Stat405

Modelling

Hadley Wickham

1. New dataset
2. 2 continuous predictors
3. 1 continuous + 1 categorical predictor
4. 3+ predictions
5. Model building
6. Non-linearity

Goals

- Start to fit more complex models (> 1 variable!)
- Develop the visualization techniques that will enable you to explore even more complicated models

Residential energy consumption

REC

http://www.eia.gov/consumption/residential/

Survey conducted every 2-3 years (1978-2009). We're going to look at 2005 data.

4382 observations, 602 variables

For this class, reduced to 21 variables.

http://www.flickr.com/photos/39747297@N05/5229733647/

Variable	Description
`division`	US census division
`typehuq`	Type of house
`hd65, cd65`	Heating/cooling degree days
`ncombath`	Number of bathrooms
`yearmade, yearmade2`	Year made, as factor and range midpoints
`numfrig`	Number of fridges
`wheatsiz`	Water heater size
`moneypy`	Income per year
`totsqft, tothsqft`	Total and total heated sq ft
`cost, cool_cost, heat_cost`	Energy spend (total, cooling, heating)

```
library(ggplot2)
library(plyr)
source("helpers.r")
rec <- readRDS("rec.rds")
```

2 continuous

Your turn

We’re going to try and predict `heat_cost` with `hd65` and `totsqft`. Is this likely to succeed? Use graphics to explore.

```r
# Always good to start with some EDA
qplot(hd65, heat_cost, data = rec) + geom_smooth()

# Eliminate very large costs to make it easier to see
# what's going on
rec$heat_cost[rec$heat_cost > 2750] <- NA
qplot(hd65, heat_cost, data = rec) + geom_smooth()
qplot(totsqft, heat_cost, data = rec) +
  geom_smooth()
qplot(hd65, totsqft, data = rec, colour = heat_cost)
```

2D Prediction grid

mod_grid can take multiple inputs. It produces the combination of all of them.

```
> head(mod_grid(rec, x = 1:5, y = 1:5))
```

	x	y
1	1	1
2	2	1
3	3	1
4	4	1
5	5	1
6	1	2

```
mod_heat <- lm(heat_cost ~ hd65 + totsqft,
   data = rec)

grid <- mod_grid(rec,
   hd65 = seq_range(hd65, 20),
   totsqft = seq_range(totsqft, 20))
grid$heat_cost <- predict(mod_heat, grid)
```

Visualization

We need a new way of visualizing the output when we have a 2d surface:

```
qplot(hd65, totsqft, data = grid,
  fill = heat_cost, geom = "tile")
```

But this is perceptually hard: often better to select a few points or draw lines.

```
qplot(hd65, totsqft, data = grid, color = heat_cost)
qplot(hd65, totsqft, data = grid, fill = heat_cost,
  geom = "tile")

qplot(hd65, heat_cost, data = grid, color = totsqft,
  group = totsqft, geom = "line")
qplot(totsqft, heat_cost, data = grid, color = hd65,
  group = hd65, geom = "line")
```

How do you interpret the relationship between heat_cost, hd65 and totsqft?

Your turn

Instead of adding the variables independently with +, we can add interactions (dependence) with:

```
mod_heat2 <- lm(heat_cost ~ hd65 *
  totsqft, data = rec)
```

What impact does this have on the model?

```
mod_heat2 <- lm(heat_cost ~ hd65 * totsqft, data = rec)
grid <- mod_grid(rec,
  hd65 = seq_range(hd65, 20),
  totsqft = seq_range(totsqft, 5))
grid$heat_cost <- predict(mod_heat2, grid)

qplot(hd65, heat_cost, data = grid, colour = totsqft,
  group = totsqft, geom = "line")

rmse(mod_heat, rec)
rmse(mod_heat2, rec)

rd(mod_heat, rec)
rd(mod_heat2, rec)
```

Interactions

Using * to add additional variables creates interactions: the response varies with the **combination** of the two values, you can no longer think about them independently.

(There are a few notes on interaction `2-interaction.r`, but we won't cover them in detail since they are only really important for linear models)

Model comparison

Theory, and statistical “significance” are available for some models. Generally don't care about this if all you're interested in is prediction.

Judge using your subject knowledge: does an increase in complexity significantly improve predictive ability? (often not the case)

Also beware the difference between statistical and practical significance

```
anova(mod_heat, mod_heat2)
Analysis of Variance Table

# Model 1: heat_cost ~ hd65 + totsqft
# Model 2: heat_cost ~ hd65 * totsqft
#   Res.Df       RSS Df Sum of Sq      F    Pr(>F)
# 1   4369 654754767
# 2   4368 645181988  1   9572779 64.809 1.054e-15 ***
```

!!!

```
# Always good idea to check where the data is
# be careful of predictions
# that are far away from the data!
grid <- mod_grid(rec,
  hd65 = seq_range(hd65, 20),
  totsqft = seq_range(totsqft, 20))
grid$heat_cost <- predict(mod_heat2, grid)

qplot(hd65, totsqft, data = grid, fill = heat_cost,
  geom = "tile") +
  geom_point(data = rec)
qplot(hd65, totsqft, data = grid, fill = heat_cost,
  geom = "tile") +
  geom_point(data = rec, aes(colour = heat_cost))
```

1 continuous +
1 categorical

Your turn

Explore the relationship between `cost`, `totsqft` and `division`, first graphically and then with a model.

