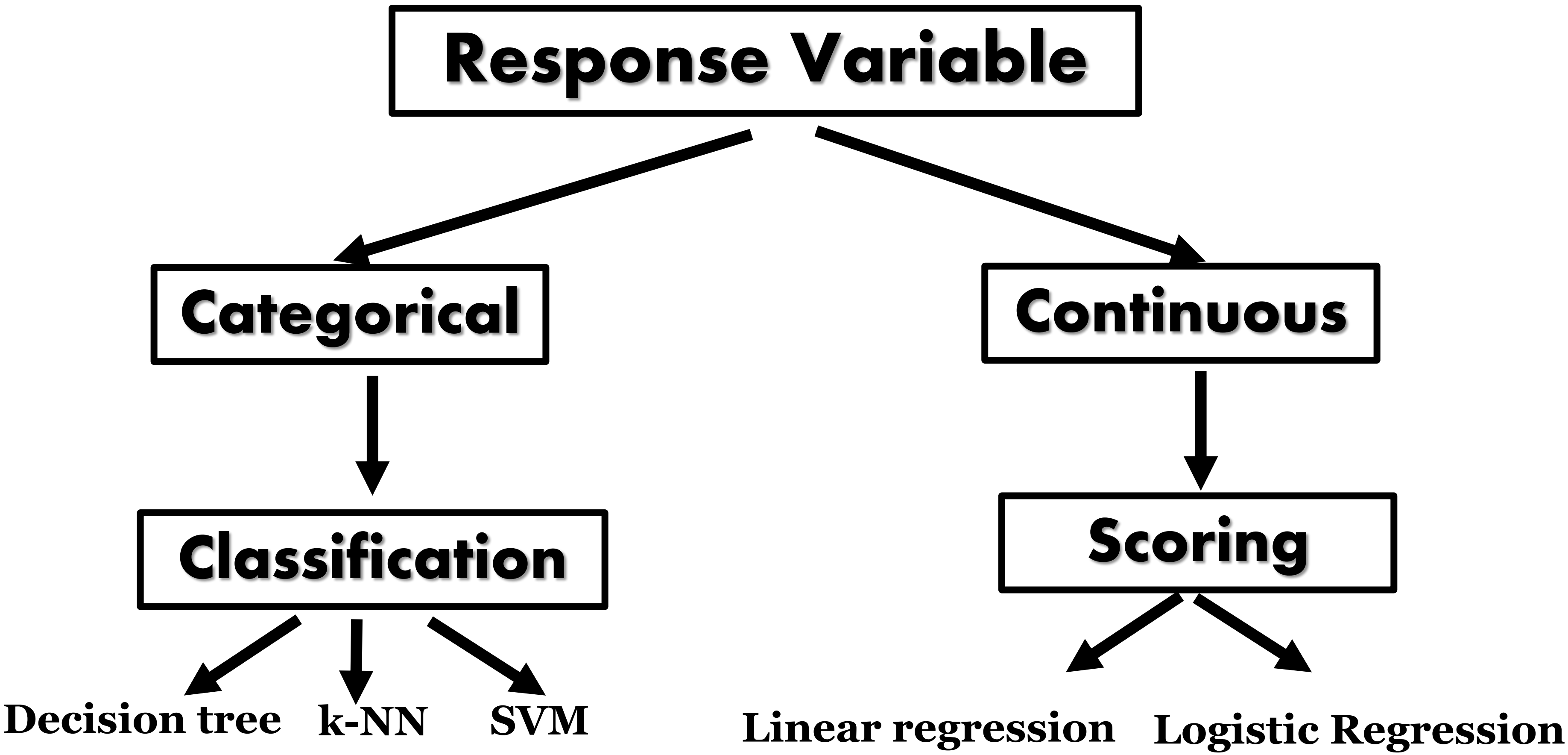
Model Evaluation Metrics: Scoring (Regression) Methods

Nagiza F. Samatova, samatova@csc.ncsu.edu
Professor, Department of Computer Science
North Carolina State University

Senior Scientist, Computer Science & Mathematics Division
Oak Ridge National Laboratory

Classification vs. Scoring Methods



Things to Watch for (Packages: car and gvlma)

