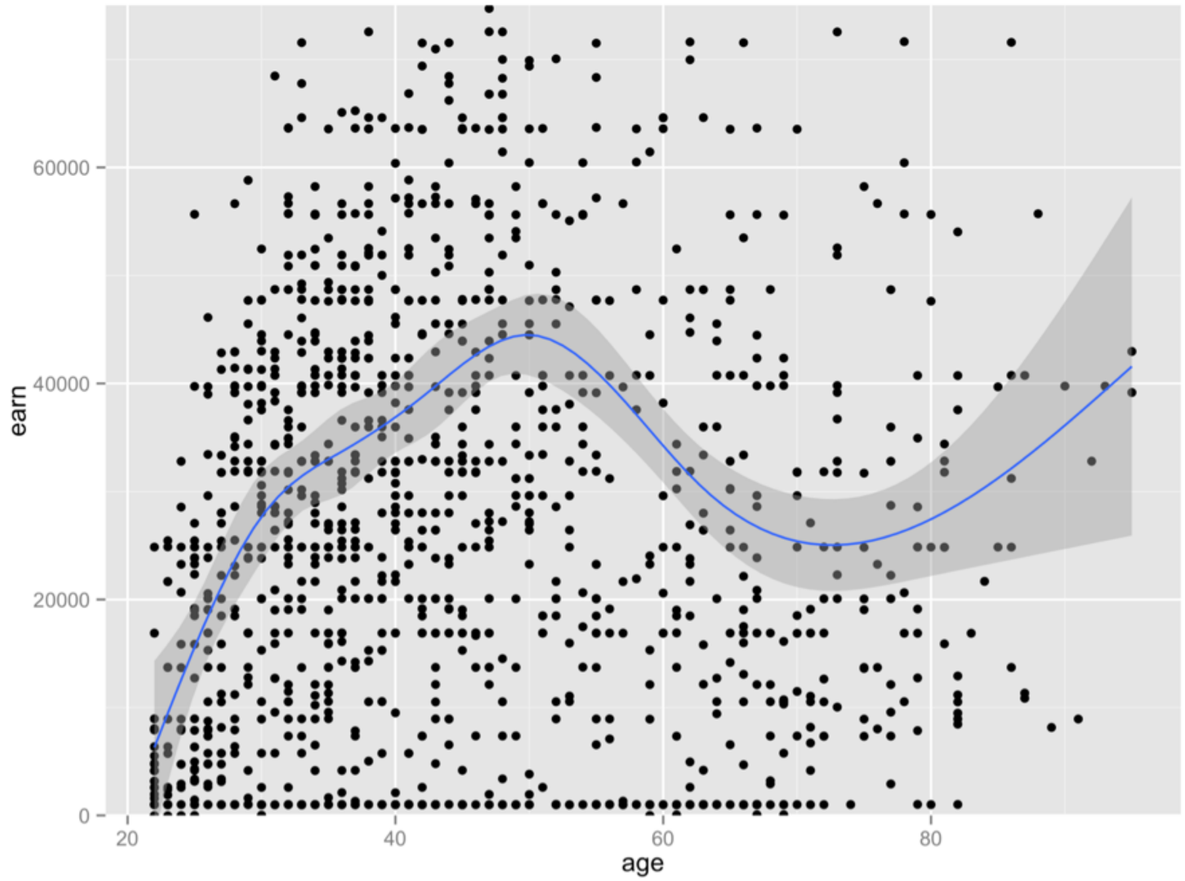
Your turn

Fit a spline regression to `earn` and `height` using a natural spline model with 6 degrees of freedom.