Does an interaction help?

```
mod <- lm(cost ~ totsqft + division, data = rec)
grid <- mod_grid(rec,
  totsqft = seq_range(totsqft, 50),
  division = unique(division))
grid$cost <- predict(mod, grid)

qplot(totsqft, cost, data = grid, colour = division, geom = "line")
```

```
mod2 <- lm(cost ~ totsqft * division, data = rec)
grid$cost <- predict(mod2, grid)
qplot(totsqft, cost, data = grid, colour = division, geom = "line")

# Very marginal impact
rd(mod, rec)
rd(mod2, rec)
```

Interaction

In this dataset, there's a really nice interaction between `aircond`, `cd65` and `cool_cost`.

Before fitting a model think about why this might be the case.

```
rec$aircond <- factor(rec$aircond)
qplot(cd65, cool_cost, data = rec, colour = aircond)
rec$cool_cost[rec$cool_cost > 2000] <- NA
qplot(cd65, cool_cost, data = rec, colour = aircond)

grid <- mod_grid(rec,
  cd65 = seq_range(cd65, 50),
  aircond = unique(aircond))

mod1 <- lm(cool_cost ~ cd65 + aircond, data = rec)
mod2 <- lm(cool_cost ~ cd65 * aircond, data = rec)

grid$cool_cost1 <- predict(mod1, grid)
grid$cool_cost2 <- predict(mod2, grid)

qplot(cd65, cool_cost1, data = grid, geom = "line", group = aircond)
qplot(cd65, cool_cost2, data = grid, geom = "line", group = aircond)

# mod2 is _much_ better!
rd(mod1, rec)
rd(mod2, rec)
```

3+

predictors

More complex models

Increasingly hard to understand/visualize.

As predictive ability ↑, understanding ↓. The less you understand the more you should worry about applicability to new data.

Often you'll need to create unique visualizations for your specific combination of variables.

General techniques

Facetting.

Explore independent components independently.

Often easier to focus on **residuals**. Do the same exploratory visualizations you did for the raw data. What signal remains?

Your turn

Imagine we've fit this model:

```
mod3 <- lm(cool_cost ~ cd65 * aircond
   * totsqft, data = rec)
```

How could we visualize the results? What special property of this data could we use to simplify the model?

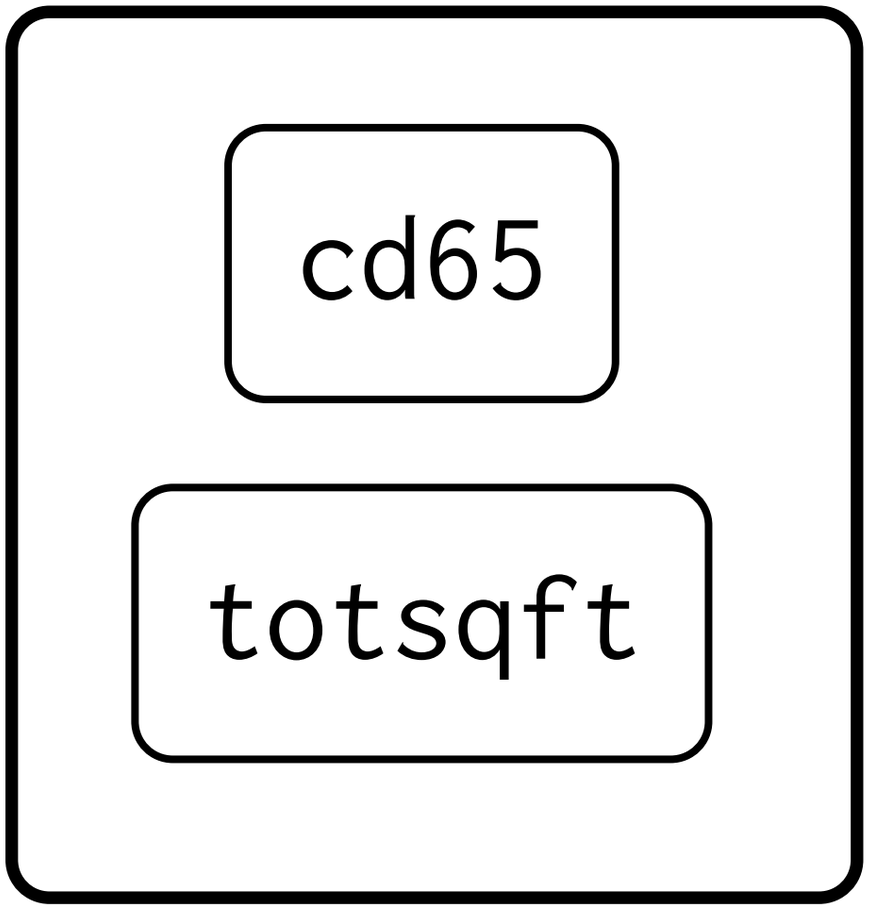
```
grid <- mod_grid(rec,
  aircond = unique(aircond),
  cd65 = seq_range(cd65, 20),
  totsqft = seq_range(cd65, 20))
grid$cool_cost <- predict(mod3, grid)

# All the variables interact, so we must examine
# them together
qplot(cd65, totsqft, data = grid, geom = "tile",
  fill = cool_cost) + facet_wrap(~ aircond)
```

```
# We can simplify this model by looking at a subset
of the data
ac <- subset(rec, aircond == 1)
mod4 <- lm(cool_cost ~ cd65 * totsqft, data = ac)
```