- **Independence:** The observations (rows) must be independent of each other
- **Linearity**: Relationship between the Response (Y) and the predictors (X's) must be linear in terms of the model parameters (β's):
  - Use **crPlots()** in the car pkg for systematic departures from linear model
  - Think how to transform X's to achieve this
- **Normality**: The Response (Y) in a linear regression model must be from the normal (Gaussian distribution):
  - Test for normality using **qqPlot()** in the **car** pkg
  - Check **gvlma** package for global test for linear model assumptions
- **Error/Noise**: The Residuals (predicted – actual) must come from the normal distribution $\mathcal{N}(0,1)$ and should not be autocorrelated (**durbinWatsonTest()**)
- **Homoscedasticity:** The errors (residuals) must be structured:
  - The **car** package provides **ncvTest()** to the hypothesis of constant error variance against the alternative that the error variance changes with the level of fitted values: significant result suggest heteroscedasticity
- **Multicollinearity**: Test for absence of multicollinearity (**vif()** in the **car** pkg)
- **Sensitivity to outliers**: Sensitivity to outliers may affect model performance:
  - Use **outlierTest()** in the car pkg to identify ***high-leverage observations*** and ***influential observations***
- **Model complexity**: Feature selection (forward, backward, stepwise regression) and significant coefficient may guide towards models with reduced complexity

Linear vs. Nonlinear Model

Response = Linear Combination of Explanatory Variables

Function of the Response = Linear Combination of Explanatory Variables

linear model
in terms of $\beta's$, unknown parameters
glm()

$$\mu_Y = \boldsymbol{\beta_0} + \boldsymbol{\beta_1} X_1 + \boldsymbol{\beta_2} X_2 + \ldots + \boldsymbol{\beta_q} X_q$$

$$\mu_Y = \boldsymbol{\beta_0} + \boldsymbol{\beta_1} X_1 + \boldsymbol{\beta_2} X_1^2 + \underline{\boldsymbol{\beta_3} X_1^3}$$

$$g(\mu_Y) = \boldsymbol{\beta_0} + \boldsymbol{\beta_1} X_1 + \underline{\boldsymbol{\beta_2} X_1 X_2}$$ coupled predictors

polynomial predictors

$$g(\mu_Y) = \boldsymbol{\beta_0} + \boldsymbol{\beta_1} X_1 + \underline{\boldsymbol{\beta_2} \exp^{X_1}}$$ transformed predictors

$$g(\mu_Y) = \boldsymbol{\beta_0} + \underline{\boldsymbol{\beta_1} log X_1} + \underline{\boldsymbol{\beta_2} \sin(X_2)}$$

Known: $X_1, X_2, \ldots,$
Unknown: $\boldsymbol{\beta_0}, \boldsymbol{\beta_1}, \ldots$

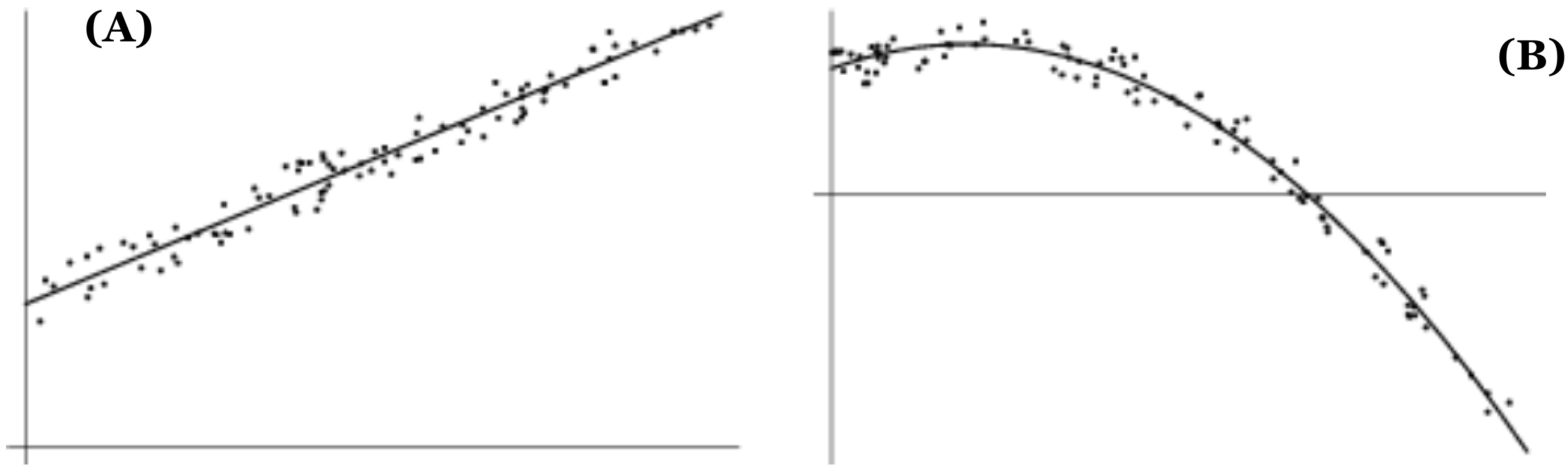
$\boxed{\text{The equation is linear in the parameters } (\beta_0, \beta_1, \ldots, \beta_q)}$

