Regression and Classification

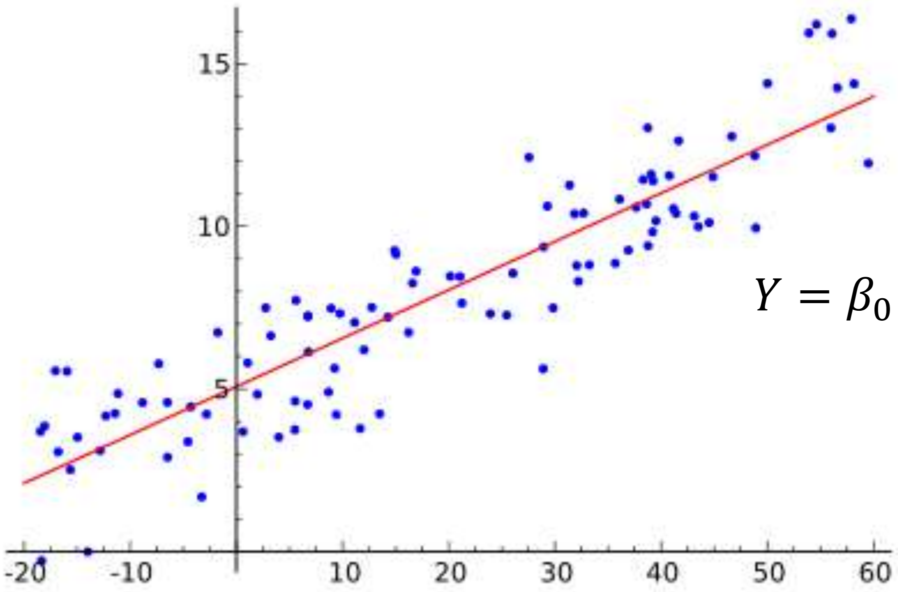
Presented By:
Sri Krishnamurthy, CFA, CAP
www.QuantUniversity.com
sri@quantuniversity.com

Regression

Multiple linear regression

Linear Regression

- In statistics, **linear regression** is an approach for modeling the relationship between a scalar dependent variable y and one or more explanatory variables (or independent variables) denoted X. The case of one explanatory variable is called simple **linear regression**.



$$Y = \beta_0 + \beta_1 X_1$$

Multiple linear regression

- In practice we often have more than one predictor.
- Instead of fitting a separate simple linear regression model for each predictor, a better approach is to extend the simple linear regression model so that it can directly accommodate multiple predictors.
- We can do this by giving each predictor a separate slope coefficient in a single model.
- In general, suppose that we have *p* distinct predictors. Then the multiple linear regression model takes the form

$$y_i = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p + \epsilon$$

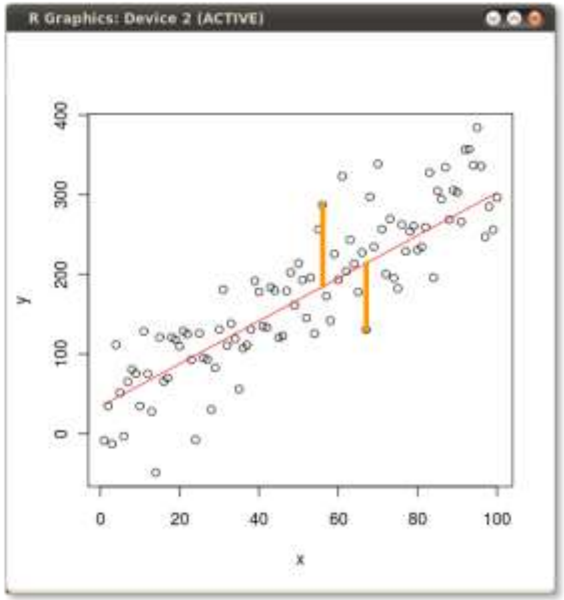
Multiple linear regression

- Given estimates $\hat{\beta}_0,\ \hat{\beta}_1, \dots, \hat{\beta}_p$ We can make predictions using the formula

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2 + \cdots + \hat{\beta}_p x_p$$

- The parameters are estimated using the same least squares approach that we saw in the context of simple linear regression

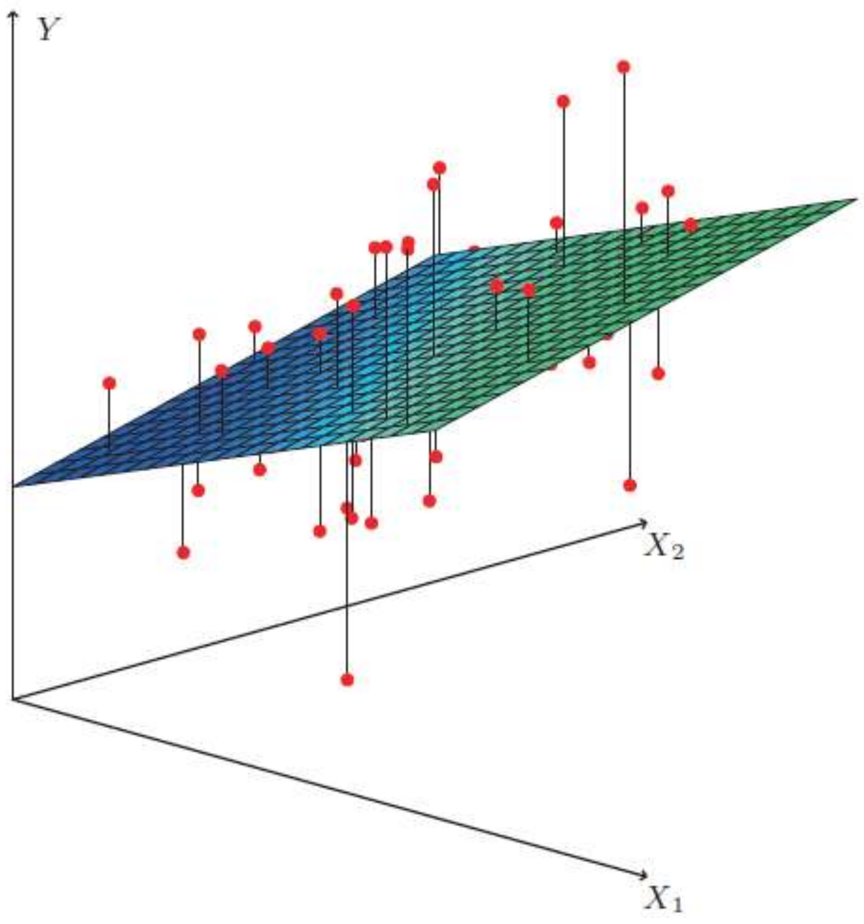
$$RSS = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$



QuantUniversity, LLC

Multiple linear regression

- In a three-dimensional setting, with two predictors and one response, the least squares regression line becomes a plane.
- The plane is chosen to minimize the sum of the squared vertical distances between each observation (shown in red) and the plane.



Example using R

- Libraries
  - The **library()** function is used to load *libraries*, or groups of functions, or data sets that are not included in the base R distribution.
  - Here we load the MASS package, which is a very large collection of data sets and functions. We also load the ISLR package, which includes the data sets associated with this topic.

```
> library(MASS)
> library(ISLR)
```

  - ISLR must be downloaded the first time it is used.

