Economic Forecasting

Forecasting with regression models

Sebastian Fossati

University of Alberta | E493 | 2023

Outline

Linear regression

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_k x_{ik} + \varepsilon_i$$

- y_i is the variable we want to predict
- each x_{ij} is a "predictor"
- the coefficients $\beta_1, \ldots, \beta_k$ measure the effect of each predictor after taking account of the effect of all other predictors in the model
- ε_i is an error term

Least squares estimation

Since we do not know the values of $\beta_0, \beta_1, \ldots, \beta_k$, these need to be estimated from the data.

Least squares

The least squares estimates $\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_k$ are obtained by minimizing the sum of squared errors:

$$\sum_{i=1}^{N} e_i^2 = \sum_{i=1}^{N} (y_i - \hat{\beta}_0 - \hat{\beta}_1 x_{i1} - \cdots - \hat{\beta}_k x_{ik})^2$$

Least squares estimation

Actual values of y can then be decomposed into a **fitted value** and a **residual** such that $y_i = \hat{y}_i + e_i$.

- fitted values: $\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_{i1} + \cdots + \hat{\beta}_k x_{ik}$
- residuals: $e_i = y_i - \hat{\beta}_0 - \hat{\beta}_1 x_{i1} - \cdots - \hat{\beta}_k x_{ik}$

Goodness-of-fit

The R^2 of the regression is a useful summary of the model.

- it is the proportion of variance accounted for (explained) by the predictors
- it can be calculated as follows:

$$R^2 = \frac{\sum_{i=1}^{N}(\hat{y}_i - \bar{y})^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2}$$

- it is equal to the square of the correlation between y and $\hat{y}$
- $0 \leq R^2 \leq 1$, larger values are associated with better fit

Standard error of the regression

The standard error of the regression $\hat{\sigma}_e$ is another useful summary of the model.

- it can be calculated as follows:

$$\hat{\sigma}_e = \sqrt{\frac{1}{N-k-1}\sum_{i=1}^{N} e_i^2}$$

- k is the number of predictors in the model

Residual diagnostics

For forecasting purposes, we require the following assumptions:

- ε_i are uncorrelated and zero mean
- ε_i are uncorrelated with each x_{ij}

Residual diagnostics

For forecasting purposes, we require the following assumptions:

- ε_i are uncorrelated and zero mean
- ε_i are uncorrelated with each x_{ij}

It is **useful** to also have $\varepsilon_i \sim \text{N}(0, \sigma^2)$ when producing prediction intervals or doing statistical tests.

Other residual plots

Useful for spotting outliers and whether the linear model was appropriate.

- scatterplot of residuals against predictors x_{ij}
- scatterplot of residuals against fitted values $\hat{y}_i$
- expect to see scatterplots resembling a horizontal band with no values too far from the band and no patterns such as curvature or increasing spread

Residual patterns

- if a plot of the residuals vs any predictor in the model shows a pattern, then the relationship is nonlinear
- if a plot of the residuals vs any predictor **not** in the model shows a pattern, then the predictor should be added to the model
- if a plot of the residuals vs fitted values shows a pattern, then there is heteroscedasticity in the errors (try a transformation)

Outliers and influential observations

Things to watch for…

- *outliers*: observations that produce large residuals
- *influential observations*: removing them would markedly change the coefficients (often outliers in the x variable)
- data should not be removed without a good explanation of why they are different

Prediction

Set up:

- let y^0 be the **new** value for which we would like a forecast
- and $x_1^0, \cdots, x_k^0$ the values of the predictors of y^0

Predicted value

$$\hat{y}^0 = \hat{\beta}_0 + \hat{\beta}_1 x_1^0 + \cdots + \hat{\beta}_k x_k^0$$

Prediction interval

To compute a prediction interval

- ignoring parameter estimation uncertainty (that is, sampling error in $\hat{y}^0$)
- and assuming forecast errors are normally distributed, then an approximate 95% PI is

Prediction interval

$$\hat{y}^0 \pm 1.96\hat{\sigma}_e$$

Outline

Regression models in R

Some useful functions:

- `lm()`: linear regression model
- `tslm()`: regression model for time series data
- `summary()`: prints standard regression output
- `coef()`, `vcov()`, `resid()`, `fitted()`: extract the regression coefficients, (estimated) covariance matrix, residuals, and fitted values respectively
- `confint()`: confidence intervals for the regression coefficien
- `predict()`: predictions for new data
- `coeftest`: coefficient tests
- `NeweyWest()`: Newey-West HAC covariance matrix
- `vcovHAC()`: more HAC covariance matrices