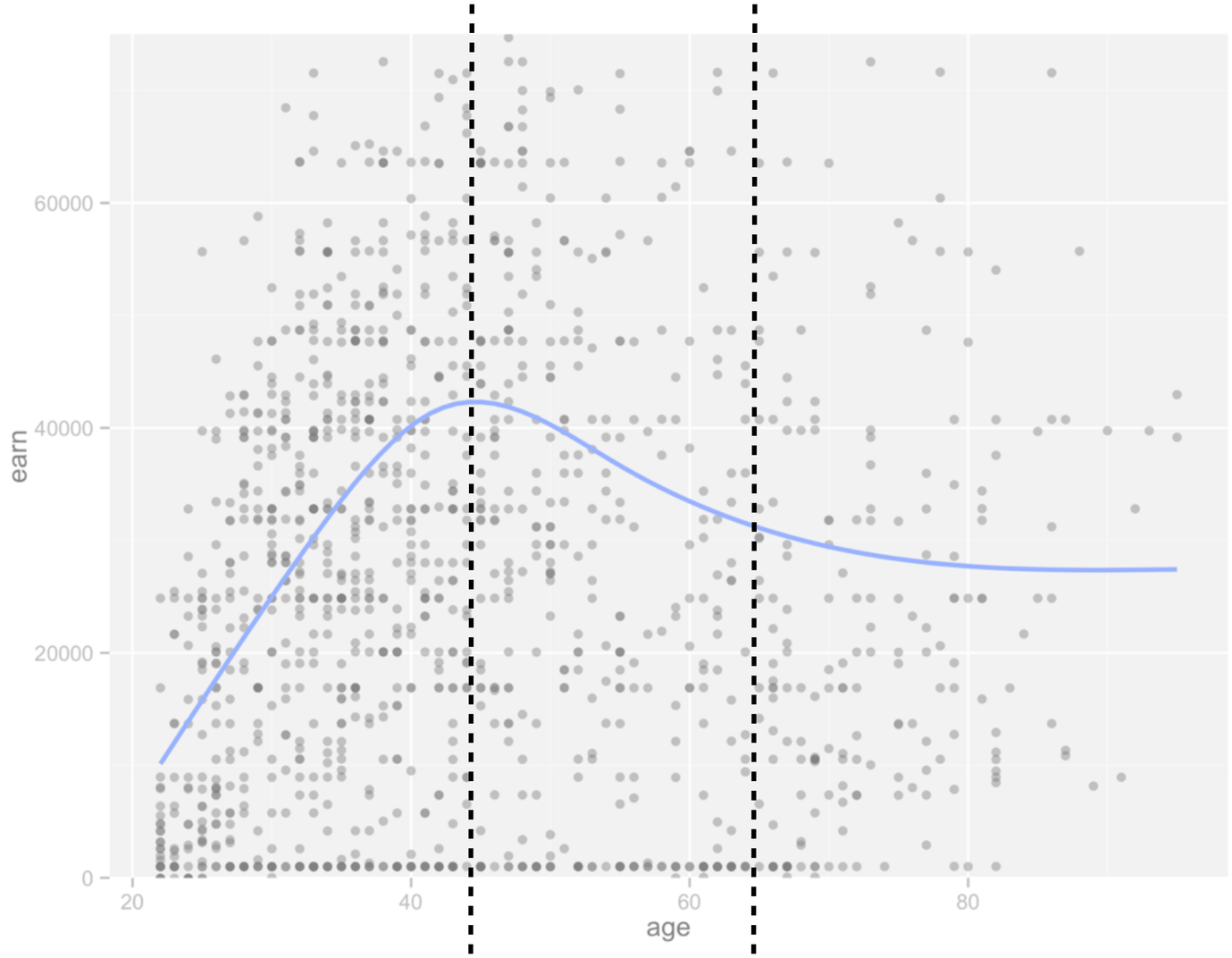
Challenge: visualize the predictions

```
lm(earn ~ ns(age, df = 6), data = wages)
```

```
qplot(age, earn, data = wages) +
  geom_smooth(method = lm, formula = y ~ ns(x, df = 6)) +
  coord_cartesian(ylim = c(0, 50000))
```