***non*-linear model**
in terms of $\beta's$, unknown parameters (**nls()**)

$$\mu_Y = \boldsymbol{\beta_0} + \boldsymbol{\beta_1} \boldsymbol{exp}^{\frac{\boldsymbol{X}}{\boldsymbol{\beta_2}}}$$

Regression: Linear Relationship between Predictors and Response

Hats: Estimates $\boxed{\hat{Y} = \widehat{\beta_0} + \widehat{\beta_1} * predictor + error}$ *line in 2-dimensions*



Q1: Do both figures depict linear regression model fitting?
Q2: How many predictors does model (B) depend on?
Q3: Is relationship between the response and the predictor in (B) linear? Why or why not?
Q4: What type of transformation one would apply to the predictor in (B) to get linear relation between the transformed feature and the predictor?
Q5: Does the regression model/method discover itself the type of transformation(s) needed for the predictors to get the linear fitting?

Residuals for Scoring Models

Residual = **Difference between**
Actual Value of Response Variable and
Predicted Value by the Scoring Model

Performance Measure = **Function** (**Residuals**)

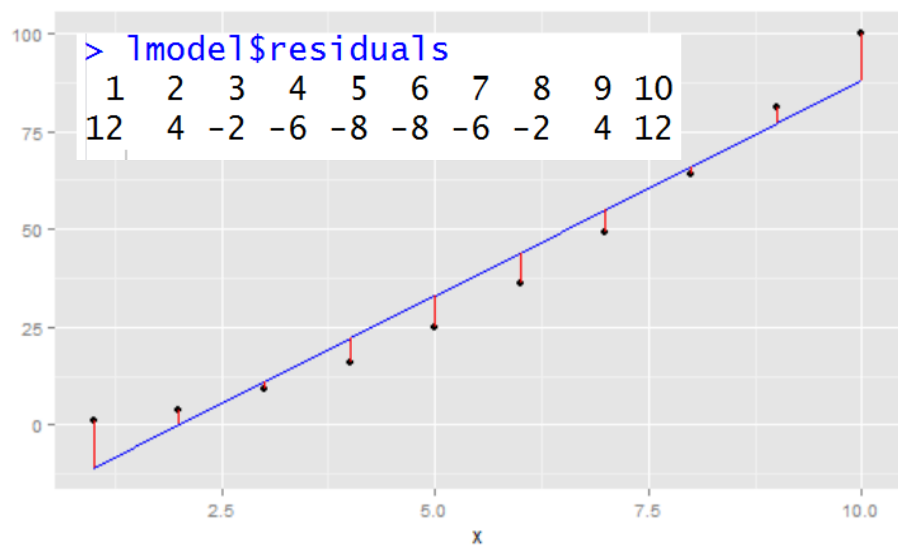
Performance Measures: Scoring Methods

Term	Definition	NOTES
Root Mean Square Error (RMSE)	The square root of the average square of the difference between model prediction and actual value of the response.	• The SAME UNITS as response values, y • Ex: y is in \$ -> RMSE is in \$ • MINIMIZE • A measure of the **width** of data cloud around the perfect prediction line
R-squared (R^2) or Multiple R-squared	The fraction of the y variation explained by the model. Defined as 1.0 minus how much unexplained variance your model leaves (**relative to** null model or the average as the prediction)	• BEST: $R^2=1$ • WORST: $R^2\sim0$ or negative • NO UNITS (dimensionless) • The same as **Multiple R-squared** reported in summary(model)
Correlation	Helpful if variables are potentially useful in a model (Pearson: linear relationship; Spearman: ordered relationship)	• DO NOT USE to measure model quality • Ignores shifts & scaling factors