Simulated example

```
# simulate data
n.obs <- 200
x1 <- rnorm(n.obs)
x2 <- rnorm(n.obs)
x3 <- rnorm(n.obs)
y <- .75*x1 + .50*x2 + .25*x3 + rnorm(n.obs, mean = 0, sd = 2)
# some irrelevant variables
x4 <- rnorm(n.obs, mean = 0, sd = 4)
x5 <- rnorm(n.obs, mean = 0, sd = 5)
# set data frame
data <- data.frame(y, x1, x2, x3, x4, x5)
head(data, 2)
```

```
##         y      x1     x2       x3     x4     x5
## 1 -0.3424 -1.2071 0.4852 -1.22682 -4.096 -6.027
## 2  1.7904  0.2774 0.6968  0.03615 -5.551  1.507
```

Simulated example

Consider the following three models to be estimated:

- y_i on x_{i1}
- y_i on x_{i1}, x_{i2}, x_{i3}
- y_i on all five variables

```
# estimate models
model1 <- lm(y ~ x1)    # missing variables
model2 <- lm(y ~ x1 + x2 + x3)  # correctly specified model
model3 <- lm(y ~ x1 + x2 + x3 + x4 + x5)    # irrelevant variables
```

Simulated example

```
# estimate models
coeftest(model2)    # correctly specified model
```

```
##
## t test of coefficients:
##
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.0733     0.1357   -0.54   0.5900
## x1            0.9658     0.1332    7.25  9.4e-12 ***
## x2            0.4420     0.1350    3.27   0.0013 **
## x3            0.2338     0.1320    1.77   0.0780 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Simulated example

```
# estimate models
coeftest(model3)    # irrelevant variables
```

```
##
## t test of coefficients:
##
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.0691     0.1361   -0.51   0.6121
## x1            0.9309     0.1369    6.80  1.3e-10 ***
## x2            0.4337     0.1354    3.20   0.0016 **
## x3            0.2391     0.1322    1.81   0.0720 .
## x4           -0.0219     0.0352   -0.62   0.5350
## x5            0.0319     0.0289    1.11   0.2696
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Simulated example

```
# add new observation
new <- list(x1 = 1, x2 = 1, x3 = 1, x4 = 1, x5 = 1)

# predict y using model 2
pred_new <- predict(
    model2,
    newdata = new,
    se.fit = TRUE,
    interval = "prediction"
  )
pred_new$fit
```

```
##     fit    lwr   upr
## 1 1.568 -2.222 5.359
```

Outline

Selecting predictors

When there are many predictors, how should we choose which ones to use?

Selecting predictors

When there are many predictors, how should we choose which ones to use?

What not to do!

- plot y against a particular predictor (x_j) and if it shows no noticeable relationship, drop it
- do a multiple linear regression on all the predictors and disregard all variables whose p-values are greater than 0.05
- maximize R^2 or minimize MSE

Comparing regression models

Computer output for regression will always give the R^2 value.

However ...

- R^2 does not correct for "degrees of freedom"
- adding *any* variable tends to increase the value of R^2, even if that variable is irrelevant

Comparing regression models

Computer output for regression will always give the R^2 value.

However ...

- R^2 does not correct for "degrees of freedom"
- adding *any* variable tends to increase the value of R^2, even if that variable is irrelevant

To overcome this problem, we can use adjusted-R^2:

$$\bar{R}^2 = 1 - (1 - R^2)\frac{N-1}{N-k-1}$$

where k is the number of predictors and N is the number of observations.

Maximizing $\bar{R}^2$ is equivalent to minimizing $\hat{\sigma}^2$.

Bias-variance trade-off

Bias-variance trade-off: as we use more flexible models (more predictors, quadratics, interactions, etc.), the variance will increase and the bias will decrease.

Model selection criteria

The model containing all the predictors will always have the smallest SSR and the largest R^2.

Solution: introduce a *trade-off* between fit and parsimony.