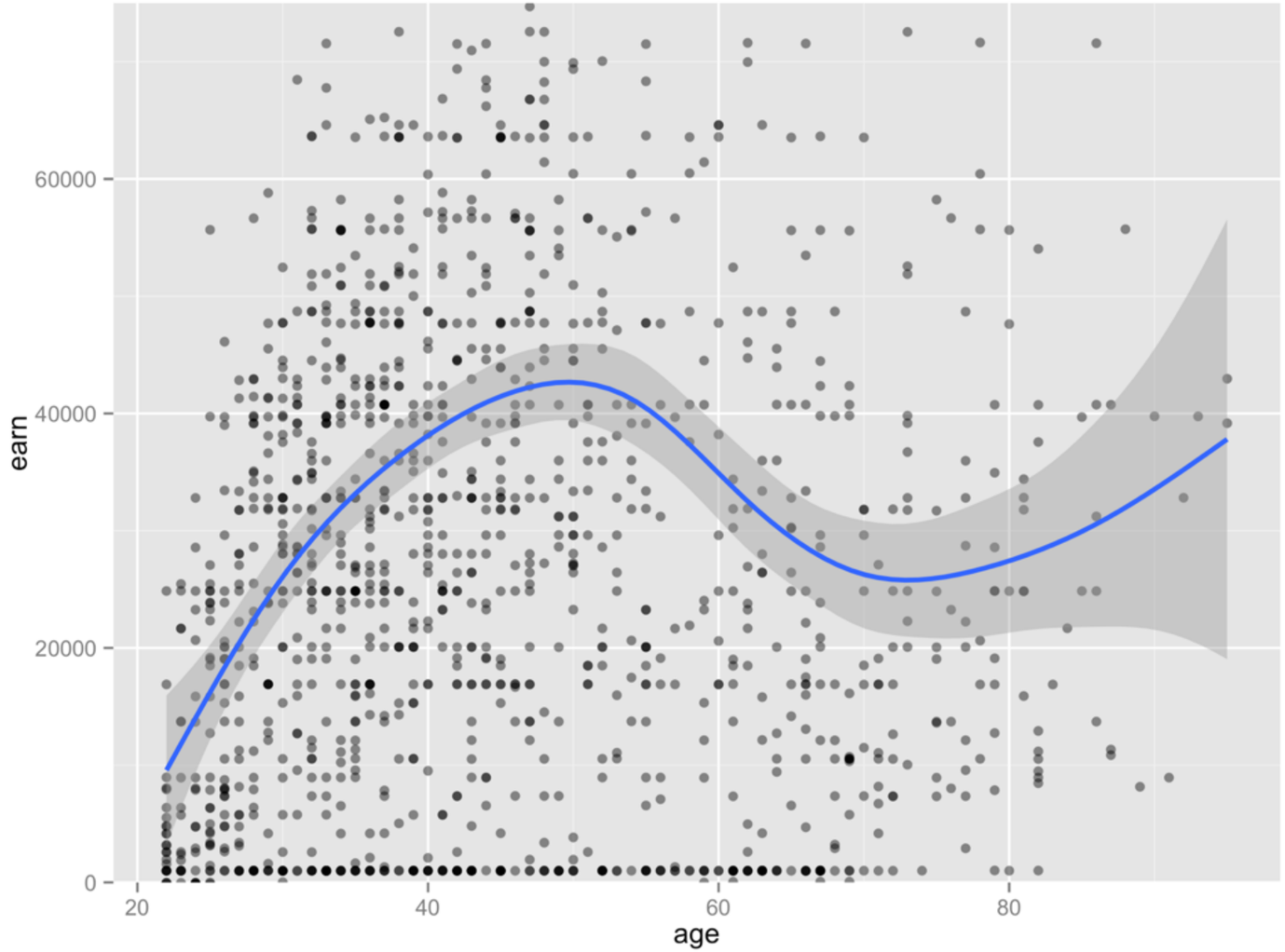


smoothing functions (gam)



Each spline method arbitrarily divided up the data

Smoothing function (smoothing spline)



No knots. Just fit the best smooth function.

smoothing splines

Approach: Use whichever smooth function fits x with the least error (e.g. minimize second equation below).