Heteroscedasticity: Systematic Errors

```r
d <- data.frame(y=(1:10)^2, x=1:10)
lmodel <- lm (y~x, data=d)
```

Ex. 1 in modelEvaluation.Part2.R



*Fitting **linear model into non-linear data** results in over-predicting for some ranges of x and under-predicting for other ranges of x:* ***heteroscedastic****, or structured errors*

Ex. 2 in modelEvaluation.Part2.R

```
> data.frame(RMSE=RMSE, Rsquared=Rsquared,
+            Pearson=pCor, Spearman=sCor)
      RMSE  Rsquared   Pearson Spearman
1 7.266361 0.9497645 0.9745586        1
```

But performance metrics are looking good despite heteroscedasticity!

Errors must be uncorrelated with the response, i.e. homoscedastic, or unstructured.

Correction for Model Complexity: Adjusted R-squared

R-squared is HIGHER for models with MORE explanatory variables added to the model!

$$\boldsymbol{Adjusted - R^2 = R^2 - \left(1 - R^2\right)\frac{m}{n - m - 1}}$$

$\boldsymbol{m}$ **: the total number of regressors**
$\boldsymbol{n}$**: the sample size**

The adjusted R^2 can be negative, and its value will always be less than or equal to that of R^2.

Performance: Scoring Methods Revisited

Term	Definition	NOTES
Degrees of freedom (df)	The number of data samples (rows) minus the number of coefficients fit	• **MAXIMIZE** • $\boldsymbol{df > 4 * m}, m$: # of regressors
Residual Standard Error (RSE)	The sum of the square of residuals divided by the degrees of freedom.	• RMSE adjusted to the # of rows to be degrees of freedom • Attempts to adjust for model complexity
Adjusted R-squared (Adjusted R²)	Multiple R^2 penalized by the ratio of the degrees of freedom to the number of training samples	• Attempts to correct the fact that more complex models tend to look better on training data due to overfitting
F-statistic	Measure whether linear regression model predicts outcome better than the constant model (the mean of y)	• Checks if the variance of the residuals from constant model and from linear model are statistically significant • Want p-value < 0.05

The F statistic is testing the hypothesis that all of the slopes ($\widehat{\beta}_i$) are equal to 0.

Ex. 3: Correction for Model Complexity

```r
states <- as.data.frame(
  state.x77[,c("Murder","Population",
               "Illiteracy", "Income", "Frost")])
```

```
> summary (murderModel)

Call:
lm(formula = Murder ~ Population + Illiteracy + Income + Frost,
    data = dtrain)

Residuals:
    Min      1Q  Median      3Q     Max 
-4.6803 -1.9564  0.8795  2.0245  3.6822 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)   
(Intercept) -0.5562389  5.3083560  -0.105  0.91759   
Population   0.0003505  0.0001413   2.481  0.02209 * 
Illiteracy   5.1626172  1.4240915   3.625  0.00169 **
Income      -0.0001088  0.0008357  -0.130  0.89774   
Frost        0.0129793  0.0168122   0.772  0.44913   
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.619 on 20 degrees of freedom
Multiple R-squared:  0.6088,	Adjusted R-squared:  0.5305 
F-statistic:  7.78 on 4 and 20 DF,  p-value: 0.0005955
```

*Q1: **Multiple R-squared**: How much (%) unexplained variance the model leaves relative to the null/average model?*

*Q2: Which **predictors** are statistically significant?*

*Q3: **Impact**: Holding all the other predictors constant, how much the increase in 1% of Illiteracy contributes to the increase/decrease in the Murder rate (%-wise)? Is it additive or multiplicative contribution? Why?*