Example study using R

- Multiple linear regression
  - The MASS library contains the Boston data set, which records medv (median house value) for 506 neighborhoods around Boston.
  - We will seek to predict medv using 13 predictors such as rm (average number of rooms per house), age (average age of houses), and lstat (percent of households with low socioeconomic status).

```
> fix(Boston)
> names(Boston)
 [1] "crim"    "zn"      "indus"   "chas"
 [5] "nox"     "rm"      "age"     "dis"
 [9] "rad"     "tax"     "ptratio" "black"
[13] "lstat"   "medv"
```

See *Supervised learning-2.R > Multiple linear regression*

Example using R

- Multiple linear regression
  - In order to fit a multiple linear regression model using least squares, we again use the **lm()** function.
  - The syntax **lm(y∼x1+x2+x3)** is used to fit a model with three predictors, **x1**, **x2**, and **x3**.

```
> lm.fit=lm(medv~lstat+age, data=Boston)
> summary(lm.fit)

Call:
lm(formula = medv ~ lstat + age, data = Boston)

Residuals:
    Min      1Q  Median      3Q     Max 
-15.981  -3.978  -1.283   1.968  23.158 

Coefficients:
            Estimate Std. Error t value
(Intercept) 33.22276    0.73085  45.458
lstat       -1.03207    0.04819 -21.416
age          0.03454    0.01223   2.826
            Pr(>|t|)    
(Intercept)  < 2e-16 ***
lstat        < 2e-16 ***
age          0.00491 ** 
---
Signif. codes:  
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.173 on 503 degrees of freedom
Multiple R-squared:  0.5513,    Adjusted R-squared:  0.5495 
F-statistic:   309 on 2 and 503 DF,  p-value: < 2.2e-16
```

See *Supervised learning-2.R > Multiple linear regression*

QuantUniversity, LLC

Example study using R

- Multiple linear regression
  - The Boston data set contains 13 variables, and so it would be cumbersome to have to type all of these in order to perform a regression using all of the predictors. Instead, we can use the following short-hand:

```
> lm.fit=lm(medv~., data=Boston)
> summary(lm.fit)
```

```
Call:
lm(formula = medv ~ ., data = Boston)

Residuals:
    Min      1Q  Median      3Q     Max 
-15.595  -2.730  -0.518   1.777  26.199 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept)  3.646e+01  5.103e+00   7.144 3.28e-12 ***
crim        -1.080e-01  3.286e-02  -3.287 0.001087 ** 
zn           4.642e-02  1.373e-02   3.382 0.000778 ***
indus        2.056e-02  6.150e-02   0.334 0.738288    
chas         2.687e+00  8.616e-01   3.118 0.001925 ** 
nox         -1.777e+01  3.820e+00  -4.651 4.25e-06 ***
rm           3.810e+00  4.179e-01   9.116  < 2e-16 ***
age          6.922e-04  1.321e-02   0.052 0.958229    
dis         -1.476e+00  1.995e-01  -7.398 6.01e-13 ***
rad          3.060e-01  6.635e-02   4.613 5.07e-06 ***
tax         -1.233e-02  3.760e-03  -3.280 0.001112 ** 
ptratio     -9.527e-01  1.308e-01  -7.283 1.31e-12 ***
black        9.312e-03  2.686e-03   3.467 0.000573 ***
lstat       -5.248e-01  5.072e-02 -10.347  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.745 on 492 degrees of freedom
Multiple R-squared:  0.7406,    Adjusted R-squared:  0.7338 
F-statistic: 108.1 on 13 and 492 DF,  p-value: < 2.2e-16
```

See *Supervised learning-2.R > Multiple linear regression*

Example study using R

- Multiple linear regression
  - In the above regression output, age has a high p-value. So if we wish to run a regression excluding this predictor, we can use the following syntax

```
> lm.fit1=lm(medv~.-age, data=Boston)
> summary(lm.fit1)
```

See *Supervised learning-2.R > Multiple linear regression*

Example study using python

- First, we need to import the numpy and scikit-learn packages.

```
In [1]: import numpy as np
```

```
In [2]: import sklearn
        from sklearn import *
```

- Then, import the Boston data set.

```
In [3]: boston = datasets.load_boston()
```

```
In [4]: print (boston)
```


```
{'data': array([[  6.32000000e-03,   1.80000000e+01,   2.31000000e+0
0, ...,
          1.53000000e+01,   3.96900000e+02,   4.98000000e+00],
       [  2.73100000e-02,   0.00000000e+00,   7.07000000e+00, ...,
          1.78000000e+01,   3.96900000e+02,   9.14000000e+00],
       [  2.72900000e-02,   0.00000000e+00,   7.07000000e+00, ...,
          1.78000000e+01,   3.92830000e+02,   4.03000000e+00],
       ...,
       [  6.07600000e-02,   0.00000000e+00,   1.19300000e+01, ...,
          2.10000000e+01,   3.96900000e+02,   5.64000000e+00],
       [  1.09590000e-01,   0.00000000e+00,   1.19300000e+01, ...,
          2.10000000e+01,   3.93450000e+02,   6.48000000e+00],
       [  4.74100000e-02,   0.00000000e+00,   1.19300000e+01, ...,
          2.10000000e+01,   3.96900000e+02,   7.88000000e+00]]), 'fe
ature_names': array(['CRIM', 'ZN', 'INDUS', 'CHAS', 'NOX', 'RM', 'AG
E', 'DIS', 'RAD',
       'TAX', 'PTRATIO', 'B', 'LSTAT'],
      dtype='|S7'), 'DESCR': "Boston House Prices dataset\n\nNote
```


See *Supervised learning-4.ipynb > Example using python*

Example study using python

- Multiple linear regression
  - Target fit on age and lstat

```
In [14]: lm.fit (boston.data[:,(6,12)], boston.target)

Out[14]: LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normaliz
         e=False)

In [15]: lm.coef_

Out[15]: array([ 0.03454434, -1.03206856])

In [16]: lm.intercept_

Out[16]: 33.2227605317929
```

See *Supervised learning-4.ipynb > Example using python*

Example study using python

- Multiple linear regression
  - Fit on all 13 variables

```
In [17]: lm.fit (boston.data, boston.target)

Out[17]: LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normaliz
         e=False)

In [18]: lm.coef_

Out[18]: array([ -1.07170557e-01,   4.63952195e-02,   2.08602395e-02,
                  2.68856140e+00,  -1.77957587e+01,   3.80475246e+00,
                  7.51061703e-04,  -1.47575880e+00,   3.05655038e-01,
                 -1.23293463e-02,  -9.53463555e-01,   9.39251272e-03,
                 -5.25466633e-01])

In [19]: lm.intercept_

Out[19]: 36.491103280360925
```

See *Supervised learning-4.ipynb > Example using python*

Regression

- ✓ Performance evaluation

Prediction Accuracy Measures

- The prediction error for record i is defined as the difference between its actual y value and its predicted y value

$$e_i = y_i - \hat{y}_i$$

- Fit measures in classical regression modeling:
  - R^2 indicates how well data fit a statistical model

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

Prediction Accuracy Measures

- Fit measures in classical regression modeling:
  - Adjusted R^2 has been adjusted for the number of predictors. It increases only when the improve of model is more than one would expect to see by chance (p is the total number of explanatory variables)

$$Adjusted\ R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2/(n-p-1)}{\sum_{i=1}^{n}(y_i - \bar{y}_i)^2\,/(n-1)}$$

- Popular numerical measures of predictive accuracy:
  - MAE or MAD (mean absolute error/deviation) gives the magnitude of the average absolute error

$$MAE = \frac{1}{n\sum_{i=1}^{n}|e_i|}$$

QuantUniversity, LLC

Prediction Accuracy Measures

- Popular numerical measures of predictive accuracy:
  - Average error retains the sign of the errors. It gives an indication of whether the predictions are on average over- or under predicting the response

$$Average\ error = \frac{1}{n \sum_{i=1}^{n} e_i}$$

  - MAPE (mean absolute percentage error) gives a percentage score of how predictions deviate on average

$$MAPE = \frac{1}{n \sum_{i=1}^{n} |e_i / y_i|} \times 100\%$$

QuantUniversity, LLC

Prediction Accuracy Measures

- Popular numerical measures of predictive accuracy:
  - RMSE (root-mean-squared error) is similar to the standard error of estimate, except that it is computed on the validation data

$$RMSE = \sqrt{1/n \sum_{i=1}^{n} {e_i}^2}$$

  - Total SSE (total sum of squared errors)

$$SSE = \sum_{i=1}^{n} {e_i}^2$$

Example in R

- Use the linear regression model in prediction module
- Split the data into training and testing

```
library(MASS)
library(ISLR)
data(Boston)
#75% of the sample size
smp_size <- floor(0.75 * nrow(Boston))

#Set the seed to make your partition reproductible
set.seed(123)
train_ind <- sample(seq_len(nrow(Boston)), size = smp_size)

#Split the data into training and testing
train <- Boston[train_ind, ]
test <- Boston[-train_ind, ]
```