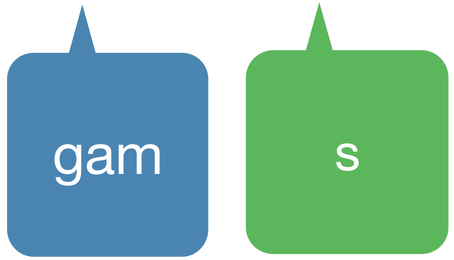
$$\sum_{i=1}^{n}(y - f(x_i))^2$$

$$\sum_{i=1}^{n}(y - f(x_i))^2 + \lambda \int f^{''}(t)^2 dt$$

generalized additive model syntax

in the mgcv package

```
library(mgcv)
mod <- gam(y ~ s(x), data = df)
```

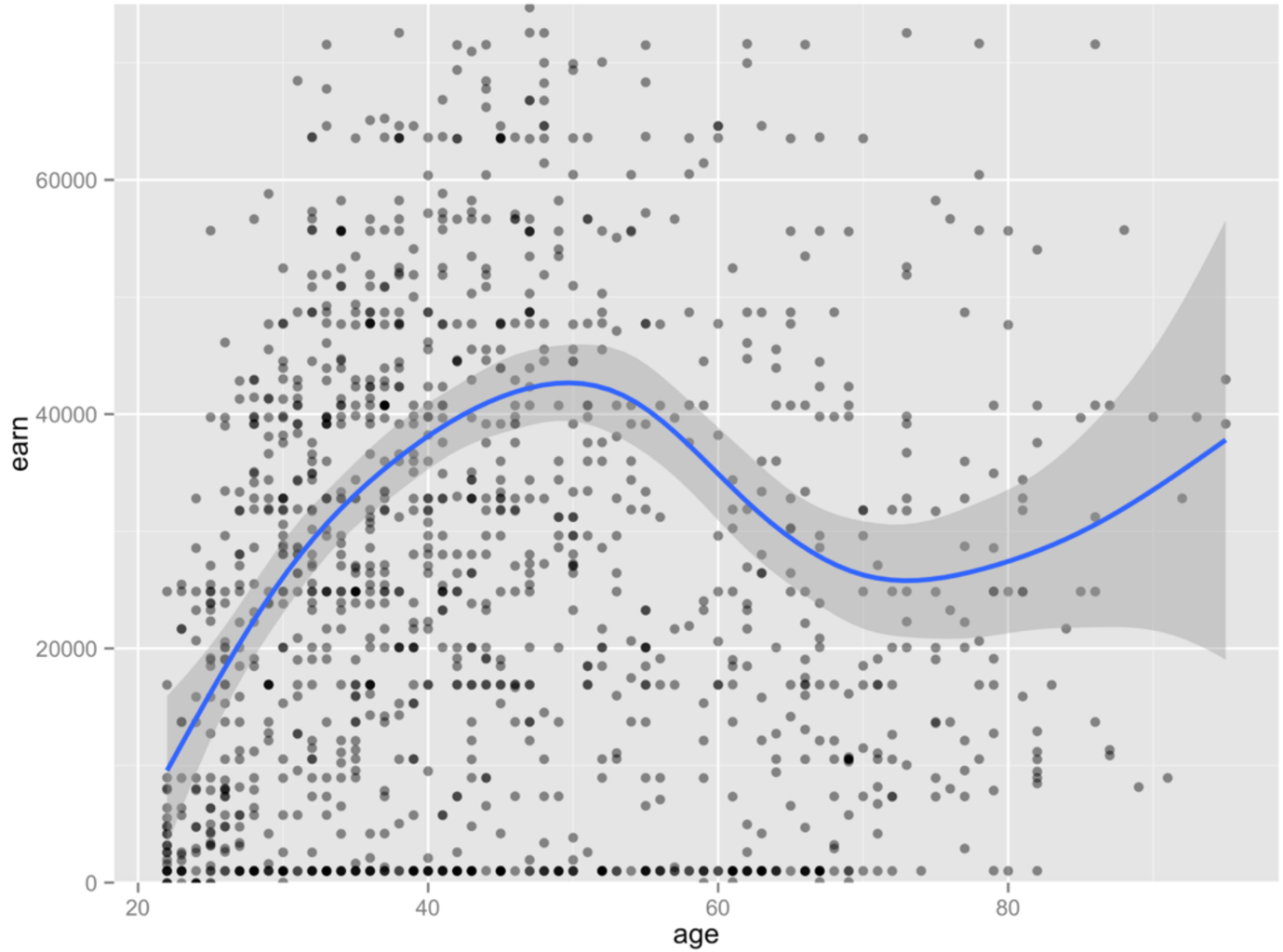


Your turn

Use `gam` to model `earn` with the best fitting smooth function of `height`.

```
library(mgcv)
gmod <- gam(earn ~ s(height), data = wages)
```

```
summary(gmod)
Family: gaussian
Link function: identity

Formula:
earn ~ s(height)

Parametric coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  32446.3      802.9   40.41   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Approximate significance of smooth terms:
            edf Ref.df     F p-value
s(height) 4.448  5.523 24.91  <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

R-sq.(adj) =  0.0902   Deviance explained = 9.31%
GCV = 8.9242e+08  Scale est. = 8.8889e+08  n = 1379
```




```
qplot(age, earn, data = wages) +
  geom_smooth(method = gam, formula = y ~ s(x)) +
  coord_cartesian(ylim = c(0, 50000))
```

multivariate cases

Linear z

```
gam(y ~ s(x) + z, data = df)
```

Smooth x and smooth z

```
gam(y ~ s(x) + s(z), data = df)
```

Smooth surface of x and z
(a smooth function that takes both x and z)

```
gam(y ~ s(x, z), data = df)
```

Non-linear relationships Recap

Non-linear relationships

`lm`	$y = \alpha + \beta x + \epsilon$	straight lines
`lm` + functions	$y = \alpha + \beta f(x) + \epsilon$	transformations
`lm` + `poly`	$y = \alpha + \beta_1 x + \beta_2 x^2 + \ldots + \epsilon$	polynomials
`lm` + `bs`, `ns`	$y = \alpha + \beta ns(x) + \epsilon$	splines
`gam` + `s`	$y = \alpha + \beta s(x) + \epsilon$	smooth functions

All of these still require your Y to be continuous (and normally distributed errors). What if this isn't so?

Logistic regression
(generalized linear models)