Model selection criteria

Schwarz (Bayesian) information criterion (BIC):

$$\text{BIC} = \ln\left(\frac{1}{N}\sum_{i=1}^{N} e_i^2\right) + k\frac{\ln(N)}{N}$$

Akaike's information criterion (AIC):

$$\text{AIC} = \ln\left(\frac{1}{N}\sum_{i=1}^{N} e_i^2\right) + k\frac{2}{N}$$

Model selection criteria

Remarks:

- BIC and AIC can be used when the models are not nested
- models with **lowest BIC or AIC are preferred**
- BIC has heavier penalty (selects smaller models)

Intuition is similar to adjusted-R^2:

$$\bar{R}^2 = 1 - \frac{\text{SSR}/(N-k)}{\text{SST}/(N-1)}$$

- unlike the R^2 statistic, the adjusted-R^2 penalizes the inclusion of unnecessary variables in the model

Simulated example

```
CV(model1)
```

```
##       CV      AIC     AICc      BIC    AdjR2
##   3.8934 273.8626 273.9851 283.7576   0.2091
```

```
CV(model2)
```

```
##       CV      AIC     AICc      BIC    AdjR2
##   3.7026 263.4543 263.7635 279.9459   0.2566
```

```
CV(model3)
```

```
##       CV      AIC     AICc      BIC    AdjR2
##   3.7398 265.7759 266.3592 288.8641   0.2552
```

- model 2 is preferred (model 2 > 3 > 1)

Outline

Mean Squared Error

In the regression setting we can use the **mean squared error** (MSE) to evaluate the predictive performance of a model:

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

- in a linear model $\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_{i1} + \cdots + \hat{\beta}_k x_{ik}$, for example
- $e_i = y_i - \hat{y}_i$ is the prediction error

Cross-validation

The objective of *cross-validation* is to evaluate the predictive performance of a model in a *test sample* not used for fitting the model.

Three options:

1. the validation set approach
2. leave-one-out cross-validation (LOOCV)
3. *k*-fold cross-validation

The validation set approach

The **validation set approach** involves randomly splitting the set of N observations into two parts:

1. a *training set*
2. a *test set* (or validation set or hold-out set)

Steps:

- use the training set to fit the model
- use the fitted model to generate a prediction for the observations in the test set
- compute MSE for observations in the test set (test error rate)

The validation set approach

FIGURE 5.1. *A schematic display of the validation set approach. A set of n observations are randomly split into a training set (shown in blue, containing observations 7, 22, and 13, among others) and a validation set (shown in beige, and containing observation 91, among others). The statistical learning method is fit on the training set, and its performance is evaluated on the validation set.*

Leave-one-out cross-validation (LOOCV)

LOOCV involves splitting the set of N observations in two: a test set of a single observation and a training set with the remaining $N - 1$ observations.

Steps:

- start using (x_1, y_1) for validation and the remaining $N - 1$ observations $(x_2, y_2), \ldots, (x_N, y_N)$ as the training set for fitting the model
- use the fitted model to generate a prediction $\hat{y}_1$ and compute the prediction error $\text{MSE}_1 = (y_1 - \hat{y}_1)^2$
- repeat for $i = 2, \ldots, N$ to generate $\text{MSE}_2, \ldots, \text{MSE}_N$
- the test MSE is $\text{CV}_{(N)} = \frac{1}{N} \sum_{i=1}^{N} \text{MSE}_i$

Leave-one-out cross-validation (LOOCV)

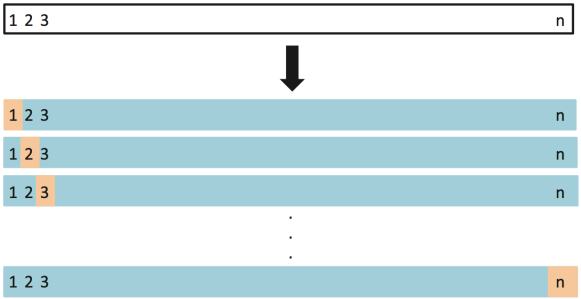


FIGURE 5.3. *A schematic display of LOOCV. A set of n data points is repeatedly split into a training set (shown in blue) containing all but one observation, and a validation set that contains only that observation (shown in beige). The test error is then estimated by averaging the n resulting MSE's. The first training set contains all but observation 1, the second training set contains all but observation 2, and so forth.*

k-fold cross-validation

k-fold CV involves randomly dividing the set of N observations into k groups (or folds) of approximately equal size.

