

# Model building: selecting a model

---

## Lecture 9

STA 371G

# Texas Suffers From A Doctor Shortage

By JONATHAN BAKER • NOV 1, 2017



Tweet



Share



Google+



Email

When it comes to having a high ratio of doctors to citizens, the State of Texas ranks near the bottom. In fact, [as The Dallas Morning News reports](#), 43 states have a higher proportion of primary care physicians to residents than Texas.

And West Texas suffers from a lack of doctors more than other parts of the state. There are 80 counties in Texas with five or fewer practicing doctors - many in West Texas. Thirty-five Texas counties have [no doctors at all](#).

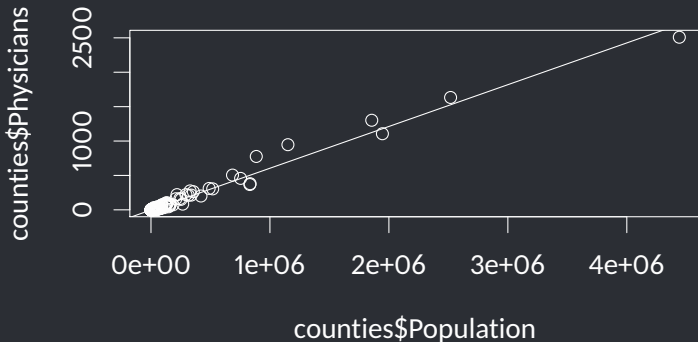


## What might explain why some counties have a doctor shortage?

- Small counties
- Poverty
- Health insurance
- Unemployment
- Large rural areas
- Something else?

## Population as a predictor of number of physicians

```
> popmodel <- lm(Physicians ~ Population, data=counties)
> plot(counties$Population, counties$Physicians)
> abline(popmodel)
```



## Transform and Subset the data

Let's define a new variable for physicians per 10,000 people:

```
> counties$PhysiciansPer10000 <-  
+   counties$Physicians / counties$Population * 10000
```

Then let's remove the very small counties as we can't reliably measure physician density in small counties:

```
> my.counties <- subset(counties, Population > 10000)
```

## Potential predictor variables

- **LandArea**: Area in square miles
- **PctRural**: Percentage rural land
- **MedianIncome**: Median household income
- **Population**: Population
- **PctUnder18**: Percent children
- **PctOver65**: Percent seniors
- **PctPoverty**: Percent below the poverty line
- **PctUninsured**: Percent without health insurance
- **PctSomeCollege**: Percent with some higher education
- **PctUnemployed**: Percent unemployed

## Parsimony

- We want a model that has a high  $R^2$  and a low  $s_e$ , because then the predictors are doing a good job of explaining  $Y$ —and our predictions will be more accurate.

## Parsimony

- We want a model that has a high  $R^2$  and a low  $s_e$ , because then the predictors are doing a good job of explaining  $Y$ —and our predictions will be more accurate.
- We also want a model that is simple, so it's easy to explain to a non-expert.



## Parsimony

- We want a model that has a high  $R^2$  and a low  $s_e$ , because then the predictors are doing a good job of explaining  $Y$ —and our predictions will be more accurate.
- We also want a model that is simple, so it's easy to explain to a non-expert.
- The ideal model is **parsimonious**: a good trade-off between simplicity (as few variables as possible) and a high  $R^2$ .

There is no purely mechanical procedure that will tell you what the most parsimonious model is; you need to use your judgement.

There is no purely mechanical procedure that will tell you what the most parsimonious model is; you need to use your judgement.

But with  $k$  variables there are  $2^k - 1$  possible models; for example, there are  $k = 10$  possible predictor variables in the data set, so there are 1,023 possible combinations of predictors you could use!

## General strategy

1. Use one or more procedures to generate candidate models: possible models that are worth considering.

## General strategy

1. Use one or more procedures to generate candidate models: possible models that are worth considering.
2. Select the candidate model with a reasonable tradeoff simplicity and predictive power (high  $R^2$ ).