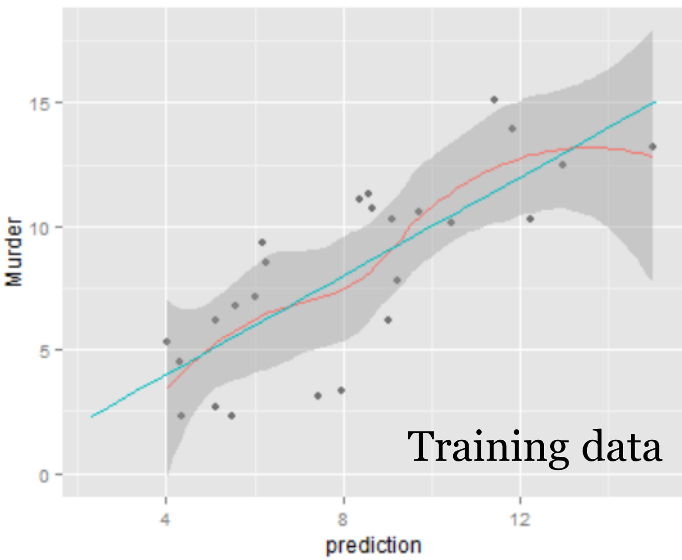
*Q4: **DF**: How many degrees of freedom does the model have?*

*Q5: **Model complexity**: How adjusted R-squared different from RSE or Multiple R-squared?*

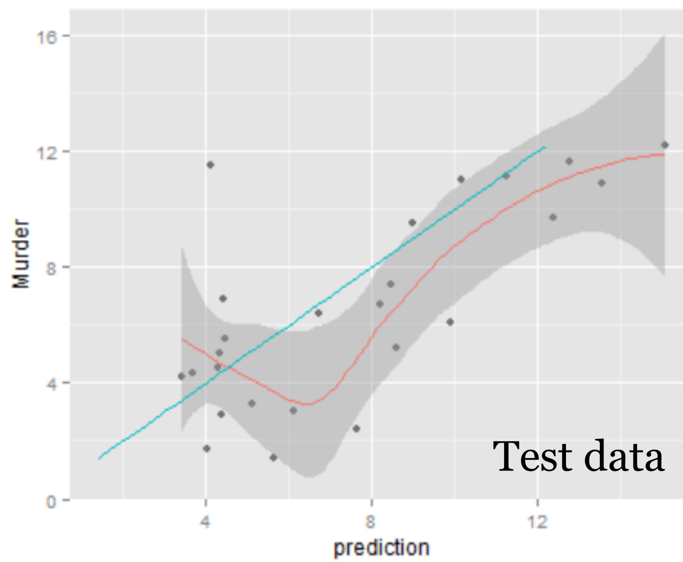
*Q6: **Baseline**: What does F-statistic tell you?*

Ex. 4: When vs. Where Scoring Model Over-/Under-predicts

Prediction vs. Truth



Prediction vs. Truth

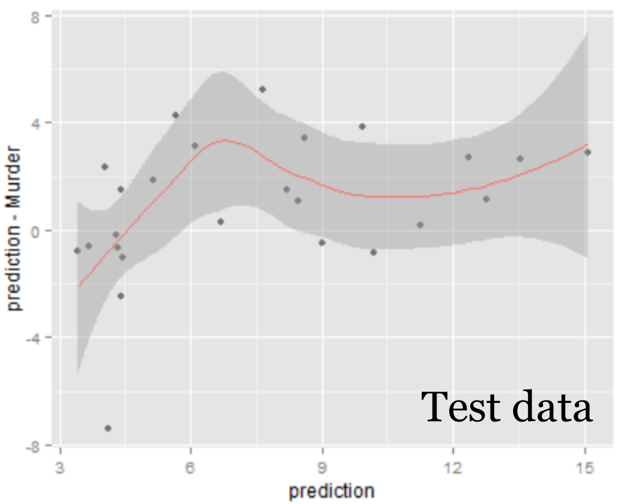


blue line: ideal relation: Murder = prediction
smoothing line: average relation between prediction and actual Murder rate

Q1: On average, are the predictions correct?
Q2: Is smoothing line along the line of perfect fit?

Ex. 4: When vs. Where
Scoring Model Over-/Under-predicts

Prediction vs. Residuals



Truth vs. Residuals

Q1: When the model is over- or under- predicting based on the model's output?
Q2: Where the model is over- or under- predicting based on the actual outcome?