See *PredictiveEvaluation.R*

Example in R

- Build a linear regression model on the training data

```
#Fit a linear regression model
lm.fit = lm(medv ~ lstat, data = train)

#Summary of the fit
summary(lm.fit)
Call:
lm(formula = medv ~ lstat, data = train)

Residuals:
    Min      1Q  Median      3Q     Max
-15.126  -3.895  -1.343   1.720  24.525

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 34.43371    0.65112   52.88   <2e-16 ***
lstat       -0.94009    0.04536  -20.72   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.222 on 377 degrees of freedom
Multiple R-squared:  0.5325,    Adjusted R-squared:  0.5313
F-statistic: 429.5 on 1 and 377 DF,  p-value: < 2.2e-16
```

- R-squared is 0.5325 and adjusted r-squared is 0.5213

See *PredictiveEvaluation.R*

QuantUniversity, LLC

Example in R

- Run the model on the test set
- Get the measures of predictive accuracy

```
> #Measures of predictive accuracy
> library(forecast)
> pred = predict(lm.fit, test)
> accuracy(pred, train$medv)
                ME     RMSE      MAE       MPE     MAPE
Test set 0.7597872 12.15109 9.430894 -16.94604 48.42126
```

See *PredictiveEvaluation.R*

QuantUniversity, LLC

Example in python

- Prepare the data
- Split the data into training and testing

```
In [1]: import numpy as np

In [2]: import sklearn
        from sklearn import datasets

In [3]: boston = datasets.load_boston()

In [10]: X = boston.data[:,(6,12)]
         y = boston.target

In [11]: #Split the data into training and testing
         from sklearn.cross_validation import train_test_split
         X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=0)
```

See *PredictiveEvaluation.ipynb*

QuantUniversity, LLC

Example in python

- Build a linear regression model on training data

```
In [12]: #Fit a linear regression model
         from sklearn import linear_model

In [13]: lm=linear_model.LinearRegression()

In [14]: lm.fit (X_train, y_train)

Out[14]: LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=False)
```

- Get r-squared score of this model

```
In [16]: train_pred = lm.predict(X_train)

In [20]: #R-squared score of this model
         from sklearn.metrics import *
         r2_score(y_train, train_pred)

Out[20]: 0.57595593678602786
```

See *PredictiveEvaluation.ipynb*

QuantUniversity, LLC

Example in python

- Run the model on test data
- Get the measures of predictive accuracy

```
In [19]: test_pred = lm.predict(X_test)

In [21]: #Mean absolute error
         mean_absolute_error(y_test, test_pred)

Out[21]: 4.6347806564254075

In [22]: #Mean squared error
         mean_squared_error(y_test, test_pred)

Out[22]: 43.164388744314081

In [23]: #Median absolute error
         median_absolute_error(y_test, test_pred)

Out[23]: 3.368180280379427
```

See *PredictiveEvaluation.ipynb*

QuantUniversity, LLC

Regression

- ✓ Variable selection

Variable selection

- Involves using techniques to select the best features that add to the predictive power of the model.
- By applying variable selection the model has actual features and Irrelevant features removed from the model.
- Four algorithms exhaustive search, forward, backward selection, stepwise regression are explored here.

Regression

- **Exhaustive search:** This method evaluates all possible combinations of variables and chooses the best model based on the chosen criterion.
- **Forward selection:** Here the model adds one predictor at a time and continues until the time that adding another predictor is no longer statistically significant.
- **Backward selection:** It is the opposite of forward selection and all variables are included in the model to start with and variables are dropped one at a time till only the statistically significant variables remain.
- **Stepwise regression:** It combines both Forward and Backward eliminations and drops/adds variables based on their statistical significance.

Exhaustive search

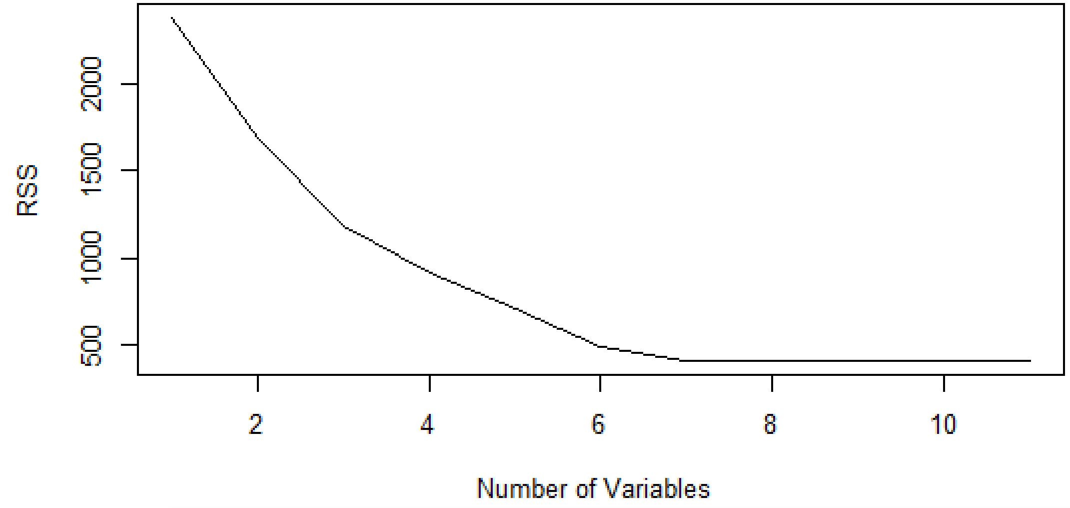
```
### Regression (Subset selection)
### Needed package and datasets
library(ISLR)
attach(Carseats)
Carseats=na.omit(Carseats) # Get rid of NAs
install.packages("leaps")
library(leaps)

##### Searching all subset models up to size 8 by default
regfit.full=regsubsets(Sales~.,data=Carseats)
summary(regfit.full)
##### Searching all subset models up to size number of variables
regfit.full=regsubsets (Sales~.,data=Carseats ,nvmax=11)
reg.summary =summary (regfit.full)
names(reg.summary)
reg.summary$rss
reg.summary$adjr2
```

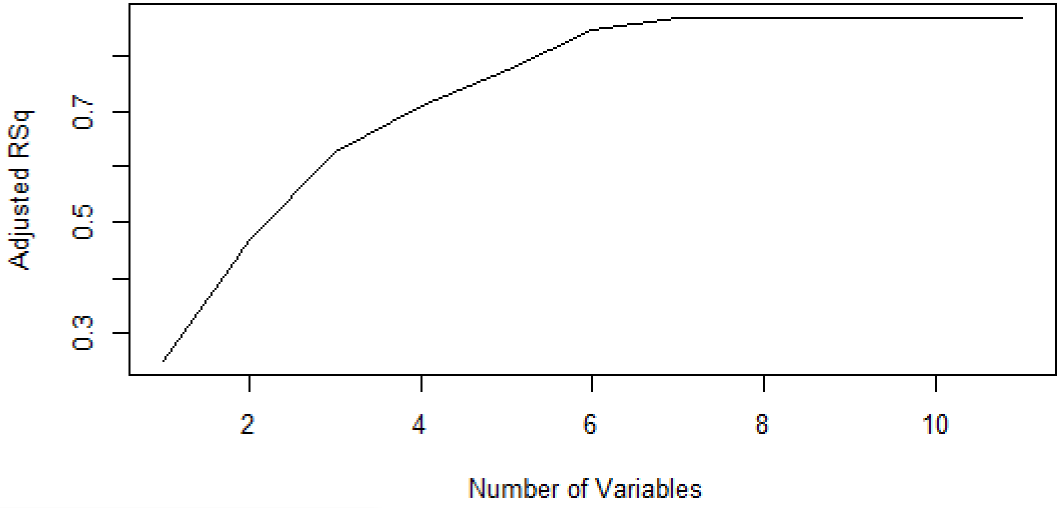
```
reg.summary$rss
[1] 2385.0818 1686.9145 1177.5148  915.1924  705.7155  484.0675  407.3869  405.7583
[9]  404.3142  403.1604  402.8335
reg.summary$adjr2
[1] 0.2486272 0.4672324 0.6271738 0.7094971 0.7754212 0.8455640 0.8696965 0.8698854
[9] 0.8700161 0.8700538 0.8698245
```

See *Regression_Carseats.R*

QuantUniversity, LLC

Exhaustive search





Let's pick 6 variables.
(Since there is no huge improvement in RSS after that)

Coefficients for the selected subset

```
  (Intercept)     CompPrice   Advertising         Price  ShelveLocGood
   6.88296902    0.09065313    0.11991086   -0.09567597     4.76861557
ShelveLocMedium          Age
   1.87668159   -0.04639023
```