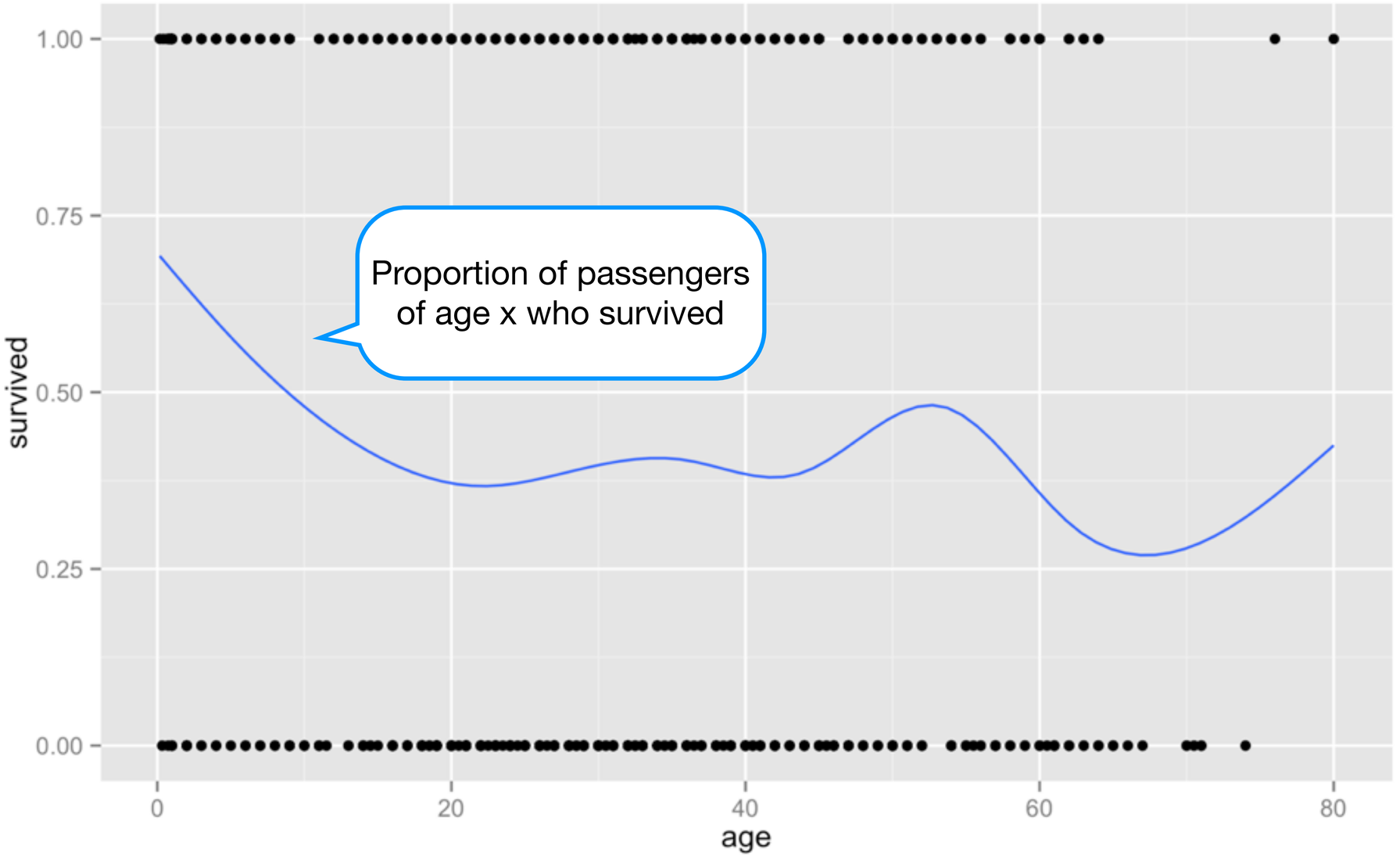
titanic3

Characteristics and fate of passengers on the Titanic.

```
titanic3 <- read.csv("data/titanic3.csv",
  stringsAsFactors = FALSE)
View(titanic3)
```

y is binomial:
survived or perished

Did age affect survival?




```
qplot(age, survived, data = titanic3) + geom_smooth(se = FALSE)
```

Notice

1. Y is a binomial variable (suggests logistic regression)
2. Want to predict odds of `y = survival` based on `x`

3. Relationship appears non-linear (suggests smooth functions)

logistic regression with gam

```
library(mgcv)
options(na.action = "na.exclude")
mod <- gam(y ~ s(x), data = df, family = binomial)
```

distribution of the y variable:
binary = binomial
count = poisson
normal = gaussian
...

Your turn

Run `options(na.action = "na.exclude")`

Then use `gam` to model `survived` with a smooth function of `age`.

```
options(na.action = "na.exclude")

gmod <- gam(survived ~ s(age), data = titanic3,
  family = binomial)

summary(gmod)
```

```
Family: binomial
Link function: logit

Formula:
survived ~ s(age)

Parametric coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.37337    0.06333  -5.896 3.72e-09 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Approximate significance of smooth terms:
         edf Ref.df Chi.sq p-value
s(age) 3.768   4.68  14.85 0.00907 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

R-sq.(adj) =  0.0139   Deviance explained = 1.27%
UBRE score = 0.34432  Scale est. = 1         n = 1046
```