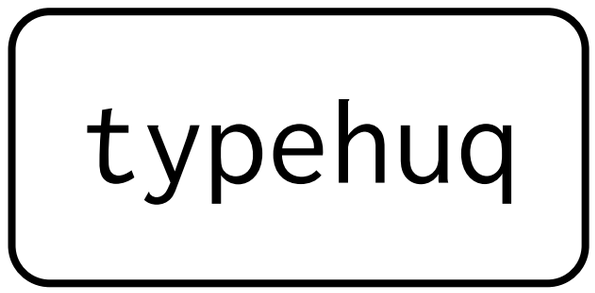
Mixed interactions

```
# New model
mod5 <- lm(cool_cost ~ cd65 * totsqft +
   typehuq + division, data = ac)
```

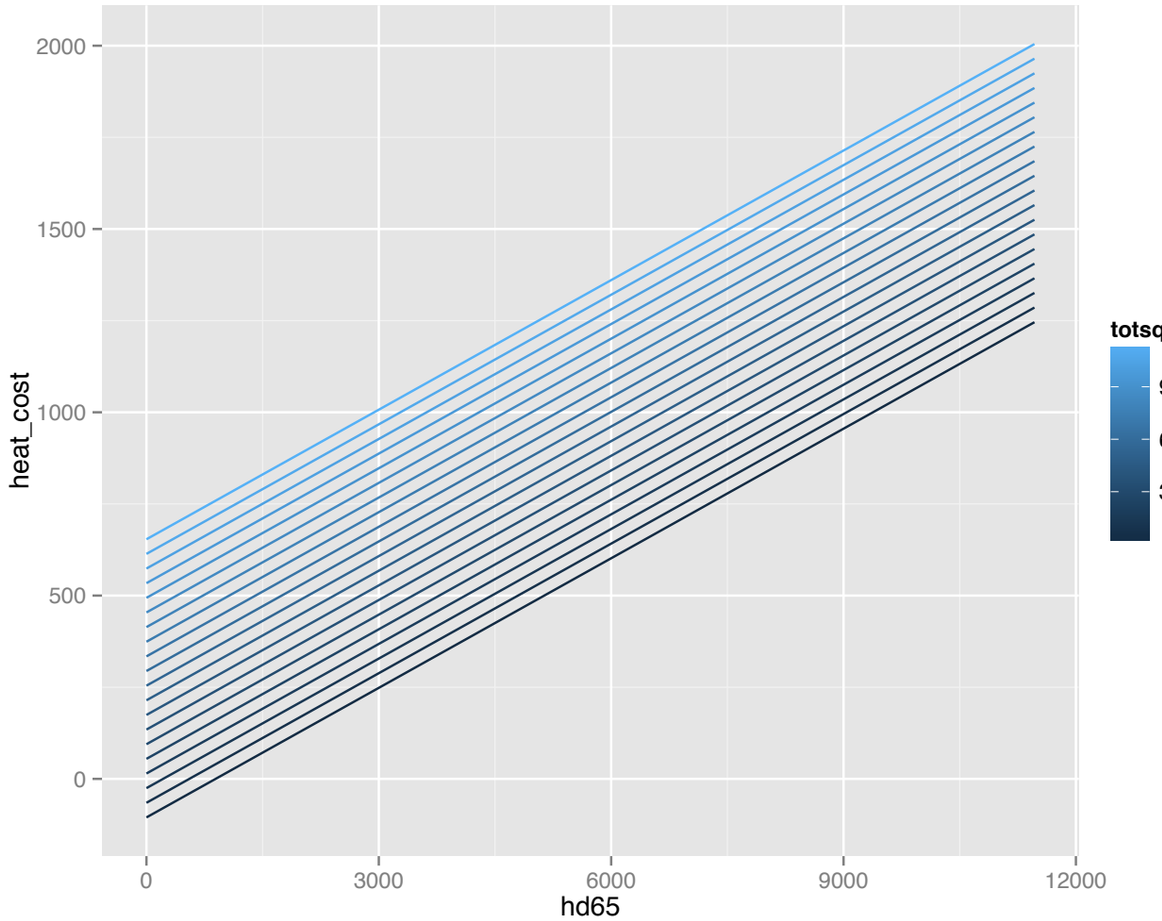
```
cool_cost ~ cd65 * totsqft + typehuq + division
```