See *Regression_Carseats.R*

QuantUniversity, LLC

Forward selection

```
#### Forward selection
regfit.fwd=regsubsets(Sales~.,data=Carseats ,nvmax=11, method="forward")
F=summary(regfit.fwd)
names(F)
F
F$rss
F$adjr2
coef(regfit.fwd,6)
```

Selected criterion and Coefficients for the selected subset

```
> F$rss
 [1] 2385.0818 1686.9145 1177.5148  915.1924  705.7155  484.0675  407.3869  405.7583
 [9]  404.3142  403.1604  402.8335
> F$adjr2
 [1] 0.2486272 0.4672324 0.6271738 0.7094971 0.7754212 0.8455640 0.8696965 0.8698854
 [9] 0.8700161 0.8700538 0.8698245
> coef(regfit.fwd,6)
    (Intercept)       CompPrice     Advertising           Price    ShelveLocGood
     6.88296902      0.09065313      0.11991086     -0.09567597       4.76861557
ShelveLocMedium             Age
     1.87668159     -0.04639023
```

See *Regression_Carseats.R*

QuantUniversity, LLC

Backward selection

```
#### Backward selection
regfit.bwd=regsubsets(Sales~.,data=Carseats ,nvmax=11, method="backward")
B=summary(regfit.bwd)
names(B)
B
B$rss
B$adjr2
coef(regfit.bwd,6)
```

Selected criterion and Coefficients for the selected subset

```
> B$rss
 [1] 2385.0818 1686.9145 1177.5148  915.1924  705.7155  484.0675  407.3869  405.7583
 [9]  404.3142  403.1604  402.8335
> B$adjr2
 [1] 0.2486272 0.4672324 0.6271738 0.7094971 0.7754212 0.8455640 0.8696965 0.8698854
 [9] 0.8700161 0.8700538 0.8698245
> coef(regfit.bwd,6)
    (Intercept)       CompPrice     Advertising           Price   ShelveLocGood
     6.88296902      0.09065313      0.11991086     -0.09567597      4.76861557
ShelveLocMedium             Age
     1.87668159     -0.04639023
```

See *Regression_Carseats.R*

QuantUniversity, LLC

Classification

- ✓ Logistic Regression

Logistic regression

- Logistic regression models the probability that response Y belongs to a particular category
- Use the logistic function to model the relationship between p(X) = Pr(Y = 1|X) (probability of Y belonging to category 1 given X) and X

$$p(X) = \frac{e^{\beta_0 + \beta_1 X}}{1 + e^{\beta_0 + \beta_1 X}}$$

QuantUniversity, LLC

Logistic regression

- After a bit of manipulation,

$$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X$$

- The left-hand side is called the log-odds or logit
- Increasing X by one unit changes the log odds by β_1, or equivalently it multiplies the odds by e^{β_1}

QuantUniversity, LLC

Multiple Logistic Regression

- Consider predicting a binary response using multiple predictors
- $X = (X_1, \ldots, X_p)$ are p predictors:

$$p(X) = \frac{e^{\beta_0+\beta_1 X_1+\cdots+\beta_p X_p}}{1+e^{\beta_0+\beta_1 X_1+\cdots+\beta_p X_p}}$$

- Odds: $\frac{p(X)}{1-p(X)}$
- Logit:

$$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$$

- Use the maximum likelihood method to estimate coefficients

QuantUniversity, LLC

Example study using R

- Data frame **Affairs** in AER package is based on a cross-sectional survey conducted by Psychology Today in 1969
- It contains 9 variables collected on 601 participants
- Variables includes how often some one had an extramarital affair during the past year, their gender, age, years married, whether they had children, their religiousness, education, occupation and self-rating of their marriage

QuantUniversity, LLC

Example study using R

```
> #Some descriptive statistics of Affairs
> data(Affairs, package="AER")
> summary(Affairs)
    affairs          gender         age         yearsmarried    children  religiousness  
 Min.   : 0.000   female:315   Min.   :17.50   Min.   : 0.125   no :171   Min.   :1.000  
 1st Qu.: 0.000   male  :286   1st Qu.:27.00   1st Qu.: 4.000   yes:430   1st Qu.:2.000  
 Median : 0.000                Median :32.00   Median : 7.000             Median :3.000  
 Mean   : 1.456                Mean   :32.49   Mean   : 8.178             Mean   :3.116  
 3rd Qu.: 0.000                3rd Qu.:37.00   3rd Qu.:15.000             3rd Qu.:4.000  
 Max.   :12.000                Max.   :57.00   Max.   :15.000             Max.   :5.000  
   education       occupation        rating     
 Min.   : 9.00   Min.   :1.000   Min.   :1.000  
 1st Qu.:14.00   1st Qu.:3.000   1st Qu.:3.000  
 Median :16.00   Median :5.000   Median :4.000  
 Mean   :16.17   Mean   :4.195   Mean   :3.932  
 3rd Qu.:18.00   3rd Qu.:6.000   3rd Qu.:5.000  
 Max.   :20.00   Max.   :7.000   Max.   :5.000  
```

- According to the statistics, 315 out of 601 respondents (52.4%) were female, 71.5% of participants had children

See *LogisticRegression.R*

Example study using R

- Transform affairs into a binary variable called **ynaffair**
- This factor ynaffair can be used as the outcome variable in further logistic regression model

```
> Affairs$ynaffair[Affairs$affairs > 0] <- 1
> Affairs$ynaffair[Affairs$affairs == 0] <- 0
> Affairs$ynaffair <- factor(Affairs$ynaffair,
+                            levels=c(0,1),
+                            labels=c("No","Yes"))
> table(Affairs$ynaffair)

 No Yes
451 150
```

- 75% of respondents reported not engaging in an infidelity

See *LogisticRegression.R*

Example study using R

- Use glm function to construct a logistic regression model for factor **ynaffairs** using all other variables

```
#Construct a logistic regression model for factor ynaffairs using all other variables
fit1 <- glm(ynaffair ~ gender + age + yearsmarried + children +
                religiousness + education + occupation + rating,
              data=Affairs, family=binomial(link="logit"))
summary(fit1)
```

Use family=binomial(link="logit") to specify a logistic regression model

```
Call:
glm(formula = ynaffair ~ gender + age + yearsmarried + children +
    religiousness + education + occupation + rating, family = binomial(link = "logit"),
    data = Affairs)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.5713  -0.7499  -0.5690  -0.2539   2.5191

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)    1.37726    0.88776   1.551 0.120807
gendermale     0.28029    0.23909   1.172 0.241083
age           -0.04426    0.01825  -2.425 0.015301 *
yearsmarried   0.09477    0.03221   2.942 0.003262 **
childrenyes    0.39767    0.29151   1.364 0.172508
religiousness -0.32472    0.08975  -3.618 0.000297 ***
education      0.02105    0.05051   0.417 0.676851
occupation     0.03092    0.07178   0.431 0.666630
rating        -0.46845    0.09091  -5.153 2.56e-07 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 675.38  on 600  degrees of freedom
Residual deviance: 609.51  on 592  degrees of freedom
AIC: 627.51

Number of Fisher Scoring iterations: 4
```

See *LogisticRegression.R*

Example study using R

```
Coefficients:
               Estimate Std. Error z value Pr(>|z|)
(Intercept)     1.37726    0.88776   1.551 0.120807
gendermale      0.28029    0.23909   1.172 0.241083
age            -0.04426    0.01825  -2.425 0.015301 *
yearsmarried    0.09477    0.03221   2.942 0.003262 **
childrenyes     0.39767    0.29151   1.364 0.172508
religiousness  -0.32472    0.08975  -3.618 0.000297 ***
education       0.02105    0.05051   0.417 0.676851
occupation      0.03092    0.07178   0.431 0.666630
rating         -0.46845    0.09091  -5.153 2.56e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- According to the p-values for the regression coefficients, variables (gender, presence of children, education, and occupation) do not make a significant contribution to the model (can't reject the hypothesis that the parameters are 0)

See *LogisticRegression.R*

Example study using R

- Construct a logistic regression model for factor ynaffairs using age, yearsmarried, religiousness and rating

```
#Construct a logistic regression model for factor ynaffairs
#Using age, yearsmarried, religiousness and rating
fit2 <- glm(ynaffair ~ age + yearsmarried + religiousness + rating,
            data=Affairs, family=binomial(link="logit"))
summary(fit2)
```