Steps:

- use the first fold for validation and the observations in the remaining $k - 1$ folds as the training set for fitting the model
- use the fitted model to generate a prediction for the observations in the first fold (the held-out fold) and compute the prediction error, MSE_1
- repeat for the remaining $k - 1$ folds to generate $\text{MSE}_2, \ldots, \text{MSE}_k$
- the test MSE is $\text{CV}_{(k)} = \frac{1}{k} \sum_{i=1}^{k} \text{MSE}_i$

k-fold cross-validation

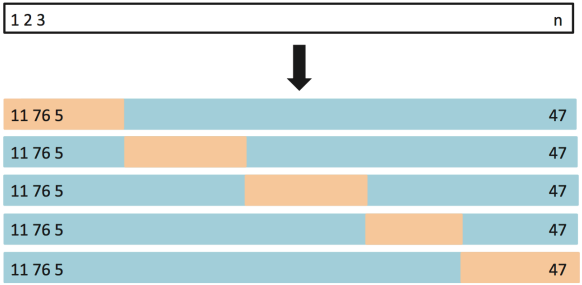


FIGURE 5.5. *A schematic display of* 5*-fold CV. A set of* n *observations is randomly split into five non-overlapping groups. Each of these fifths acts as a validation set (shown in beige), and the remainder as a training set (shown in blue). The test error is estimated by averaging the five resulting MSE estimates.*

Cross-validation remarks

The validation set approach:

- the test error rate can be very variable
- may overestimate true test error rate

Cross-validation remarks

LOOCV:

- there is no randomness in the training/test set split
- but can be very time consuming

Cross-validation remarks

k-fold CV:

- variability in test error rate is much lower than in validation set approach
- less time consuming than LOOCV
- using $k = 5$ or $k = 10$ usually works well ($k = 2$? $k = N$?)

Simulated example

```
# validation set approach
cv.error <- rep(NA,3)
# split the sample into train/test sets
train <- sample(nrow(data), round(nrow(data)/2))
# model 1
model1.cv <- lm(y ~ x1, subset = train)
cv.error[1] <- mean((y - predict(model1.cv, data))[-train]^2)
# model 2
model2.cv <- lm(y ~ x1 + x2 + x3, subset = train)
cv.error[2] <- mean((y - predict(model2.cv, data))[-train]^2)
# model 3
model3.cv <- lm(y ~ x1 + x2 + x3 + x4 + x5, subset = train)
cv.error[3] <- mean((y - predict(model3.cv, data))[-train]^2)
# results
print(cv.error)
```

```
## [1] 3.765 3.830 4.008
```

Simulated example

```
# leave-one-out cross-validation
cv.error.1 <- rep(NA,3)
# model 1
model1.cv <- glm(y ~ x1)
cv.error.1[1] <- cv.glm(data, model1.cv)$delta[1]
# model 2
model2.cv <- glm(y ~ x1 + x2 + x3)
cv.error.1[2] <- cv.glm(data, model2.cv)$delta[1]
# model 3
model3.cv <- glm(y ~ x1 + x2 + x3 + x4 + x5)
cv.error.1[3] <- cv.glm(data, model3.cv)$delta[1]
# results
print(cv.error.1)
```

```
## [1] 3.893 3.703 3.740
```

Simulated example

```
# k-fold cross-validation
cv.error.2 <- rep(NA,3)
# model 1
model1.cv <- glm(y ~ x1)
cv.error.2[1] <- cv.glm(data, model1.cv, K = 10)$delta[1]
# model 2
model2.cv <- glm(y ~ x1 + x2 + x3)
cv.error.2[2] <- cv.glm(data, model2.cv, K = 10)$delta[1]
# model 3
model3.cv <- glm(y ~ x1 + x2 + x3 + x4 + x5)
cv.error.2[3] <- cv.glm(data, model3.cv, K = 10)$delta[1]
# results
print(cv.error.2)
```

```
## [1] 3.915 3.684 3.716
```

Simulated example

The function `CV` (available through the package `fpp2`) performs LOOCV and reports other useful statistics.

```
CV(model1)
```

```
##       CV      AIC     AICc      BIC    AdjR2
##   3.8934 273.8626 273.9851 283.7576   0.2091
```

```
CV(model2)
```

```
##       CV      AIC     AICc      BIC    AdjR2
##   3.7026 263.4543 263.7635 279.9459   0.2566
```

```
CV(model3)
```

```
##       CV      AIC     AICc      BIC    AdjR2
##   3.7398 265.7759 266.3592 288.8641   0.2552
```

Outline

Best subset selection

The methods described before are useful to compare a reduced number of models (for example, two models based on different economic theories). But sometimes economic theory will not provide much guidance.