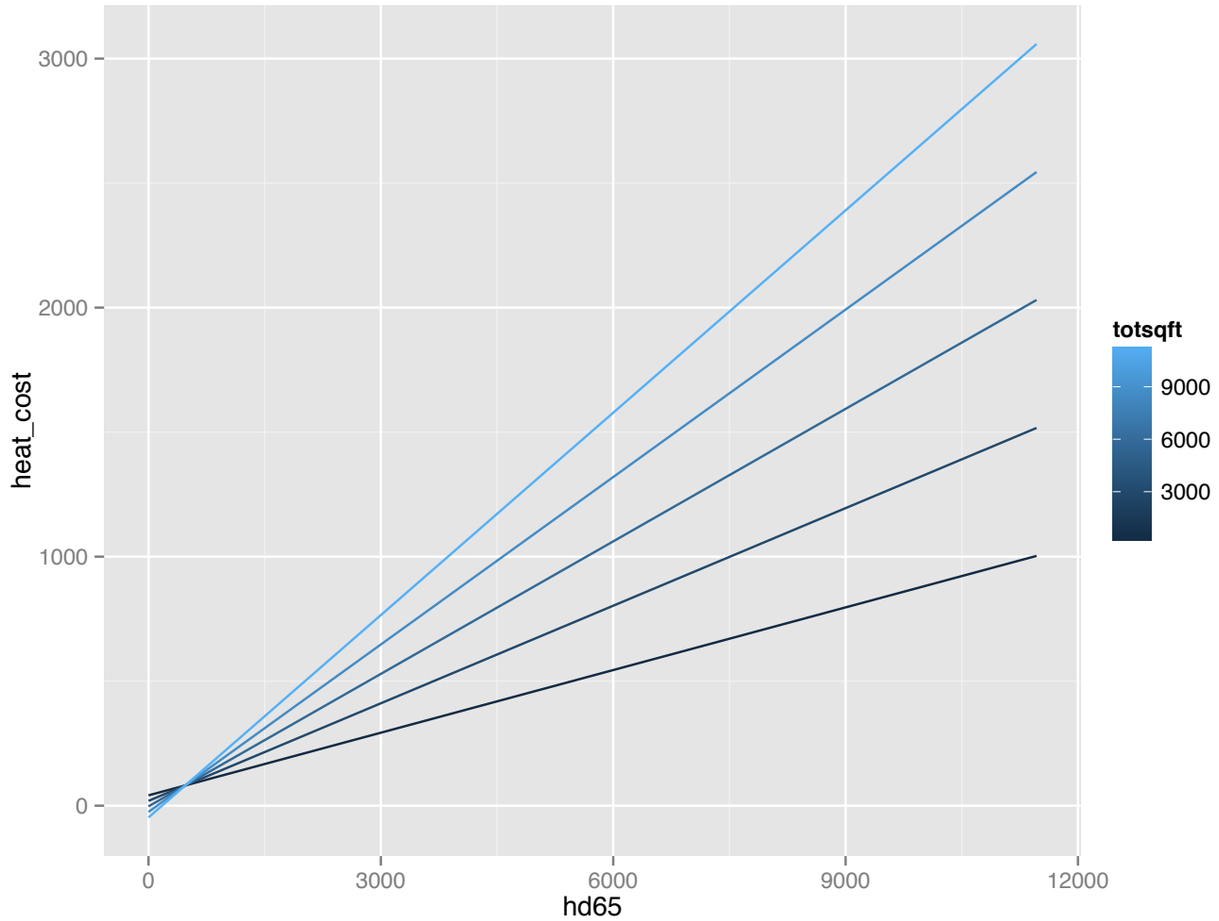


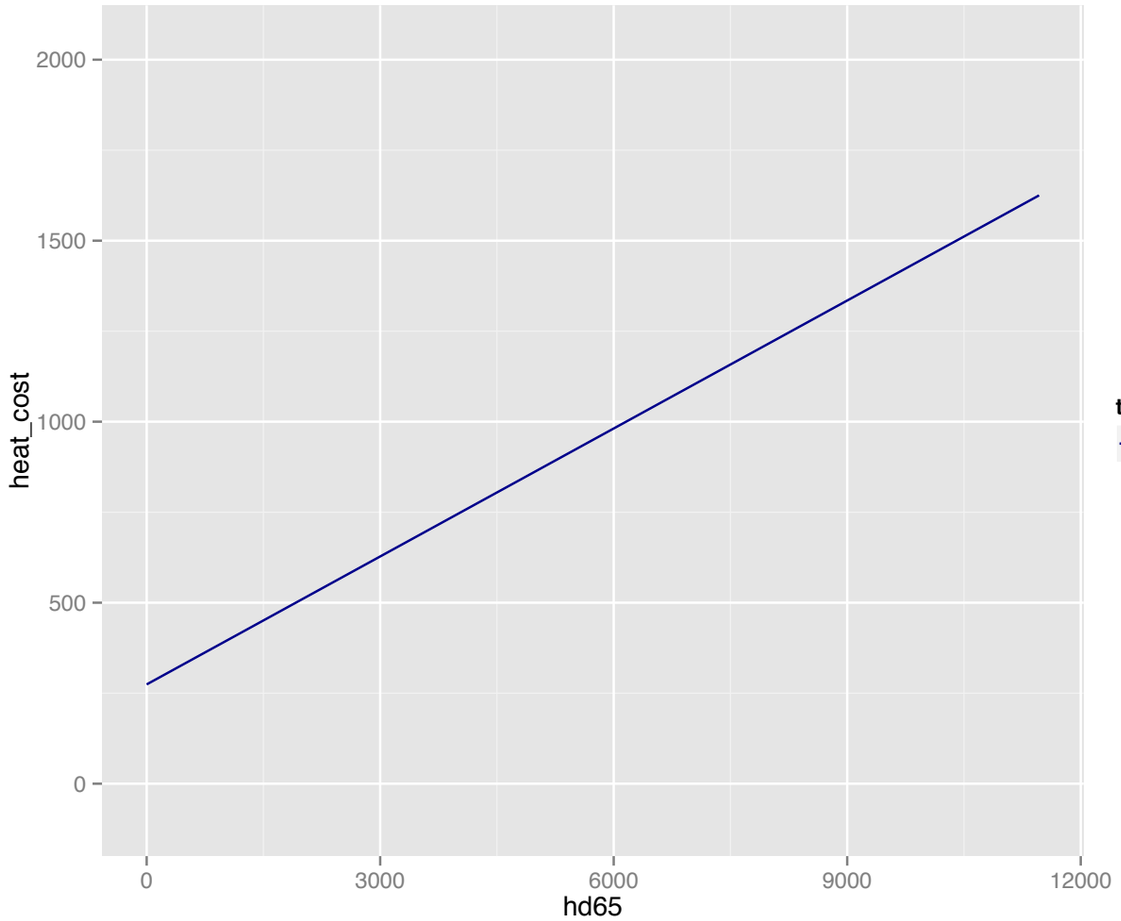
Interaction means that we can’t understand the impact of these variables on the model in isolation