```
Call:
glm(formula = ynaffair ~ age + yearsmarried + religiousness +
    rating, family = binomial(link = "logit"), data = Affairs)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.6278  -0.7550  -0.5701  -0.2624   2.3998

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)    1.93083    0.61032   3.164 0.001558 **
age           -0.03527    0.01736  -2.032 0.042127 *
yearsmarried   0.10062    0.02921   3.445 0.000571 ***
religiousness -0.32902    0.08945  -3.678 0.000235 ***
rating        -0.46136    0.08884  -5.193 2.06e-07 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 675.38  on 600  degrees of freedom
Residual deviance: 615.36  on 596  degrees of freedom
AIC: 625.36

Number of Fisher Scoring iterations: 4
```

See *LogisticRegression.R*

Example study using R

- Logistic model parameters

```
> coef(fit2)
(Intercept)         age  yearsmarried religiousness        rating
 1.93083017  -0.03527112   0.10062274   -0.32902386   -0.46136144
```

$$\log(\text{odds}) = 1.931 - 0.035\text{age} + 0.100\text{yearsmarried} - 0.329\text{religiousness} - 0.461\text{rating}$$

- Use predict() function to predict the probabilities of the outcome

```
> newdata <- data.frame(rating=mean(Affairs$rating),
+                       age=mean(Affairs$age),
+                       yearsmarried=mean(Affairs$yearsmarried),
+                       religiousness=mean(Affairs$religiousness))
> newdata
   rating      age yearsmarried religiousness
1 3.93178 32.48752     8.177696      3.116473
> prob <- predict(fit2, newdata=newdata, type="response")
> prob
        1
0.2259112
```

See *LogisticRegression.R*

QuantUniversity, LLC

Example study using python

- Export Affairs dataset to an Excel spreadsheet

```
#Export dataset
data(Affairs, package="AER")
write.csv(Affairs,"Affairs.csv")
```

- Import data in python

```
In [45]: #Import Affairs from csv file
         import pandas as pd
         import numpy as np
         Affairs = pd.read_csv("Affairs.csv", header=0)
         del Affairs['Unnamed: 0']
         print(Affairs.describe())
```

```
          affairs         age  yearsmarried  religiousness   education  \
count  601.000000  601.000000    601.000000     601.000000  601.000000
mean     1.455907   32.487521      8.177696       3.116473   16.166389
std      3.298758    9.288762      5.571303       1.167509    2.402555
min      0.000000   17.500000      0.125000       1.000000    9.000000
25%      0.000000   27.000000      4.000000       2.000000   14.000000
50%      0.000000   32.000000      7.000000       3.000000   16.000000
75%      0.000000   37.000000     15.000000       4.000000   18.000000
max     12.000000   57.000000     15.000000       5.000000   20.000000

       occupation      rating
count  601.000000  601.000000
mean     4.194676    3.931780
std      1.819443    1.103179
min      1.000000    1.000000
25%      3.000000    3.000000
50%      5.000000    4.000000
75%      6.000000    5.000000
max      7.000000    5.000000
```

See *LogisticRegression.R*

QuantUniversity, LLC

Example study using python

- Transform affairs into a dichotomous factor called ynaffair

In [18]:
```
#Transform affairs into a binominal factor called ynaffair
Affairs['ynaffair'] = (Affairs.affairs > 0).astype(int)
print(Affairs.head(10))
```

```
   affairs  gender  age  yearsmarried children  religiousness  education  \
0        0    male   37         10.00       no              3         18
1        0  female   27          4.00       no              4         14
2        0  female   32         15.00      yes              1         12
3        0    male   57         15.00      yes              5         18
4        0    male   22          0.75       no              2         17
5        0  female   32          1.50       no              2         17
6        0  female   22          0.75       no              2         12
7        0    male   57         15.00      yes              2         14
8        0  female   32         15.00      yes              4         16
9        0    male   22          1.50       no              4         14

   occupation  rating  ynaffair
0           7       4         0
1           6       4         0
2           1       4         0
3           6       5         0
4           6       3         0
5           5       5         0
6           1       3         0
7           4       4         0
8           1       2         0
9           4       5         0
```

See *LogisticRegression.ipynb*

QuantUniversity, LLC

Example study using python

```
In [20]: Affairs.groupby('ynaffair').mean()
```

Out[20]:

	affairs	age	yearsmarried	religiousness	education	occupation	rating
ynaffair							
0	0.000000	32.18071	7.727279	3.203991	16.139690	4.155211	4.093126
1	5.833333	33.41000	9.531947	2.853333	16.246667	4.313333	3.446667

- On average, respondents who have affairs rate their marriages lower

```
In [21]: print pd.crosstab(Affairs['ynaffair'], Affairs['rating'], rownames=['ynaffair'])

rating    1   2   3    4    5
ynaffair
0         8  33  66  146  198
1         8  33  27   48   34
```

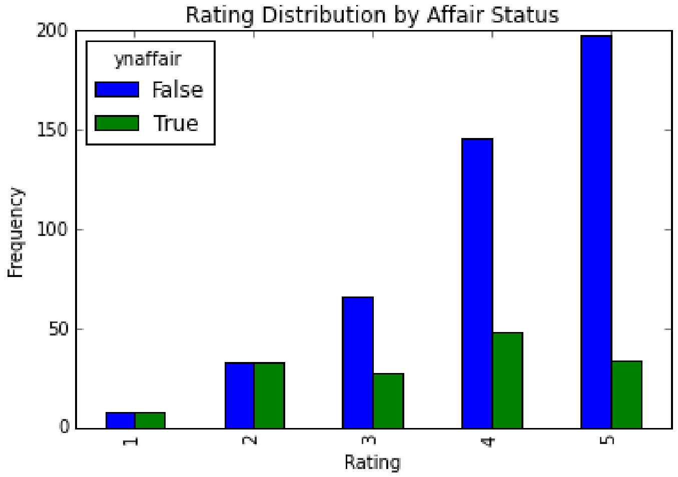
See *LogisticRegression.ipynb*

QuantUniversity, LLC

Example study using python

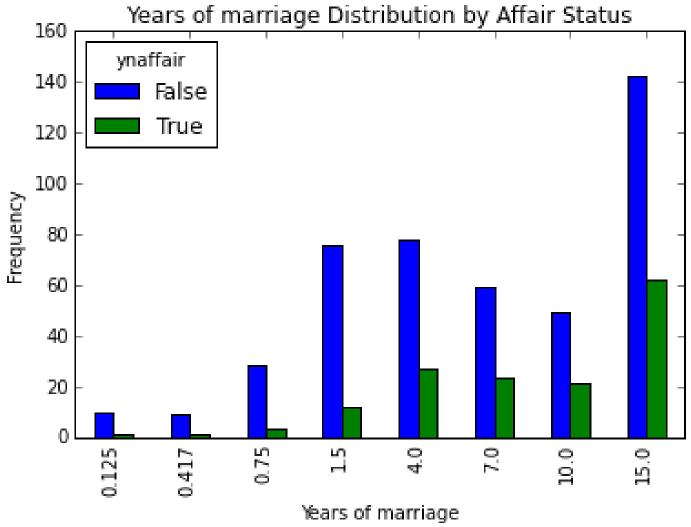
In [24]:
```
import matplotlib.pyplot as plt
%matplotlib inline
#Barplot of rating grouped by ynaffair
pd.crosstab(Affairs.rating, Affairs.ynaffair.astype(bool)).plot(kind='bar')
plt.title('Rating Distribution by Affair Status')
plt.xlabel('Rating')
plt.ylabel('Frequency')
```

Out[24]: `<matplotlib.text.Text at 0x1fc8d978>`



In [25]:
```
#Barplot of years of marriage grouped by ynaffair
pd.crosstab(Affairs.yearsmarried, Affairs.ynaffair.astype(bool)).plot(kind='bar')
plt.title('Years of marriage Distribution by Affair Status')
plt.xlabel('Years of marriage')
plt.ylabel('Frequency')
```

Out[25]: `<matplotlib.text.Text at 0x1fe74e48>`



See *LogisticRegression.ipynb*

QuantUniversity, LLC

Example study using python

- Create dummy variables for gender and children