Predictions

For logistic models, you can extract two types of predictions

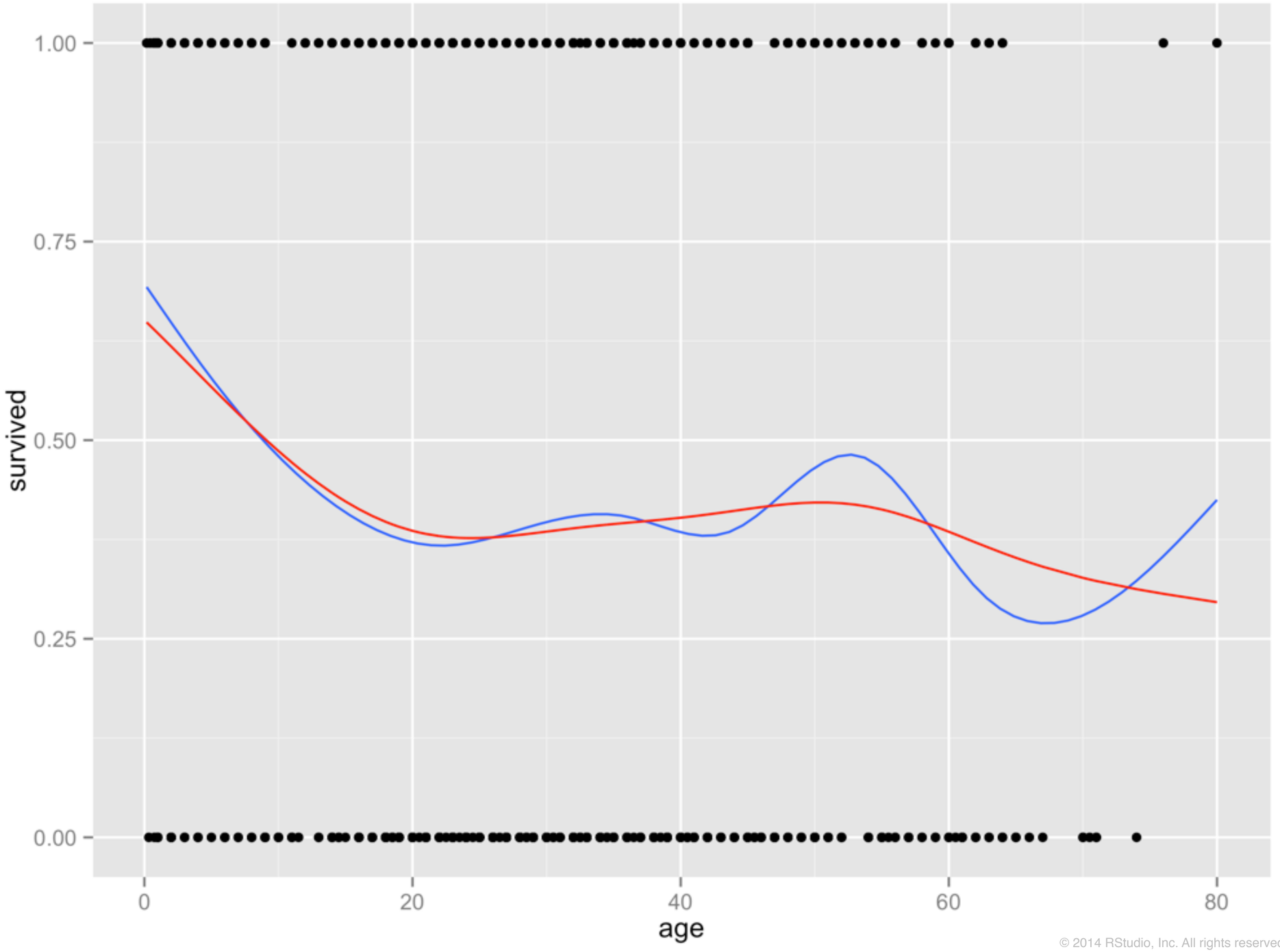
```
# predicted values of the link function
# (not usually what you want)
predict(gmod)
predict(gmod, type = "link")

# predicted values of y (here probabilities)
fitted(gmod)
predict(gmod, type = "response")
```

Your turn

Plot the fitted values of `gmod` against the `titanic3` data

```
qplot(age, survived, data = titanic3) +
  geom_smooth(se = FALSE) +
  geom_line(aes(y = _____________)), color = "red")
```

```
qplot(age, survived, data = titanic3) +
  geom_smooth(se = FALSE) +
  geom_line(aes(y = fitted(gmod)), color = "red")
```



na.exclude

By default, `gam` throws out data points that have an NA in the predictors. IMHO, **bad idea**.

As a result the output of `predict`, `fitted`, etc. will not align with original data frame. (The last graph would return an error).

You can prevent this behavior by running `options(na.action = "na.exclude")` before fitting any models

Residuals

Extract residuals with `resid`

```
# deviance residuals (default)
resid(g)
resid(gmod, type = "deviance")

# pearson residuals
resid(gmod, type = "pearson")
```

Non-linear relationships

`lm`	$y = \alpha + \beta x + \epsilon$	straight lines
`lm` + functions	$y = \alpha + \beta f(x) + \epsilon$	transformations
`lm` + `poly`	$y = \alpha + \beta_1 x + \beta_2 x^2 + \ldots + \epsilon$	polynomials
`lm` + `bs`, `ns`	$y = \alpha + \beta ns(x) + \epsilon$	splines
`gam` + `s`	$y = \alpha + \beta s(x) + \epsilon$	smooth functions
`gam` + `family`	$g(y) = \alpha + \beta x + \epsilon$	non-linear y