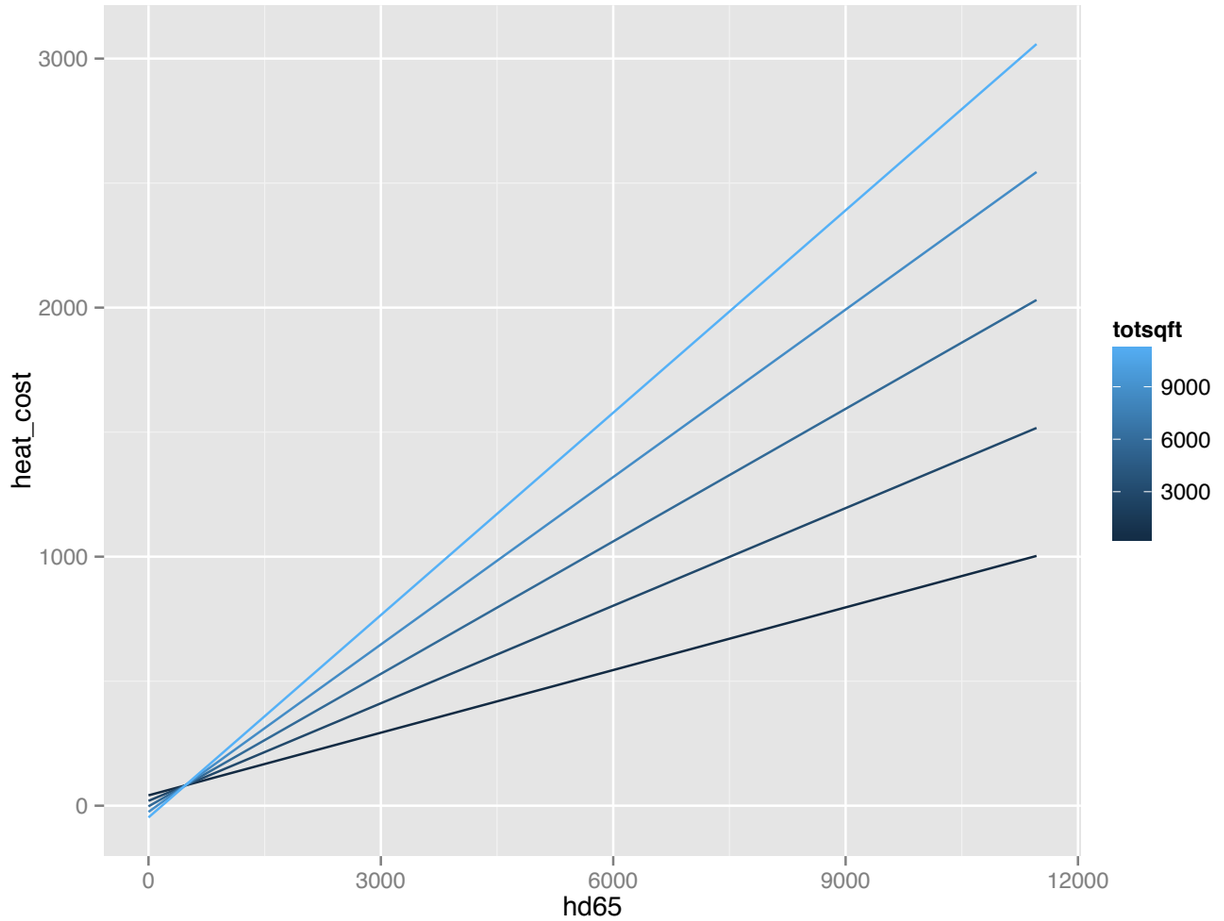


Lack of interaction means that we can understand the impact of these variables on the model individually


heat_cost
2000
1500
1000
500
0
0
3000
6000
9000
12000
hd65
totsqft
9000
6000
3000


heat_cost
3000
2000
1000
0
0
3000
6000
9000
12000
hd65
totsqft
9000
6000
3000


heat_cost
2000
1500
1000
500
0
0
3000
6000
9000
12000
hd65
totsqft
6000


heat_cost
3000
2000
1000
0
0
3000
6000
9000
12000
hd65
totsqft
9000
6000
3000

Prediction grids

For this problem, we need 3 grids:

`cd65` & `totsqft`

`typehuq`

`division`.