```
In [51]: #Create dummy variables for gender and children
         dummy_gender = pd.get_dummies(Affairs['gender'], prefix='gender')
         dummy_children = pd.get_dummies(Affairs['children'], prefix='children')
         Affairs = Affairs.join(dummy_gender)
         Affairs = Affairs.join(dummy_children)
         print Affairs.head()
```

```
   affairs  gender  age  yearsmarried children  religiousness  education  \
0        0    male   37         10.00       no              3         18
1        0  female   27          4.00       no              4         14
2        0  female   32         15.00      yes              1         12
3        0    male   57         15.00      yes              5         18
4        0    male   22          0.75       no              2         17

   occupation  rating  ynaffair  gender_female  gender_male  children_no  \
0           7       4         0              0            1            1
1           6       4         0              1            0            1
2           1       4         0              1            0            0
3           6       5         0              0            1            0
4           6       3         0              0            1            1

   children_yes
0             0
1             0
2             1
3             1
4             0
```

See *LogisticRegression.ipynb*

Example study using python

- Set response variable y and predictors X

```
In [8]: #Transform gender variable, 1 represents male, 0 represents female
        #Transform children variable, 1 represents yes, 0 represents no
        Affairs['gender_1'] = (Affairs.gender == 'male').astype(int)
        Affairs['ynchildren'] = (Affairs.children == 'yes').astype(int)
        print Affairs.head()

   affairs  gender  age  yearsmarried children  religiousness  education  \
0        0    male   37         10.00       no              3         18
1        0  female   27          4.00       no              4         14
2        0  female   32         15.00      yes              1         12
3        0    male   57         15.00      yes              5         18
4        0    male   22          0.75       no              2         17

   occupation  rating  ynaffair  gender_1  ynchildren
0           7       4         0         1           0
1           6       4         0         0           0
2           1       4         0         0           1
3           6       5         0         1           1
4           6       3         0         1           0
```

```
In [9]: #Select response y and predictors X
        y = Affairs['ynaffair']
        cols_to_keep = ['age', 'yearsmarried', 'religiousness', 'education', 'occupation', 'rating',
                        'gender_1', 'ynchildren']
        X = Affairs[cols_to_keep]
```

```
In [10]: #Flatten y into a 1-D array
         y = np.ravel(y)
```

See *LogisticRegression.ipynb*

Example study using python

- Use LogisticRegression() function in scikit-learn module to perform logistic regression model

```
In [11]: #Instantiate a logistic regression model
         #Fit with X and y
         from sklearn.linear_model import LogisticRegression
         model = LogisticRegression()
         model = model.fit(X, y)

In [12]: #Check the accuracy of the model
         model.score(X, y)

Out[12]: 0.76039933444259566

In [13]: model.coef_

Out[13]: array([[-0.03912426,  0.09021842, -0.30526004,  0.04443947,  0.02617262,
                 -0.44644188,  0.23156946,  0.40074224]])
```

See *LogisticRegression.ipynb*

QuantUniversity, LLC

Example study using python

- Otherwise, logit function in statsmodels module can be used to perform logistic regression model

In [14]:
```
import statsmodels.api as sm
from statsmodels.formula.api import logit, probit, poisson, ols
logit = sm.Logit(Affairs['ynaffair'], Affairs[cols_to_keep])
affair_mod = logit.fit()
print(affair_mod.summary())
```

```
Optimization terminated successfully.
         Current function value: 0.509098
         Iterations 6
                           Logit Regression Results
==============================================================================
Dep. Variable:               ynaffair   No. Observations:                  601
Model:                          Logit   Df Residuals:                      593
Method:                           MLE   Df Model:                            7
Date:                Thu, 12 Nov 2015   Pseudo R-squ.:                 0.09393
Time:                        11:14:53   Log-Likelihood:                -305.97
converged:                       True   LL-Null:                       -337.69
                                        LLR p-value:                 3.092e-11
=================================================================================
                  coef    std err          z      P>|z|      [95.0% Conf. Int.]
---------------------------------------------------------------------------------
age            -0.0333      0.017     -2.012      0.044        -0.066    -0.001
yearsmarried    0.0827      0.031      2.666      0.008         0.022     0.143
religiousness  -0.2873      0.086     -3.338      0.001        -0.456    -0.119
education       0.0742      0.037      1.981      0.048         0.001     0.148
occupation      0.0193      0.071      0.271      0.786        -0.120     0.159
rating         -0.4257      0.086     -4.921      0.000        -0.595    -0.256
gender_1        0.1917      0.233      0.824      0.410        -0.264     0.648
ynchildren      0.4789      0.290      1.654      0.098        -0.089     1.046
=================================================================================
```

See *LogisticRegression.ipynb*

QuantUniversity, LLC

Example study using python

	coef	std err	z	P>|z|	[95.0% Conf. Int.]	
age	-0.0443	0.018	-2.425	0.015	-0.080	-0.008
yearsmarried	0.0948	0.032	2.942	0.003	0.032	0.158
religiousness	-0.3247	0.090	-3.618	0.000	-0.501	-0.149
education	0.0211	0.051	0.417	0.677	-0.078	0.120
occupation	0.0309	0.072	0.431	0.667	-0.110	0.172
rating	-0.4685	0.091	-5.153	0.000	-0.647	-0.290
gender_female	0.7180	nan	nan	nan	nan	nan
gender_male	0.9983	nan	nan	nan	nan	nan
children_no	0.6593	nan	nan	nan	nan	nan
children_yes	1.0570	nan	nan	nan	nan	nan

- According to the p-value, only age, yearsmarried, religiousness and rating make significant contributions to the model.

See *LogisticRegression.ipynb*

Classification

- ✓ Performance evaluation

Classification matrix

- Classification matrix (confusion matrix) summarizes the correct and incorrect classifications that a classifier produced for a certain dataset
- Classification matrix gives estimates of the true classification and misclassification rates
- Compute the classification matrix from the validation data to get an honest estimate

Classification matrix

- Consider a two-class case with classes C_0 and C_1
- Classification matrix:

	Predicted Class	
Actual Class	C_0	C_1
C_0	$n_{0,0}$= number of C_0 cases classified correctly	$n_{0,1}$= number of C_0 cases classified incorrectly as C_1
C_1	$n_{1,0}$= number of C_1 cases classified incorrectly as C_0	$n_{1,1}$= number of C_1 cases classified correctly

QuantUniversity, LLC

Accuracy Measures

- Estimated misclassification rate (overall error rate) is a main accuracy measure

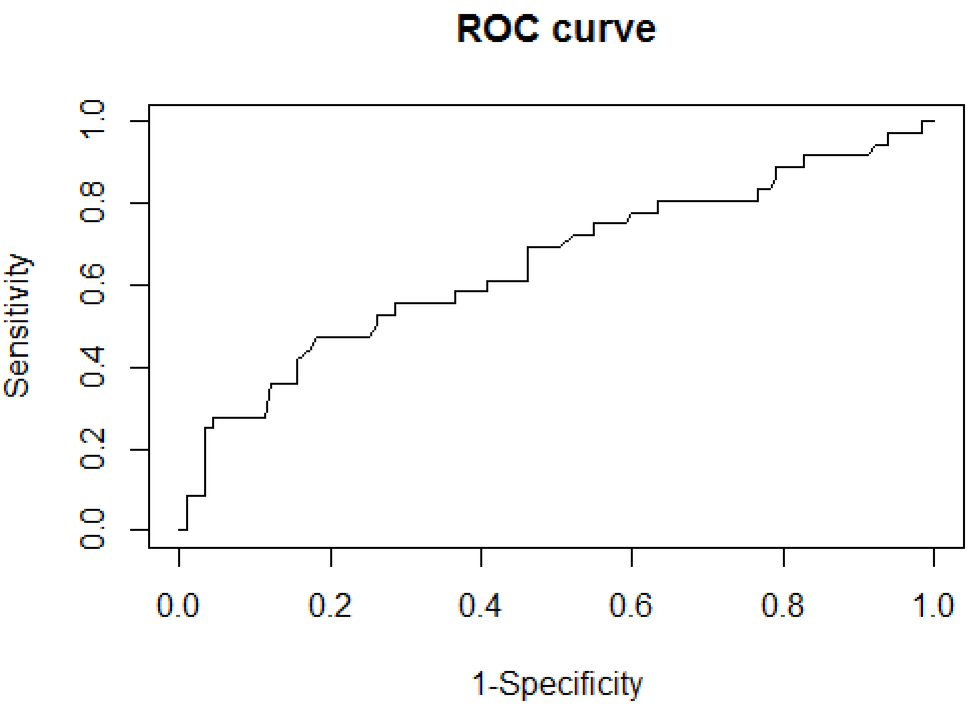
$$err = \frac{n_{0,1} + n_{1,0}}{n_{0,0} + n_{0,1} + n_{1,0} + n_{1,1}} = \frac{n_{0,1} + n_{1,0}}{n}$$

- Overall accuracy:

$$Accuracy = 1 - err = \frac{n_{0,0} + n_{1,1}}{n}$$

QuantUniversity, LLC

ROC Curve



- The ROC curve plots the pairs {sensitivity, 1-specificity} as the cutoff value increases from 0 and 1
  - **Sensitivity** (also called the **true positive rate**, or the recall in some fields) measures the proportion of positives that are correctly identified (e.g., the percentage of sick people who are correctly identified as having the condition).
  - **Specificity** (also called the **true negative rate**) measures the proportion of negatives that are correctly identified as such (e.g., the percentage of healthy people who are correctly identified as not having the condition).
- Better performance is reflected by curves that are closer to the top left corner

Lift Charts

- **Lift** is a measure of the effectiveness of a predictive model calculated as the ratio between the results obtained with and without the predictive model.
- Cumulative gains and **lift charts** are visual aids for measuring model performance. Both **charts** consist of a **lift**curve and a baseline.

Example in R

- Use the logistic regression model in classification module
- Split the data into training and testing