Finding the relevant predictors is called *variable selection*. Two main reasons to perform variable selection:

- *model interpretability*: including irrelevant variables leads to unnecessary complexity
- *prediction accuracy*: including irrelevant variables leads to less accurate predictions, specially when N is not much larger than p

Best subset selection

Best subset selection involves estimating a least squares regression for each possible combination of the p predictors.

Steps:

- fit a model with no predictors, call it the null model M_0
- fit all models that contain exactly $k = 1$ predictors and find the model with the smallest SSR, call it M_1
- repeat for $k = 2, \ldots, p$ to generate $M_2, \ldots, M_p$
- select a single best model from among $M_0, \ldots, M_p$ using the cross-validation prediction error, BIC, AIC, or adjusted-R^2

Best subset selection

Remarks: the number of models grows as p increases, with $p = 10$ there are $2^{10} = 1024$ possible models!

Simulated example

```
regfit.all <- regsubsets(y~. , data = data, nvmax = 5)
(reg.summary <- summary(regfit.all))
```

```
## Subset selection object
## Call: regsubsets.formula(y ~ ., data = data, nvmax = 5)
## 5 Variables  (and intercept)
##    Forced in Forced out
## x1     FALSE      FALSE
## x2     FALSE      FALSE
## x3     FALSE      FALSE
## x4     FALSE      FALSE
## x5     FALSE      FALSE
## 1 subsets of each size up to 5
## Selection Algorithm: exhaustive
##          x1  x2  x3  x4  x5
## 1  ( 1 ) "*" " " " " " " " "
## 2  ( 1 ) "*" "*" " " " " " "
## 3  ( 1 ) "*" "*" "*" " " " "
## 4  ( 1 ) "*" "*" "*" " " "*"
```

Simulated example

```
# ssr statistics
print(reg.summary$rss)
```

```
## [1] 763.3 721.6 710.2 705.7 704.3
```

```
# bic statistics
print(reg.summary$bic)
```

```
## [1] -37.34 -43.27 -41.15 -37.13 -32.23
```

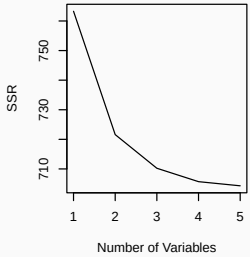
```
# cp statistics (same model as aic)
print(reg.summary$cp)
```

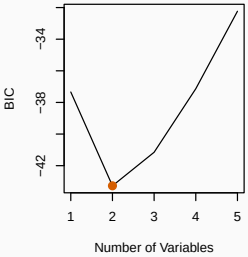
```
## [1] 14.249  4.767  3.635  4.386  6.000
```

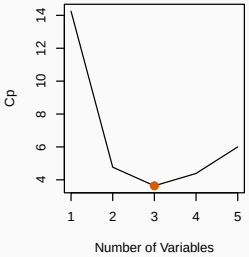
```
# adjusted-rsq statistics
print(reg.summary$adjr2)
```

```
## [1] 0.2091 0.2485 0.2566 0.2576 0.2552
```

Simulated example







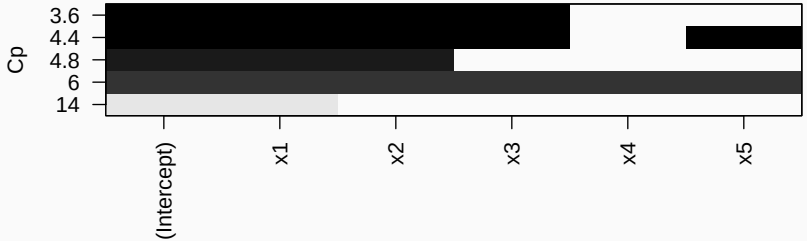
- BIC selects $k = 2$, AIC (Cp) selects $k = 3$

Simulated example

```
# plot variable
plot(regfit.all, scale = "bic")
```

Simulated example

```
# plot variable
plot(regfit.all, scale = "Cp")
```



Outline

Example: Used cars

Suppose you want to sell your car of a certain make, type, year, miles, condition and other features.

Prediction analysis can help you uncover the average advertised price of cars with similar characteristics.

- that helps decide what price you may want to put on your ad

Example: Used cars

Consider a sample of offers for used Toyota Camry cars in 2018 in Chicago (Békés and Kézdi, 2021).