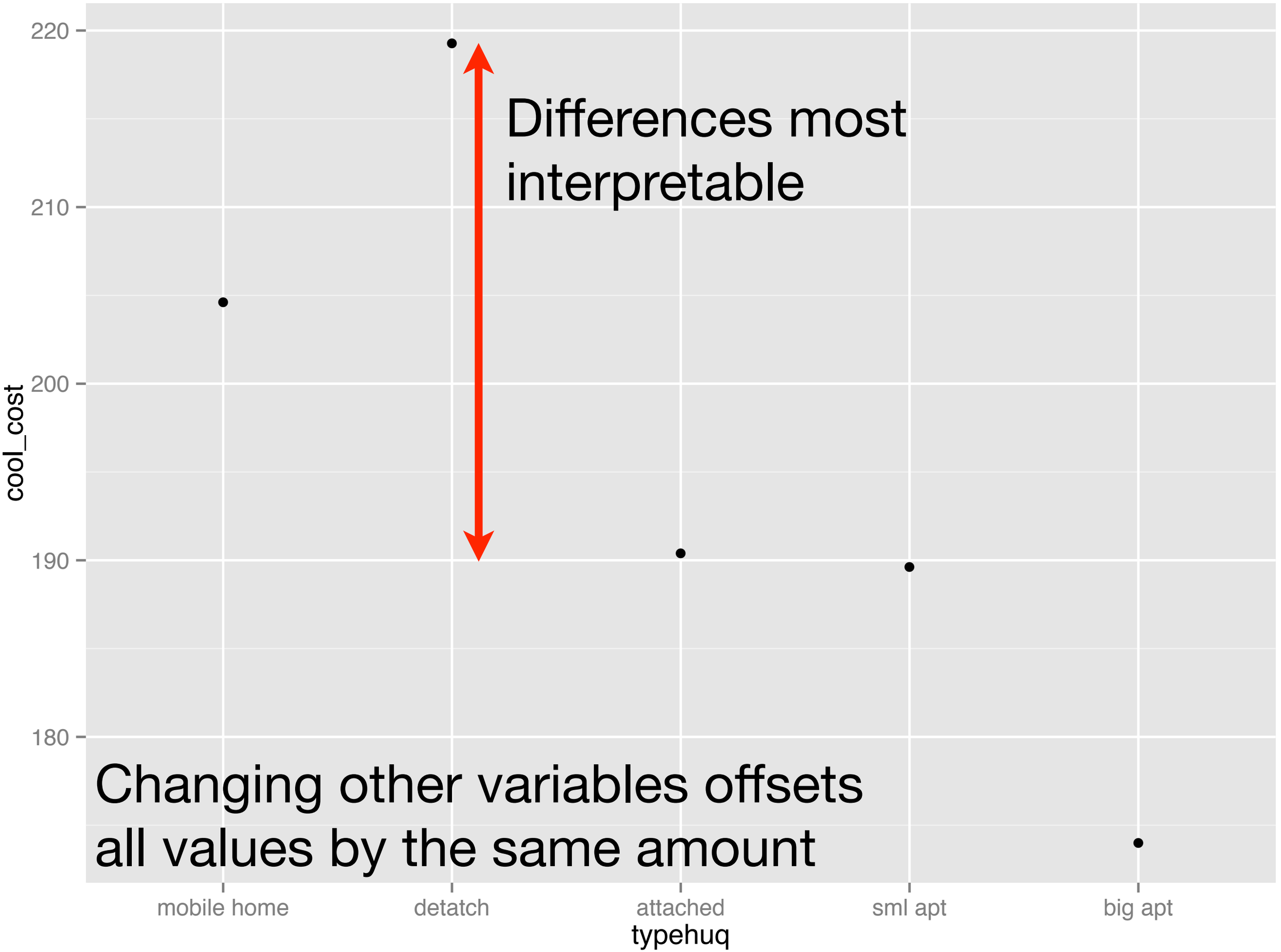
Prediction grids

For the variables that don't vary within the grid, we use a "typical" value: the mean for a continuous variable, or the mode for a categorical variable.

These grids + predictions are sometimes know as least squares means or population marginal means.

```
grid1 <- mod_grid(ac,
  cd65 = seq_range(cd65, 20),
  totsqft = seq_range(cd65, 20),
  typehuq = factor("detatch"),
  division = factor("pacific"))
grid1$cool_cost <- predict(mod5, grid1)
qplot(cd65, totsqft, data = grid1, geom = "tile",
  fill = cool_cost)
```

```
grid2 <- mod_grid(ac,
  cd65 = mean(cd65),
  totsqft = mean(cd65),
  typehuq = unique(typehuq),
  division = factor("pacific"))
grid2$cool_cost <- predict(mod5, grid2)
qplot(typehuq, cool_cost, data = grid2)
```


220
210
200
190
180
cool_cost
Differences most interpretable
Changing other variables offsets all values by the same amount
mobile home
detatch
attached
sml apt
big apt
typehuq

Your turn

Create the final grid and explore the impact of `division`.

Residuals

For more complicated models, looking at residuals is also very helpful.

Using residuals makes it easier to see new unknown patterns, once you've removed the known patterns.

Your turn

Using the residuals from mod5, explore the remaining variables. Would it be a good idea to include the `moneypy` or `temphomeac` variables?

Model building

Model building

How do we select the best combination of variables for a model?

1. Subject matter theory
2. data driven
3. stepwise regression
4. feature selection

Support in R is shaky at the moment.

Subject matter theory

Rely on current scientific knowledge in the area

Pros: Ideal! Unbiased model. Strengthened by theory's prior testing, scrutiny and experimental evidence.

Cons: Requires research, expertise. May not have required data.

Data driven

Use personal knowledge and visualization to add variables to the model. (What we've been doing so far)