## General strategy

1. Use one or more procedures to generate candidate models: possible models that are worth considering.
2. Select the candidate model with a reasonable tradeoff simplicity and predictive power (high  $R^2$ ).
3. Check assumptions and model diagnostics (more on this to come); apply transformations and other fixes if needed to the final model. If the problems are unfixable, select a different candidate model.

## Backward stepwise regression

1. Start with a “full” model containing all of the predictors.
2. Remove the least significant (highest  $p$ -value / smallest  $t$ -statistic) predictor.
3. Re-run the model with that predictor removed.
4. Repeat steps 2-3 until all predictors are significant.

## Forward stepwise regression

1. Start with a “null” model containing none of the predictors.
2. Try adding each predictor, one at a time, and pick the one that ends up being the most significant (lowest  $p$ -value / highest  $t$ -statistic) predictor.
3. Re-run the model with that predictor added.
4. Repeat steps 2-3 until no more significant predictors can be added.



## Other stepwise regression possibilities

- Add (or remove) variables one at a time based on the change in  $R^2$ , Adjusted  $R^2$ , or AIC (another similar model fit criterion) when that variable is added (or removed).
- Run the stepwise regression in both directions, allowing addition or removal of a variable at each step.
- R's step function incorporates both of these methods.

## The problem with stepwise regression

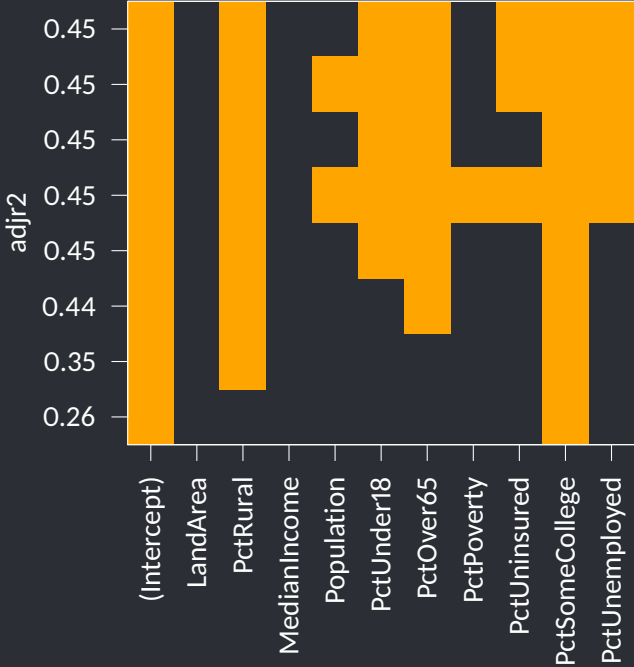
Stepwise regression will not necessarily give you the best model; by only adding or removing one variable at a time, you can get locked into a particular “path” that means you may never consider better models.

## Best subsets regression

- Computers are fast! Just let R try out all of the  $2^k - 1$  possible models for you.
- R will present you the model with the best Adjusted  $R^2$  for each possible number of predictors.

# Best-subsets regression

```
> library(leaps)
> plot(regsubsets(PhysiciansPer10000 ~ LandArea + PctRural
+               + MedianIncome + Population + PctUnder18
+               + PctOver65 + PctPoverty + PctUninsured
+               + PctSomeCollege + PctUnemployed,
+               data=my.counties), scale="adjr2")
```



- Best-subsets regression presents us with a candidate model for each possible number of predictors.
- The label on the y-axis show the Adjusted  $R^2$  value for the model corresponding to the filled-in squares for that row.

## Putting things together

- Look at multiple statistics. They generally say similar things.

## Putting things together

- Look at multiple statistics. They generally say similar things.
- Find the **parsimonious** middle ground between an underspecified model and extraneous variables.



## Putting things together

- Look at multiple statistics. They generally say similar things.
- Find the **parsimonious** middle ground between an underspecified model and extraneous variables.
- Fine-tune the model to ensure the model meets assumptions and captures key relationships: you may need to transform predictors and/or add interactions.