```
data(Affairs, package="AER")
#Transform affairs into a dichotomous factor
Affairs$ynaffair[Affairs$affairs > 0] <- 1
Affairs$ynaffair[Affairs$affairs == 0] <- 0
Affairs$ynaffair <- factor(Affairs$ynaffair,
                           levels=c(0,1),
                           labels=c("No","Yes"))
table(Affairs$ynaffair)

#75% of the sample size
smp_size <- floor(0.75 * nrow(Affairs))

#Set the seed to make your partition reproductible
set.seed(123)
train_ind <- sample(seq_len(nrow(Affairs)), size = smp_size)

#Split the data into training and testing
train <- Affairs[train_ind, ]
test <- Affairs[-train_ind, ]
```

See *ClassificationEvaluation.R*

QuantUniversity, LLC

Example in R

- Build a logistic regression model on the training data

```
#Fit a logistic regression model
fit <- glm(ynaffair ~ age + yearsmarried + religiousness + rating,
            data=train, family=binomial(link="logit"))
summary(fit)
Call:
glm(formula = ynaffair ~ age + yearsmarried + religiousness +
    rating, family = binomial(link = "logit"), data = train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.6481  -0.7656  -0.5562   0.7715   2.3462

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)    2.29065    0.72072   3.178  0.00148 **
age           -0.02131    0.01971  -1.081  0.27958
yearsmarried   0.06497    0.03342   1.944  0.05186 .
religiousness -0.46848    0.10562  -4.436 9.18e-06 ***
rating        -0.47887    0.10427  -4.593 4.37e-06 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 509.37  on 449  degrees of freedom
Residual deviance: 460.74  on 445  degrees of freedom
AIC: 470.74

Number of Fisher Scoring iterations: 4
```

See *ClassificationEvaluation.R*

Example in R

- Run the model on the test set with cutoff value = 0.5
- Generate classification matrix using confusionMatrix() function in caret package

```
#Run the model on the test set
test.probs <- predict(fit, test, type='response')
pred <- rep("No",length(test.probs))

#Set the cutoff value =0.5
pred[test.probs>=0.5] <- "Yes"

#Classification matrix
library(caret)
confusionMatrix(test$ynaffair, pred)
```

```
Confusion Matrix and Statistics

          Reference
Prediction  No Yes
       No  111   4
       Yes  27   9

               Accuracy : 0.7947
                 95% CI : (0.7214, 0.856)
    No Information Rate : 0.9139
    P-Value [Acc > NIR] : 1

                  Kappa : 0.2757
 Mcnemar's Test P-Value : 7.772e-05

            Sensitivity : 0.8043
            Specificity : 0.6923
         Pos Pred Value : 0.9652
         Neg Pred Value : 0.2500
             Prevalence : 0.9139
         Detection Rate : 0.7351
   Detection Prevalence : 0.7616
      Balanced Accuracy : 0.7483

       'Positive' Class : No
```

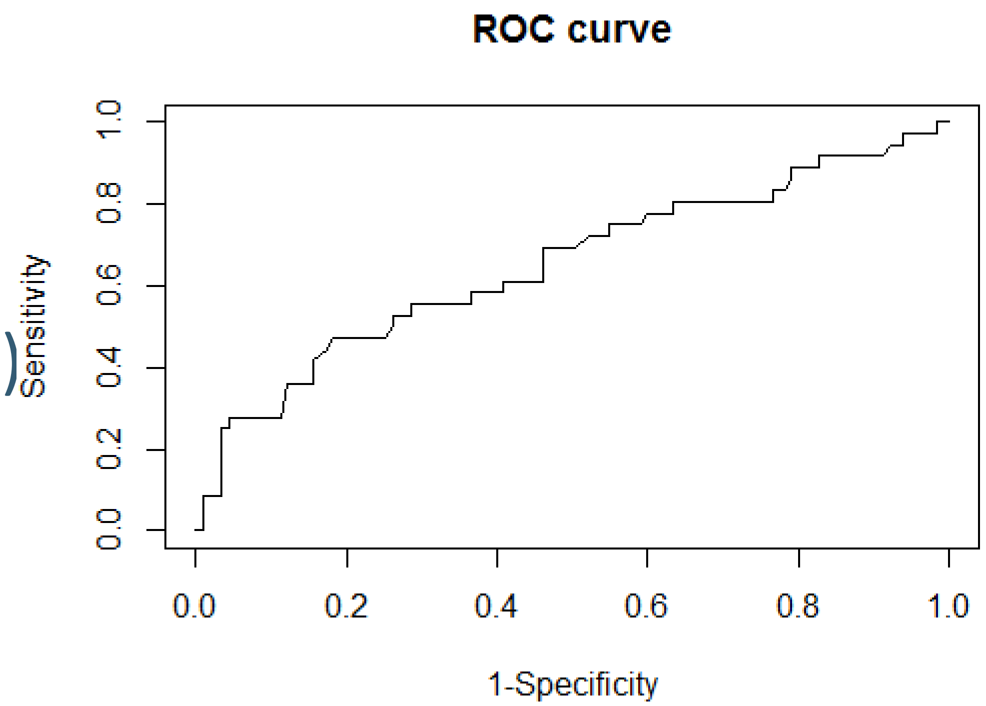
See *ClassificationEvaluation.R*

Example in R

- Generate ROC curve using ROCR package

```
#ROC curve
library(ROCR)
prediction <- prediction(test.probs, test$ynaffair)
performance <- performance(prediction, measure = "tpr", x.measure = "fpr")
plot(performance, main="ROC curve", xlab="1-Specificity", ylab="Sensitivity")
```

- The perfect ROC curve would yield a point in the upper left corner (0,1)



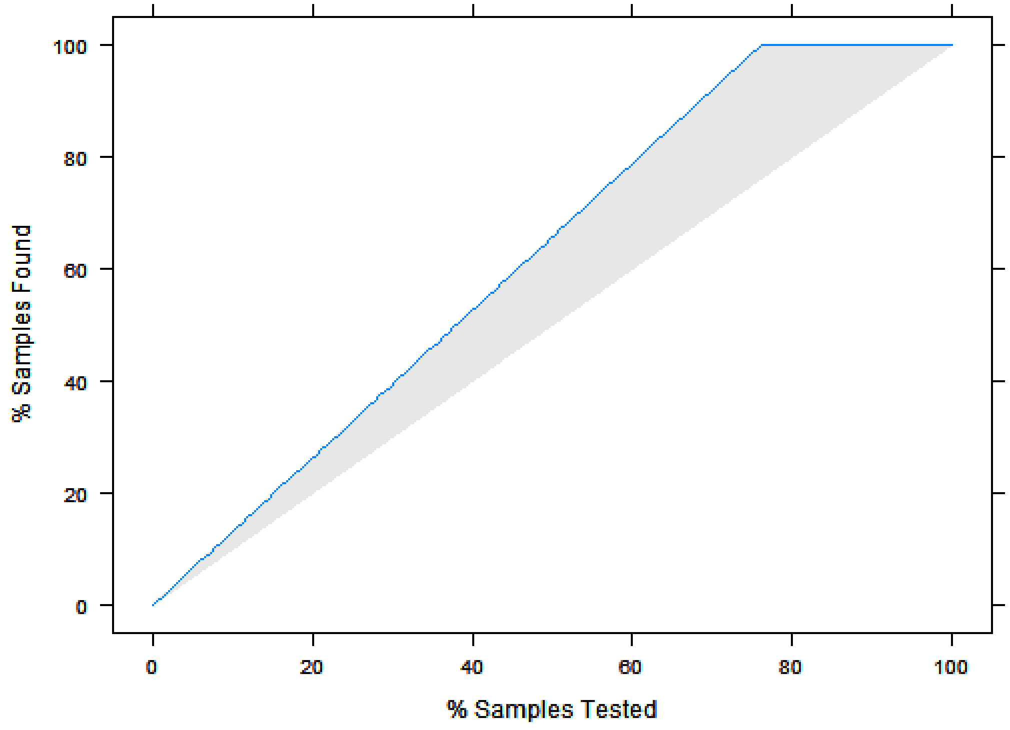
See *ClassificationEvaluation.R*

QuantUniversity, LLC

Example in R

- Generate cumulative lift curve using lift() function in caret package