Pros: build up deep understanding of the data and model.

Cons: time consuming. Doesn't scale well with number of variables. Bias towards data set.

Stepwise regression

Old statistical technique. At each point either drop the worst variable in the model, or add the best variable not in the model.

Pros: easy to do (with `step`)

Cons: overfitting extremely easy, not statistically well-founded. Biased model.

Feature selection

Most machine learning techniques have variable selection built-in. For linear models, the technique is known in generally as penalised regression (e.g. lasso or elastic net)

Pros: statistical well found, usually fast. Diminishes bias.

Cons: While there are many penalised regression packages available on CRAN, none of them are easy to use

Non-linearity

“Linear” models

While linear models are always linear in the predictor variables, it’s possible to transform the predictors to add various types of non-linearity.

In this data, a good example of non-linearity is the relationship between `yearmade` and `totsqft`

```
qplot(yearmade, totsqft, data = rec,
  geom = "boxplot")
qplot(yearmade2, totsqft, data = rec,
  geom = "boxplot", group = yearmade2)

mod1 <- lm(totsqft ~ yearmade2, data = rec)
mod2 <- lm(totsqft ~ yearmade2 +
  I(yearmade2 ^ 2), data = rec)
mod3 <- lm(totsqft ~ yearmade2 +
  I(yearmade2 ^ 2) + I(yearmade2 ^ 3), data = rec)
```

Your turn

Compare the three models. What do they look like? Which is best at predicting `totsqft`?

```
rmse(mod1, rec)
rmse(mod2, rec)
rmse(mod3, rec)

grid <- data.frame(yearmade2 = seq(1920, 2005,
  length = 100))
grid$totsqft1 <- predict(mod1, grid)
grid$totsqft2 <- predict(mod2, grid)
grid$totsqft3 <- predict(mod3, grid)

qplot(yearmade2, totsqft1, data = grid, geom = "line")
qplot(yearmade2, totsqft2, data = grid, geom = "line")
qplot(yearmade2, totsqft3, data = grid, geom = "line")
```