Data:

- source: scraped from a website
- characteristics: year of make (age), odometer (miles), etc.
- data cleaning: drop erroneous observations, hybrid cars, trucks…

Example: Used cars

Load data set and filter offers from the Chicago area.

```
# read data
data <-
  read.csv(
    "data/used_cars_work.csv",
    header = TRUE,
    stringsAsFactors = TRUE
  )
# focus only on Chicago
data <- data %>%
  filter(area == "chicago")
```

Example: Used cars

We will use the following predictors:

- age: measuring how old is the car (continuous, linear)
- odometer: measuring miles the car traveled (continuous, linear)
- car type: LE, XLE, SE (missing in about 30% of the observations, factor–set of dummies, incl. N/A)
- condition: good condition, excellent condition, or it is like new (missing for about one third of the ads, factor–set of dummies, incl. N/A)
- car's engine has 6 cylinders (20% of ads say this; 43% says 4 cylinders, and the rest has no information, binary for 6 cylinders)

Example: Used cars

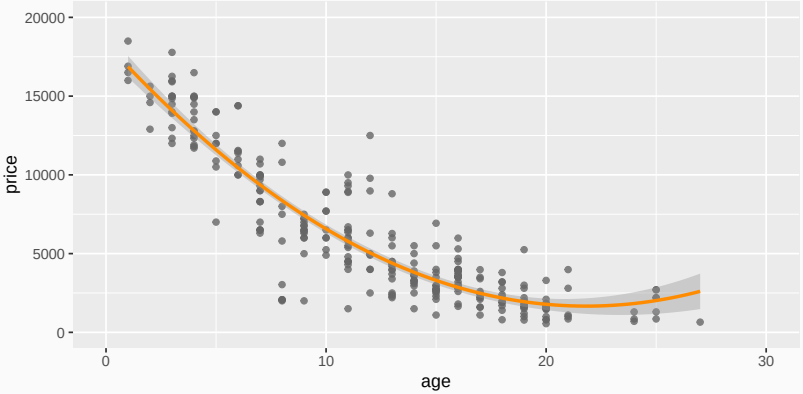
Summary statistics for some of the variables.

```
# data summary
data %>%
  dplyr::select(price, age, odometer) %>%
  summary()
```

```
##      price            age           odometer
##  Min.   :  550   Min.   : 1.0   Min.   : 0.232
##  1st Qu.: 2500   1st Qu.: 7.0   1st Qu.: 8.140
##  Median : 4400   Median :13.0   Median :13.656
##  Mean   : 6061   Mean   :12.3   Mean   :12.522
##  3rd Qu.: 8995   3rd Qu.:17.0   3rd Qu.:16.430
##  Max.   :18495   Max.   :27.0   Max.   :25.300
```

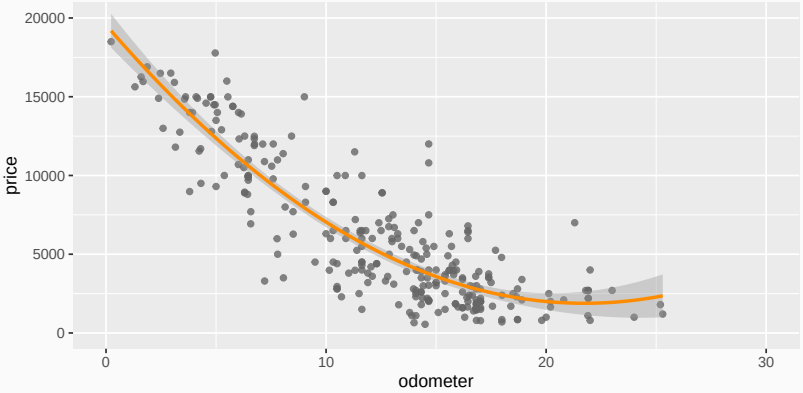
Example: Used cars

There is evidence of **nonlinear** relationships.



Example: Used cars

There is evidence of **nonlinear** relationships.



Example: Used cars

Some predictive models:

- Model 1: age, age squared
- Model 2: age, age squared, odometer, odometer squared
- Model 3: age, age squared, odometer, odometer squared, LE, XLE, SE, like new condition, excellent condition, good condition
- Model 4: age, age squared, odometer, odometer squared, LE, XLE, SE, like new condition, excellent condition, good condition, cylinder

Example: Used cars

Remarks:

1. when doing prediction, coefficients are less important
2. but we shall use them for sanity check: age negative, convex (flattens out)

Example: Used cars

```
# estimate models
coeftest(reg2)
```

```
##
## t test of coefficients:
##
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 20101.69     374.20   53.72  < 2e-16 ***
## age          -888.49      75.16  -11.82  < 2e-16 ***
## agesq          18.58       2.65    7.02  1.8e-11 ***
## odometer     -807.51      84.96   -9.50  < 2e-16 ***
## odometersq     19.32       3.12    6.19  2.2e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Example: Used cars

Make a prediction using Model 3.