Training vs. Test Performance

```
77 dtrain <- states[1:25,]
78 dtest <- states[26:50,]
```

```
> rmse(dtrain$Murder,dtrain$prediction)
[1] 2.342566
> rmse(dtest$Murder,dtest$prediction)
[1] 2.71301
>
> rsq(dtrain$Murder,dtrain$prediction)
[1] 0.6087654
> rsq(dtest$Murder,dtest$prediction)
[1] 0.35418
```

Q1: What is the difference between the RMSE and R-squared metrics for training and test data?
Q2: How much (%) predictor variables account for the variance in murder rates for the TEST data? How does it compare with the TRAINING data?
Q3: Why such a difference? What is wrong with how the training and test data are constructed?

Reduced Model w/ Significant Predictors

```
> summary(murderModelReduced)

Call:
lm(formula = Murder ~ Population + Illiteracy, data = dtrain)

Residuals:
    Min      1Q  Median      3Q     Max 
-4.6108 -2.1009  0.7153  1.6668  3.4725 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) 1.3811538  1.3005460   1.062   0.2998    
Population  0.0002965  0.0001183   2.507   0.0201 *  
Illiteracy  4.3687909  0.8129197   5.374 2.14e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.535 on 22 degrees of freedom
Multiple R-squared:  0.597,	Adjusted R-squared:  0.5603 
F-statistic: 16.29 on 2 and 22 DF,  p-value: 4.559e-05
```

Q: How performance measures for the reduced model change compared to the original model? Does it make sense to use reduced model? Why?

```
> dtrain$prediction <- predict(murderModelReduced,newdata=dtrain)
> dtest$prediction <- predict(murderModelReduced,newdata=dtest)
> rsq(dtrain$Murder,dtrain$prediction)
[1] 0.5969553
> rsq(dtest$Murder,dtest$prediction)
[1] 0.431016
```

Ex. 5: Inflated Correlation / R-squared

Perfect prediction of a few OUTLIERS produces INFLATED Correlation/R-squared!

```
> y <- c(1,2,3,4,5,9,10)
> ypred <- c(0.5, 0.5, 0.5, 0.5, 0.5, 9, 10)
> cor(y,ypred)
[1] 0.9264604
```

Multicollinearity

Example:
Predictors = (Date_of_birth, Age)
Response = Grip_strength

When you regress grip strength on Predictors, F-test is significant but coefficients are not, i.e. no evidence that either are related to response.

Reasons for such misleading results:

- Highly correlated predictors influence the assessment of how an individual predictor impacts the response, holding all the other predictors constant:
  - e.g. relationship between grip strength and age, holding DOB constant

Side effects of multicollinearity:

- The problem of multicollinearity leads to large confidence intervals for the model parameters and makes interpretation of individual coefficients difficult
- It can also inflate the performance of the model

Ex. 6: Inflation due to Multi-collinearity

Multi-collinearity can inflate performance metrics!

```
> fit <- lm(mpg ~ hp + wt, data=mtcars)
> mpgpred <- predict(fit, newdata=mtcars)
> rmse(mtcars$mpg, mpgpred)
[1] 2.468854
> rsq(mtcars$mpg, mpgpred)
[1] 0.8267855
>
> vifstats <- vif (fit)
> sqrt(vifstats) > 2.0
   hp    wt
FALSE FALSE
```

```
191  data(mtcars)
```

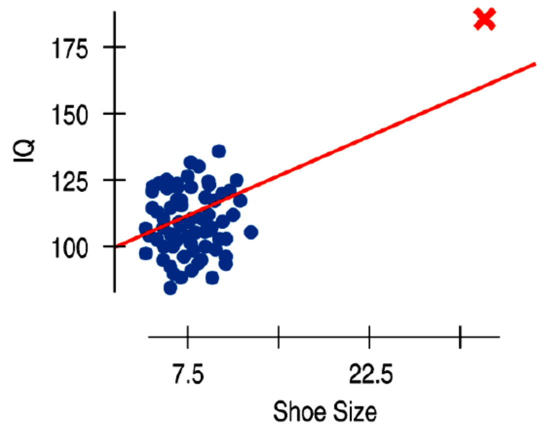
```
> fit <- lm(mpg ~ hp + wt + hp:wt, data=mtcars)
> mpgpred <- predict(fit, newdata=mtcars)
> rmse(mtcars$mpg, mpgpred)
[1] 2.013715
> rsq(mtcars$mpg, mpgpred)
[1] 0.8847637
> vifstats <- vif (fit)
> sqrt(vifstats) > 2.0
   hp    wt hp:wt
 TRUE  TRUE  TRUE
```