Learning resources

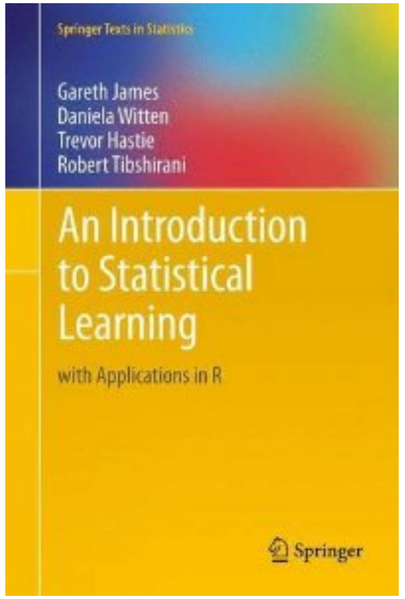
Introduction

"An Introduction to Statistical Learning" is the best beginners guide I have read.

Each chapter ends with a demonstration done in R code.

Free pdf at http://www-bcf.usc.edu/~gareth/ISL/

http://amzn.com/1461471370

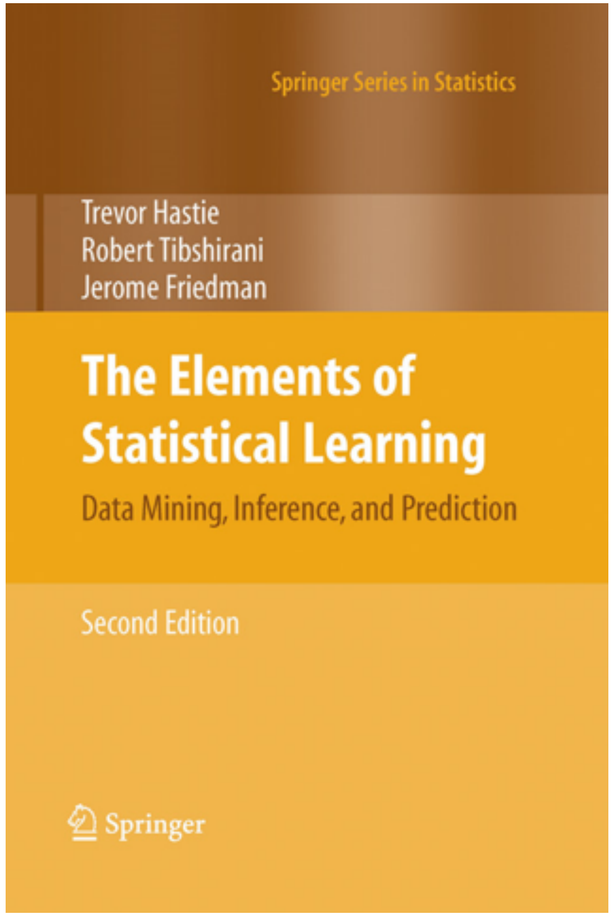


Theory

I recommend “The Elements of Statistical Learning”.

Free pdf copy at http://www-stat.stanford.edu/~tibs/ElemStatLearn/

http://amzn.com/0387848576



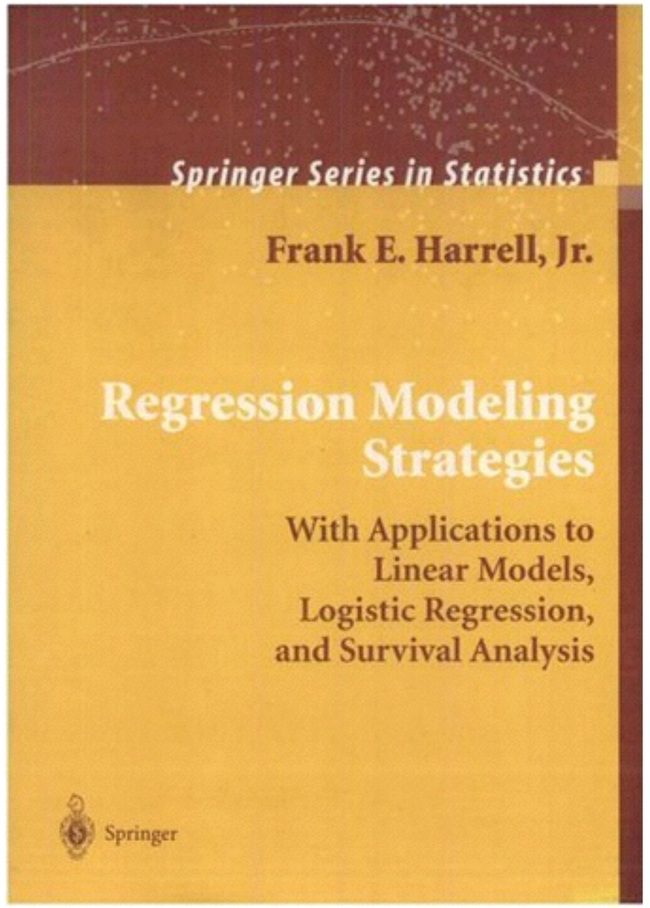
Application

Imputing missing values, calibrating models, cross validating.