```
#Lift curve
test$probs=test.probs
test$prob=sort(test$probs,decreasing = T)
lift <- lift(ynaffair ~ prob, data = test)
lift
xyplot(lift,plot = "gain")
```



See *ClassificationEvaluation.R*

QuantUniversity, LLC

Example in python

- Prepare the data

```
In [15]: #Import Affairs from csv file
         import pandas as pd
         import numpy as np
         import matplotlib.pyplot as plt
         Affairs = pd.read_csv("Affairs.csv", header=0)
         del Affairs['Unnamed: 0']

In [16]: #Transform affairs into a binary factor called ynaffair
         Affairs['ynaffair'] = (Affairs.affairs > 0).astype(int)

In [17]: #Transform gender variable, 1 represents male, 0 represents female
         #Transform children variable, 1 represents yes, 0 represents no
         Affairs['gender_1'] = (Affairs.gender == 'male').astype(int)
         Affairs['ynchildren'] = (Affairs.children == 'yes').astype(int)

In [4]:  #Select response y and predictors X
         y = Affairs['ynaffair']
         cols_to_keep = ['age', 'yearsmarried', 'religiousness', 'education', 'occupation', 'rating',
                         'gender_1', 'ynchildren']
         X = Affairs[cols_to_keep]

In [5]:  #Flatten y into a 1-D array
         y = np.ravel(y)
```

See *ClassificationEvaluation.ipynb*

QuantUniversity, LLC

Example in python

- Spilt the data into training and testing
- Build a logistic regression model on training data

In [7]:
```
#Split the data into training and testing
from sklearn.cross_validation import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=0)
```

In [8]:
```
#Build logistic regression model on training data
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
model = model.fit(X_train, y_train)
```

See *ClassificationEvaluation.ipynb*

QuantUniversity, LLC

Example in python

- Run the model on the test set
- Generate classification matrix using confusion_matrix() function in sklearn.metrics

```
In [9]:  #Run the model on the test set
         y_pred = model.predict(X_test)

In [11]: #Compute confusion matrix
         from sklearn.metrics import confusion_matrix
         cm = confusion_matrix(y_test, y_pred)
         print(cm)

         [[110   8]
          [ 27   6]]
```

See *ClassificationEvaluation.ipynb*

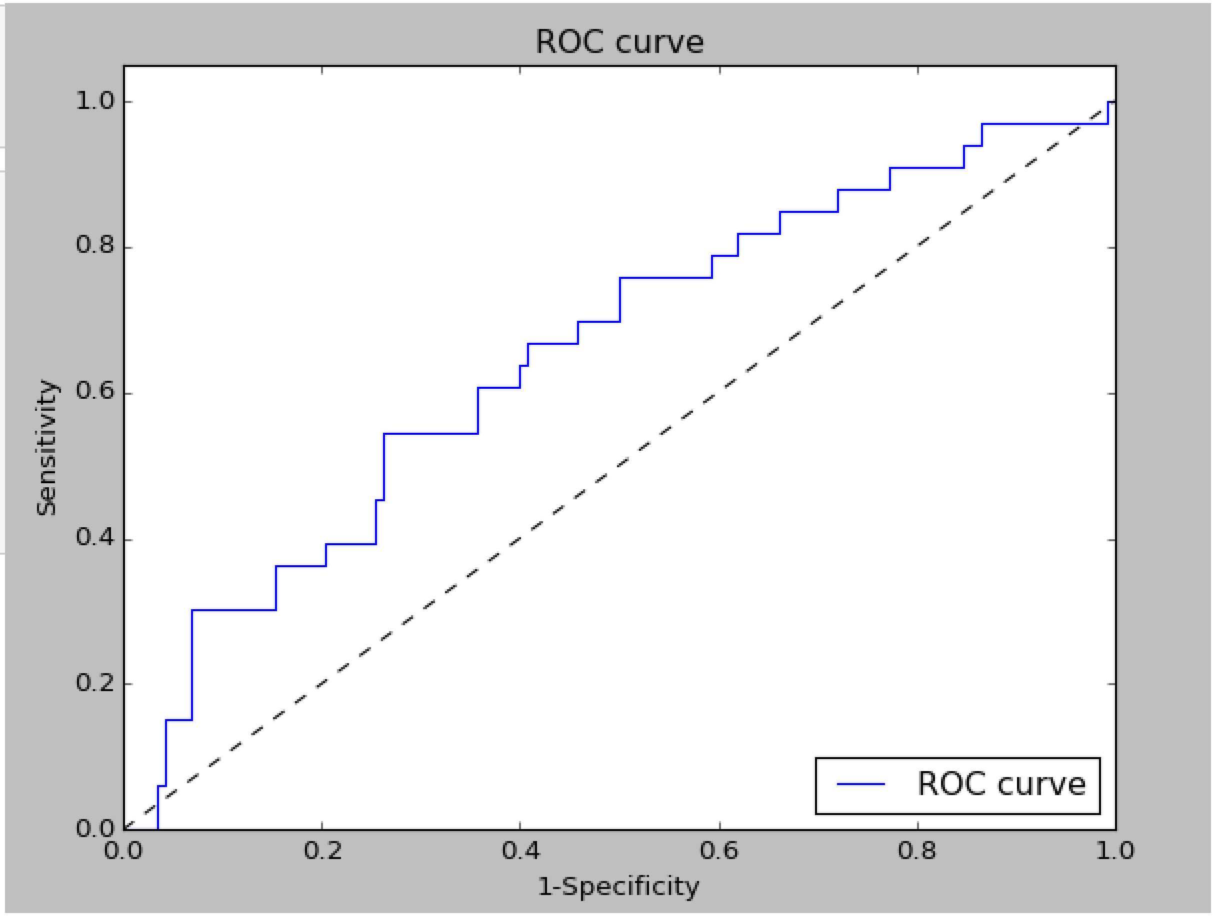
QuantUniversity, LLC

Example in python

- Generate ROC curve using roc_curve() function in sklearn.metrics

In [14]:
```
#Compute FPR and TPR
from sklearn.metrics import roc_curve
preds = model.predict_proba(X_test)[:,1]
fpr, tpr, _ = roc_curve(y_test, preds)
```

In [*]:
```
#Plot ROC curve
import matplotlib.pyplot as plt
plt.figure()
plt.plot(fpr, tpr, label='ROC curve')
plt.plot([0, 1], [0, 1], 'k--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('1-Specificity')
plt.ylabel('Sensitivity')
plt.title('ROC curve')
plt.legend(loc="lower right")
plt.show()
```



See *ClassificationEvaluation.ipynb*

QuantUniversity, LLC

Reference:

- scikit-learn.org
- Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. An introduction to statistical learning. Springer, 2013

Q&A

QuantUniversity, LLC

www.quantuniversity.com

Thank you!

Contact

Sri Krishnamurthy, CFA, CAP
Founder and CEO
QuantUniversity LLC.

LinkedIn srikrishnamurthy

www.QuantUniversity.com