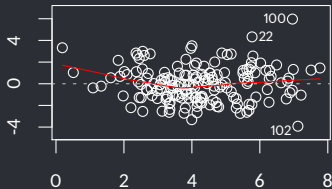
## Putting things together

- Look at multiple statistics. They generally say similar things.
- Find the **parsimonious** middle ground between an underspecified model and extraneous variables.
- Fine-tune the model to ensure the model meets assumptions and captures key relationships: you may need to transform predictors and/or add interactions.
- Think about logical reasons why certain predictors might be useful; don't just focus on  $p$ -values.

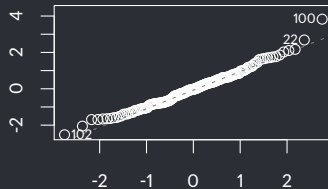
# Check assumptions of the best model

```
> candidate <- lm(PhysiciansPer10000 ~ PctRural + PctOver65  
+  
+ PctSomeCollege, data=my.counties)  
> plot(candidate)
```

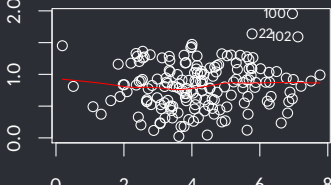
Residuals vs Fitted



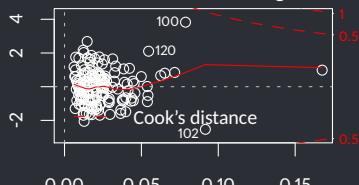
Normal Q-Q



Scale-Location



Residuals vs Leverage



# MYSTERY SPOT



OPEN

PLEASE SMOKE

50¢ MAKE YOUR  
OWN SODA

WANT A  
KID  
ATTRACTION, MAKE YOUR  
OWN



## How reliable is $R^2$ ?

- The mystery data set contains 20 predictor variables  $X_1$ - $X_{20}$ .

## How reliable is $R^2$ ?

- The mystery data set contains 20 predictor variables  $X_1$ - $X_{20}$ .
- Backwards stepwise regression or best subsets regression yields a data set with multiple significant predictors.

```
> parsimonious.model <- lm(Y ~ X10 + X13 + X16, data=mystery)
> summary(parsimonious.model)
```

Call:

```
lm(formula = Y ~ X10 + X13 + X16, data = mystery)
```

Residuals:

Min	1Q	Median	3Q	Max
-2.5839	-0.6636	-0.0255	0.6312	3.5081

Coefficients:

	Estimate	Std. Error	t value	Pr(> t )	
(Intercept)	0.006384	0.031188	0.205	0.8379	
X10	0.074640	0.030694	2.432	0.0152	*
X13	-0.065601	0.030809	-2.129	0.0335	*
X16	0.071064	0.032880	2.161	0.0309	*

---

Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.9857 on 996 degrees of freedom

Multiple R-squared: 0.01434, Adjusted R-squared: 0.01137

F-statistic: 4.829 on 3 and 996 DF, p-value: 0.00242

## Mystery revealed!

- This data set consists of nothing but random numbers!



## Mystery revealed!

- This data set consists of nothing but random numbers!
- Remember that when we set  $\alpha = 0.05$  as we usually do, our Type I error is 5%, meaning 1 out of roughly every 20 variables will be significant just by chance.

## Mystery revealed!

- This data set consists of nothing but random numbers!
- Remember that when we set  $\alpha = 0.05$  as we usually do, our Type I error is 5%, meaning 1 out of roughly every 20 variables will be significant just by chance.
- As a result, it's dangerous to build a model by just selecting the most significant from among a large list of variables.

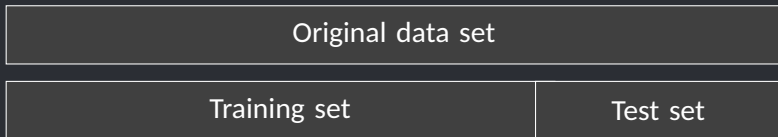
## Mystery revealed!

- This data set consists of nothing but random numbers!
- Remember that when we set  $\alpha = 0.05$  as we usually do, our Type I error is 5%, meaning 1 out of roughly every 20 variables will be significant just by chance.
- As a result, it's dangerous to build a model by just selecting the most significant from among a large list of variables.
- Doing so can result in **overfitting**: creating a model that fits the noise in the data well but won't generalize well to new data.