Focuses on linear models, logistic regression and survival analysis.

Corresponds with the `rms` package*

http://amzn.com/0387952322



*rms is called Design in the book

Methods survey

Linear models

Linear models

Modern applied statistics with S

http://amzn.com/1441930086

linear models

```
ols
```

```
rms
```

Ordinary least squares regression

Regression Modeling Strategies

http://amzn.com/0387952322

Generalized linear model

glm

stats

Response not normal: Poisson (counts), Binomial (proportions), gamma, ...

Regression Modeling Strategies: With Applications to Linear Models, Logistic Regression, and Survival Analysis

http://amzn.com/1441929185

Extending the Linear Model with R: Generalized Linear, Mixed Effects and Nonparametric Regression Models

http://amzn.com/158488424X

Generalized linear model

Glm

rms

rms version of glm

Regression Modeling Strategies

http://amzn.com/0387952322

logistic models

```
lrm
```

rms

logistic regression

Regression Modeling Strategies

http://amzn.com/0387952322

Generalized additive model

gam

mgcv

Generalises glm to include smooth functions of the predictors

Generalized Additive Models: An Introduction with R

http://amzn.com/1584884746

Non-linear model

nlm

stats

Non-linear relationship between response and predictors

Nonlinear Regression Analysis and Its Applications

http://amzn.com/0470139005

Survival Analysis

survival curves

survfit

survival

Creates survival curves using Kaplan-Meier estimates or other methods

Modern applied statistics with S

http://amzn.com/1441930086

parametric survival models

survreg

survival

survival models using an assumed distribution such as weibull, exponential, etc.

Modern applied statistics with S

http://amzn.com/1441930086

parametric survival models

psm

rms

survival models using an assumed distribution such as weibull, exponential, etc.

Regression Modeling Strategies

http://amzn.com/0387952322

cox proportional hazard model

coxph

survival

Non parametric method for fitting survival models

Modern applied statistics with S

http://amzn.com/1441930086

cox proportional hazard model

cph

rms

Non parametric method for fitting survival models

Regression Modeling Strategies

http://amzn.com/0387952322

Big data

linear modeling for large data

biglm

biglm

Regression for data too large to fit
in memory

logistic modeling for large data

bigglm

biglm

Logistic regression for data too
large to fit in memory

Robust regression

Robust linear model

rlm

MASS

Heavy-tailed errors
Less sensitive to outliers

Modern applied statistics with S

http://amzn.com/1441930086

Principal Components Regression

pcr

pls

Regresses on the principal components of X_i.
Avoids multicollinearity issues.

Partial Least Squares Regression

plsr

pls

Regresses on projections of Y and X_i.
Avoids multicollinearity issues.

Feature selection

ridge regression

lm.ridge

MASS

Constrains size of coefficients instead of limiting number of explanatory variables. Useful for p >> n.

Lasso and least angle regression

lars

lars

Linear regression penalized by an L2 penalty. Identifies useful variables. Avoids over fitting when p >> n.

The Elements of Statistical Learning

http://www-stat.stanford.edu/~tibs/ElemStatLearn/

elastic net

enet

elasticnet

A generalized form of ridge regression and lasso.

The Elements of Statistical Learning

http://www-stat.stanford.edu/~tibs/ElemStatLearn/

glmnet

glmnet

glmnet

Generalized linear models with feature selection (penalized regression)

Regularization Paths for Generalized Linear Models via Coordinate Descent

http://www.stanford.edu/~hastie/Papers/glmnet.pdf

Hierarchical modelling

Linear mixed effects model

`lme`

`nlme`

Multiple sources of error.
A type of hierarchical model.

Being reimplemented in `lme4::lmer`. Faster, and handles crossed random effects, but not complete.

Mixed-Effects Models in S and S-PLUS

http://amzn.com/1441903178

Data Analysis Using Regression and Multilevel/ Hierarchical Models

http://amzn.com/052168689X

http://lme4.r-forge.r-project.org/book/

Machine learning
(and Classification)

Trees

cart

rpart

Non-linear model based on
recursively breaking up the space

Neural networks

nnet

nnet

Neural networks.
Ensembles of linear models.

K nearest neighbors

knn

class

Simple, all purpose classification algorithm

Random forests

randomForest

randomForest

Ensemble of trees

Support vector machines

ksvm

kernlab

Machine learning technique that selects small set of **support vectors** used for classification/ regression

See http://www.jstatsoft.org/v15/i09/paper for comparison of other implementations