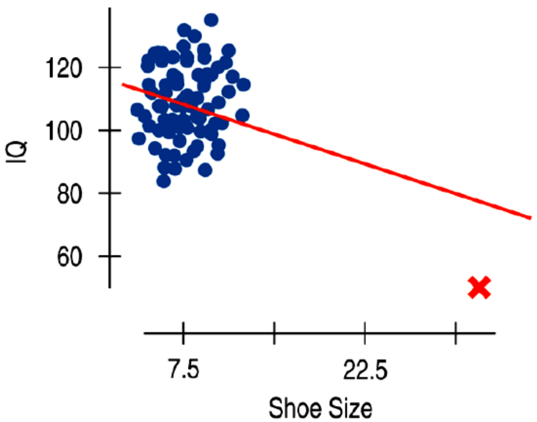
Detect if multicollinearity is present in the data:

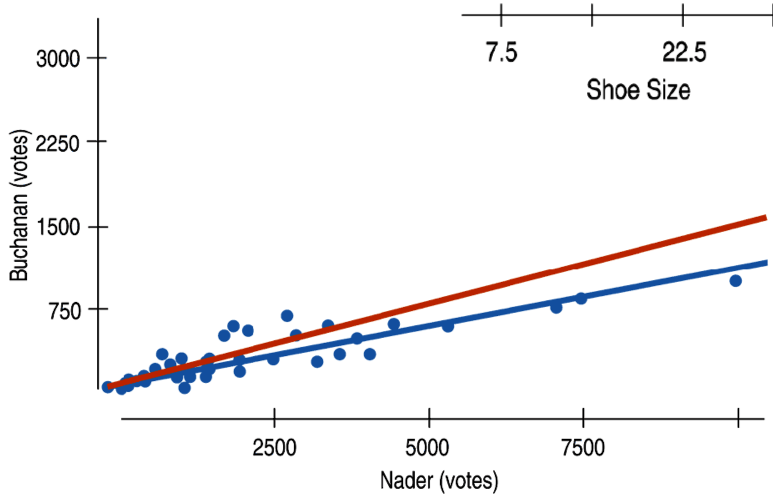
- Test Variance Inflation Factor (vif) statistics
- If sqrt() if vif results > 2.0, then multicollinearity is present in the data

Outliers: High-leverage and Influential

A data point can also be unusual if its *x*-value is far from the mean of the *x*-values. Such points are said to have high *leverage*.







What do you do about outliers?

This outlier is *influential*—it changes the regression line if you omit it.

Things to Watch for (Packages: car and gvlma)

- **Independence:** The observations (rows) must be independent of each other
- **Linearity**: Relationship between the Response (Y) and the predictors (X's) must be linear in terms of the model parameters (β's):
  - Use **crPlots()** in the car pkg for systematic departures from linear model
  - Think how to transform X's to achieve this
- **Normality**: The Response (Y) in a linear regression model must be from the normal (Gaussian distribution):
  - Test for normality using **qqPlot()** in the **car** pkg
  - Check **gvlma** package for global test for linear model assumptions
- **Error/Noise**: The Residuals (predicted – actual) must come from the normal distribution $\mathcal{N}(0,1)$ and should not be autocorrelated (**durbinWatsonTest()**)
- **Homoscedasticity:** The errors (residuals) must be structured:
  - The **car** package provides **ncvTest()** to the hypothesis of constant error variance against the alternative that the error variance changes with the level of fitted values: significant result suggest heteroscedasticity
- **Multicollinearity**: Test for absence of multicollinearity (**vif()** in the **car** pkg)
- **Sensitivity to outliers**: Sensitivity to outliers may affect model performance:
  - Use **outlierTest()** in the car pkg to identify ***high-leverage observations*** and ***influential observations***
- **Model complexity**: Feature selection (forward, backward, stepwise regression) and significant coefficient may guide towards models with reduced complexity