```
# add new observation
new <-
  list(
    age = 10, agesq = 10^2, odometer = 12, odometersq = 12^2,
    SE = 0, XLE = 0, LE = 1,
    cond_likenew = 0, cond_excellent = 1, cond_good = 0, cylind6 = 0,
    price=NA
  )
# predict price with model 3
pred_new <- predict(reg3, newdata = new, se.fit = TRUE, interval = "prediction")
pred_new$fit
```

```
##    fit  lwr  upr
## 1 6124 3450 8799
```

Example: Used cars

But... which model should we use?

```
# evaluation of the models
table <- rbind(CV(reg1), CV(reg2), CV(reg3), CV(reg4))
rownames(table) <- c('Model1', 'Model2', 'Model3', 'Model4')
table
```

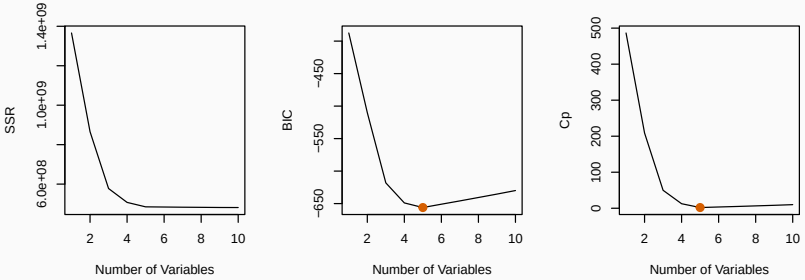
```
##              CV  AIC AICc  BIC  AdjR2
## Model1 3134869 4206 4206 4221 0.8456
## Model2 1867851 4060 4060 4082 0.9089
## Model3 1887761 4058 4059 4102 0.9113
## Model4 1895462 4059 4061 4107 0.9113
```

Example: Used cars

Perform **best subset selection** using all the variables in Model 4 allowing for up to 10 variables.

```
# select variables
data2 <- data %>%
  dplyr::select(
    age, agesq, odometer, odometersq, SE, LE, XLE,
    cond_likenew, cond_excellent, cond_good, price,
    cylind6
  )
#
regfit.all.2 <- regsubsets(price~. , data = data2, nvmax = 10)
reg.summary.2 <- summary(regfit.all.2)
```

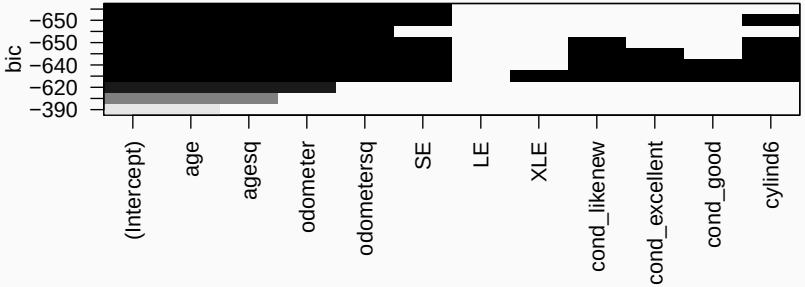
Example: Used cars



- BIC and AIC (Cp) select $k = 5$

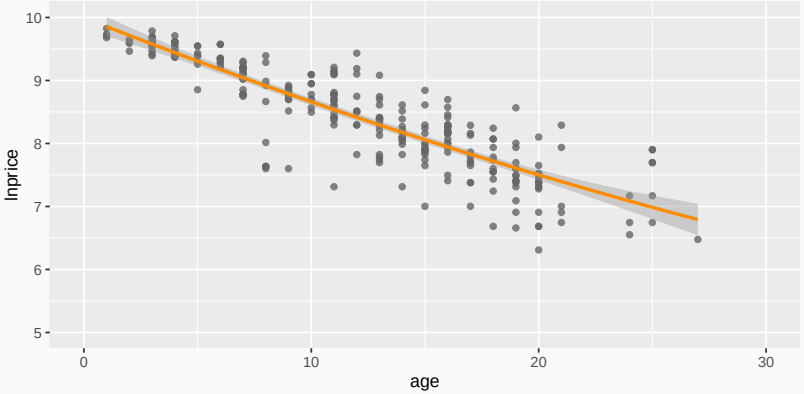
Example: Used cars