Polynomial models

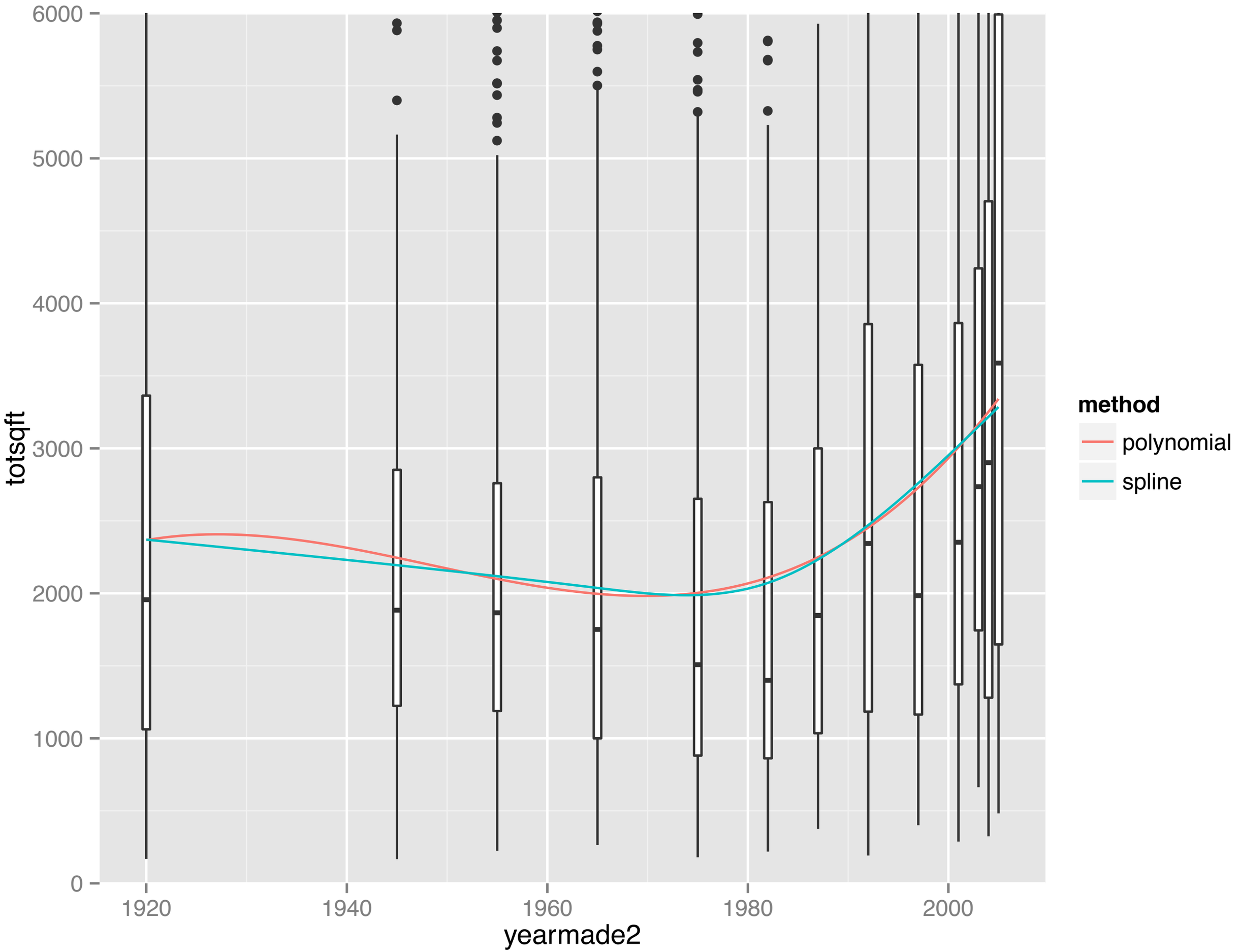
Easier (and numerically better behaved) to do:

```
mod_poly3 <- lm(totsqft ~ poly(yearmade2, 3),
data = rec)
```

Natural splines are an extension to polynomials that have linear behaviour outside the bounds of the data:

```
library(splines)
mod_ns3 <- lm(totsqft ~ ns(yearmade2, 3),
data = rec)
```

```r
# polynomial
grid$totsqft4 <- predict(mod_poly3, grid)
qplot(yearmade2, totsqft4, data = grid,
  geom = "line")

# natural splines
grid$totsqft5 <- predict(mod_ns3, grid)
qplot(yearmade2, totsqft5, data = grid,
  geom = "line")
```


6000
5000
4000
3000
2000
1000
0
totsqft
1920
1940
1960
1980
2000
yearmade2
method
polynomial
spline

```
# What do splines look like?
----------------------------
spline <- ns(rec$yearmade2, 5)
knots <- attr(spline, "knots")

x <- seq(1920, 2005, length = 100)
preds <- as.data.frame(predict(spline, x))
preds$x <- x
predsm <- reshape2::melt(preds, id = "x")

qplot(x, value, data = predsm, group = variable,
  geom = "line") +
  geom_vline(xintercept = knots, colour = "white",
    size = 2)
```