## Mystery revealed!

- This data set consists of nothing but random numbers!
- Remember that when we set  $\alpha = 0.05$  as we usually do, our Type I error is 5%, meaning 1 out of roughly every 20 variables will be significant just by chance.
- As a result, it's dangerous to build a model by just selecting the most significant from among a large list of variables.
- Doing so can result in **overfitting**: creating a model that fits the noise in the data well but won't generalize well to new data.
- In general, the  $R^2$  of a model gives an overoptimistic view of how well it will generalize to new data.

## Combatting overfitting with training and test sets



- Split the data into a **training set** and a **test set** (a typical split is 70% training set / 30% test set).
- We use the training set to build the model, and then evaluate the quality of the model on how well it predicts  $Y$  in the test set.

First, we'll take 50 random cases for the test set (about 30% of the  $n = 168$  cases in the whole data set), and the rest for the training set:

```
> test.cases <- sample(1:168, 50)
> training.cases <- setdiff(1:168, test.cases)
> training.set <- my.counties[training.cases,]
> test.set <- my.counties[test.cases,]
```

First, we'll take 50 random cases for the test set (about 30% of the  $n = 168$  cases in the whole data set), and the rest for the training set:

```
> test.cases <- sample(1:168, 50)
> training.cases <- setdiff(1:168, test.cases)
> training.set <- my.counties[training.cases,]
> test.set <- my.counties[test.cases,]
```

Then, we “train” the model using the cases in the **training** set, instead of the whole data set:

```
> candidate <- lm(PhysiciansPer10000 ~ PctRural + PctOver65
+                  + PctSomeCollege, data=training.set)
```

Finally, we predict  $Y$  for each value in the **test** set using this model:

```
> predicted.y <- predict(candidate, test.set)
```



Finally, we predict  $Y$  for each value in the **test** set using this model:

```
> predicted.y <- predict(candidate, test.set)
```

We can calculate  $R^2$  in the test set using the fact that

$R^2 = \text{cor}(\hat{Y}, Y)^2$ :

```
> y <- test.set$PhysiciansPer10000
```

```
> cor(predicted.y, y)^2
```

```
[1] 0.362666
```

Finally, we predict  $Y$  for each value in the **test** set using this model:

```
> predicted.y <- predict(candidate, test.set)
```

We can calculate  $R^2$  in the test set using the fact that

$R^2 = \text{cor}(\hat{Y}, Y)^2$ :

```
> y <- test.set$PhysiciansPer10000
```

```
> cor(predicted.y, y)^2
```

```
[1] 0.362666
```

This is somewhat lower than the  $R^2$  from the original model (0.45), but it's a fairer estimate of how good our model will perform on unseen data.

## Be careful of getting too crazy!

- A general guideline is that you should not even consider more than one variable for every 10 to 15 cases in your dataset.

## Be careful of getting too crazy!

- A general guideline is that you should not even consider more than one variable for every 10 to 15 cases in your dataset.
- Think about the meaning of the variables in your model. Be careful if the model looks too good to be true.

## Be careful of getting too crazy!

- A general guideline is that you should not even consider more than one variable for every 10 to 15 cases in your dataset.
- Think about the meaning of the variables in your model. Be careful if the model looks too good to be true.
- Do not just use a mechanical process for model selection and call it a day; you need to use your judgement and select a parsimonious model.

## Be careful of getting too crazy!

- A general guideline is that you should not even consider more than one variable for every 10 to 15 cases in your dataset.
- Think about the meaning of the variables in your model. Be careful if the model looks too good to be true.
- Do not just use a mechanical process for model selection and call it a day; you need to use your judgement and select a parsimonious model.
- Consider using a training/test set split to ensure you are not “capitalizing on chance.”

## Be careful of getting too crazy!

- A general guideline is that you should not even consider more than one variable for every 10 to 15 cases in your dataset.
- Think about the meaning of the variables in your model. Be careful if the model looks too good to be true.
- Do not just use a mechanical process for model selection and call it a day; you need to use your judgement and select a parsimonious model.
- Consider using a training/test set split to ensure you are not “capitalizing on chance.”
- Don't forget to check the model assumptions for your final model!