```
# plot variable
plot(regfit.all.2, scale = "bic")
```



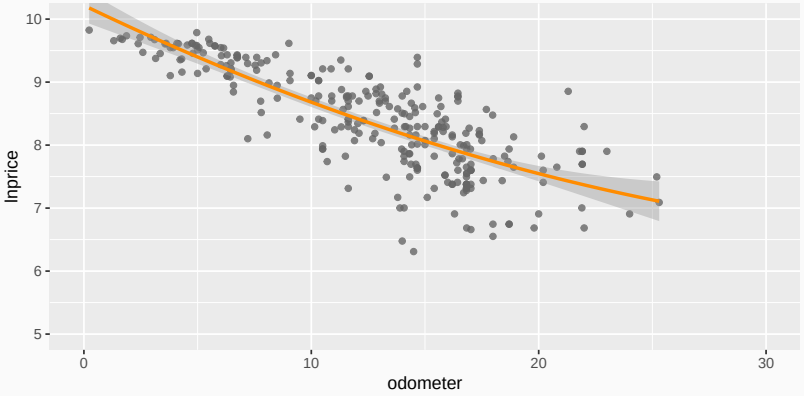
Example: Used cars

Should we work in levels or **logs**?



Example: Used cars

Should we work in levels or **logs**?

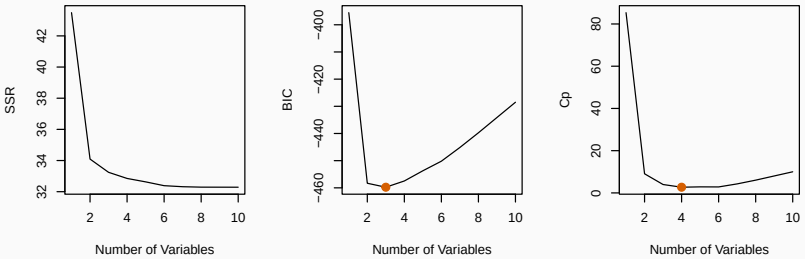


Example: Used cars

Perform **best subset selection** using all the variables in Model 4 allowing for up to 10 variables and prices in logs.

```
# select variables
data3 <- data %>%
  dplyr::select(
    age, agesq, odometer, odometersq, SE, LE, XLE,
    cond_likenew, cond_excellent, cond_good, lnprice,
    cylind6
  )
#
regfit.all.3 <- regsubsets(lnprice~. , data = data3, nvmax = 10)
reg.summary.3 <- summary(regfit.all.3)
```

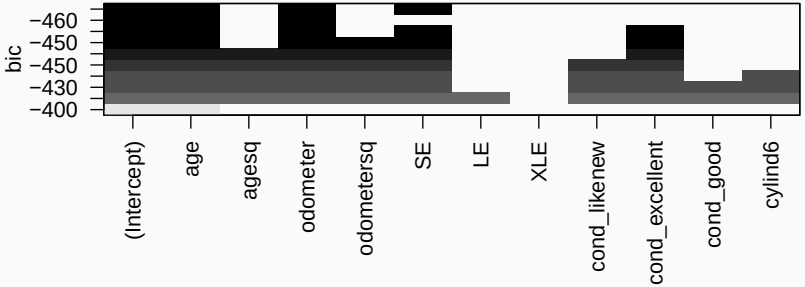
Example: Used cars



- BIC selects $k = 3$, AIC (Cp) selects $k = 4$
- a simple transformation (taking logs) can greatly simplify your predictive model

Example: Used cars

```
# plot variable
plot(regfit.all.3, scale = "bic")
```



Outline

Correlation is not causation

Remarks:

- when x is useful for predicting y, it is not necessarily causing y (e.g., predict number of drownings y using number of ice-creams sold x)
- correlations are useful for forecasting, even when there is no causality
- better models usually involve causal relationships (e.g., temperature x and people z to predict drownings y)

Multicollinearity

In regression analysis, multicollinearity occurs when…

- two predictors are highly correlated (i.e., the correlation between them is close to ± 1)
- a linear combination of some of the predictors is highly correlated with another predictor
- a linear combination of one subset of predictors is highly correlated with a linear combination of another subset of predictors

Multicollinearity

If multicollinearity exists…

- numerical estimates of coefficients may be wrong
- can't rely on *p*-values to determine significance
- there is no problem with model *predictions* provided the predictors used for forecasting are within the range used for fitting
- omitting variables can help
- combining variables can help