Society of Actuaries in Ireland

R for actuaries: Generalized Linear Models in R

Jean Rea

Disclaimer

The views expressed in this presentation are those of the presenter(s) and not necessarily of the Society of Actuaries in Ireland

Introduction

- Basic introduction to GLMs in R
- Not intended to be advanced
- Assumes some statistical knowledge and basic R knowledge
- Will work through a practical example based on the Titanic data from the kaggle competition
- Uses

Initial analysis

- Understanding of the topic / area
- Understand the problem / objectives of the analysis
- Express the problem in statistical terms
- Data quality
- Exploratory data analysis e.g. numerical and graphical summaries

Linear regression

- Linear regression refresh
- Linear relationship between x (explanatory variable) and y (dependent variable)

$$y_i = \alpha + \beta * x_i + \varepsilon_i$$

- Y is the value of the dependent variable, based on 2 components
  - Non random / structural component $\alpha + \beta * x_i$
  - Random component / error term
- Parameter estimation is based on minimising the prediction error / residual sum of squares

GLM introduction

- GLMs are a flexible generalization of ordinary linear model
- GLMs allow for response distributions other than Normal
- It allows for non linearity in the model structure by allowing the linear model to be related to the response variable via a link function
- Applies to data from an exponential family distribution (Normal, Poisson, Gamma, Binomial…)

GLM introduction

- Response variable in a glm can have any distribution from an exponential family
- Form of the exponential family:

$$f_\theta(y) = exp\left[\frac{y\theta - b(\theta)}{a(\phi)}\right] + c(y, \phi)$$

- a, b and c are arbitrary functions, ϕ is a scale parameter
- Normal, Binomial, Poisson, Gamma…

Exponential family – binomial example

$$
\begin{aligned}
P(Y=y) &= \binom{n}{y} p^{y}(1-p)^{n-y} \\
&= \exp\left\{\log\left[\binom{n}{y} p^{y}(1-p)^{n-y}\right]\right\} \\
&= \exp\left\{\log\binom{n}{y} + y\log p + (n-y)\log(1-p)\right\} \\
&= \exp\left\{\log\binom{n}{y} + y\log p - y\log(1-p) + n\log(1-p)\right\} \\
&= \exp\left\{\log\binom{n}{y} + y\log\frac{p}{1-p} + n\log(1-p)\right\} \\
&= \exp\left\{y\log\frac{p}{1-p} + n\log(1-p) + \log\binom{n}{y}\right\} \\
&= \exp\left\{\frac{y\log\frac{p}{1-p} - n\log\frac{1}{1-p}}{1} + \log\binom{n}{y}\right\}
\end{aligned}
$$

$$f(y) = \exp\left[\frac{y\theta - b(\theta)}{a(\phi)} + c(y,\phi)\right]$$

- $\theta = \log\frac{p}{1-p} = \log\frac{np}{n-np} = \log\frac{\mu}{n-\mu} = g(\mu)$
- $b(\theta) = n\log\frac{1}{1-p} = n\log(1+\exp(\theta))$
- $a(\phi) = 1$
- $c(y,\phi) = \log\binom{n}{y}$

https://www.unc.edu/courses/2010fall/ecol/563/001/docs/lectures/lecture20.htm

Fitting GLMs in R

- ?glm

Fitting Generalized Linear Models

Description

`glm` is used to fit generalized linear models, specified by giving a symbolic description of the linear predictor and a description of the error distribution.

Usage

```
glm(formula, family = gaussian, data, weights, subset,
    na.action, start = NULL, etastart, mustart, offset,
    control = list(...), model = TRUE, method = "glm.fit",
    x = FALSE, y = TRUE, contrasts = NULL, ...)

glm.fit(x, y, weights = rep(1, nobs),
        start = NULL, etastart = NULL, mustart = NULL,
        offset = rep(0, nobs), family = gaussian(),
        control = list(), intercept = TRUE)

## S3 method for class 'glm'
weights(object, type = c("prior", "working"), ...)
```

Basic formula - glm(formula, family=family(link=linkfunction), data=)

Fitting GLMs in R

- ?family

```
binomial(link = "logit")
gaussian(link = "identity")
Gamma(link = "inverse")
inverse.gaussian(link = "1/mu^2")
poisson(link = "log")
quasi(link = "identity", variance = "constant")
quasibinomial(link = "logit")
quasipoisson(link = "log")
```

- Binomial – logistic (binary) regression / response number of successes from known number of trials
- Gamma – strictly positive real valued data
- Poisson – count data

Example: Titanic kaggle competition

- https://www.kaggle.com/c/titanic

Competition Description

- The sinking of the RMS Titanic is one of the most infamous shipwrecks in history. On April 15, 1912, during her maiden voyage, the Titanic sank after colliding with an iceberg, killing 1502 out of 2224 passengers and crew. This sensational tragedy shocked the international community and led to better safety regulations for ships.
- One of the reasons that the shipwreck led to such loss of life was that there were not enough lifeboats for the passengers and crew. Although there was some element of luck involved in surviving the sinking, some groups of people were more likely to survive than others, such as women, children, and the upper-class.
- In this challenge, we ask you to complete the analysis of what sorts of people were likely to survive. In particular, we ask you to apply the tools of machine learning to predict which passengers survived the tragedy.

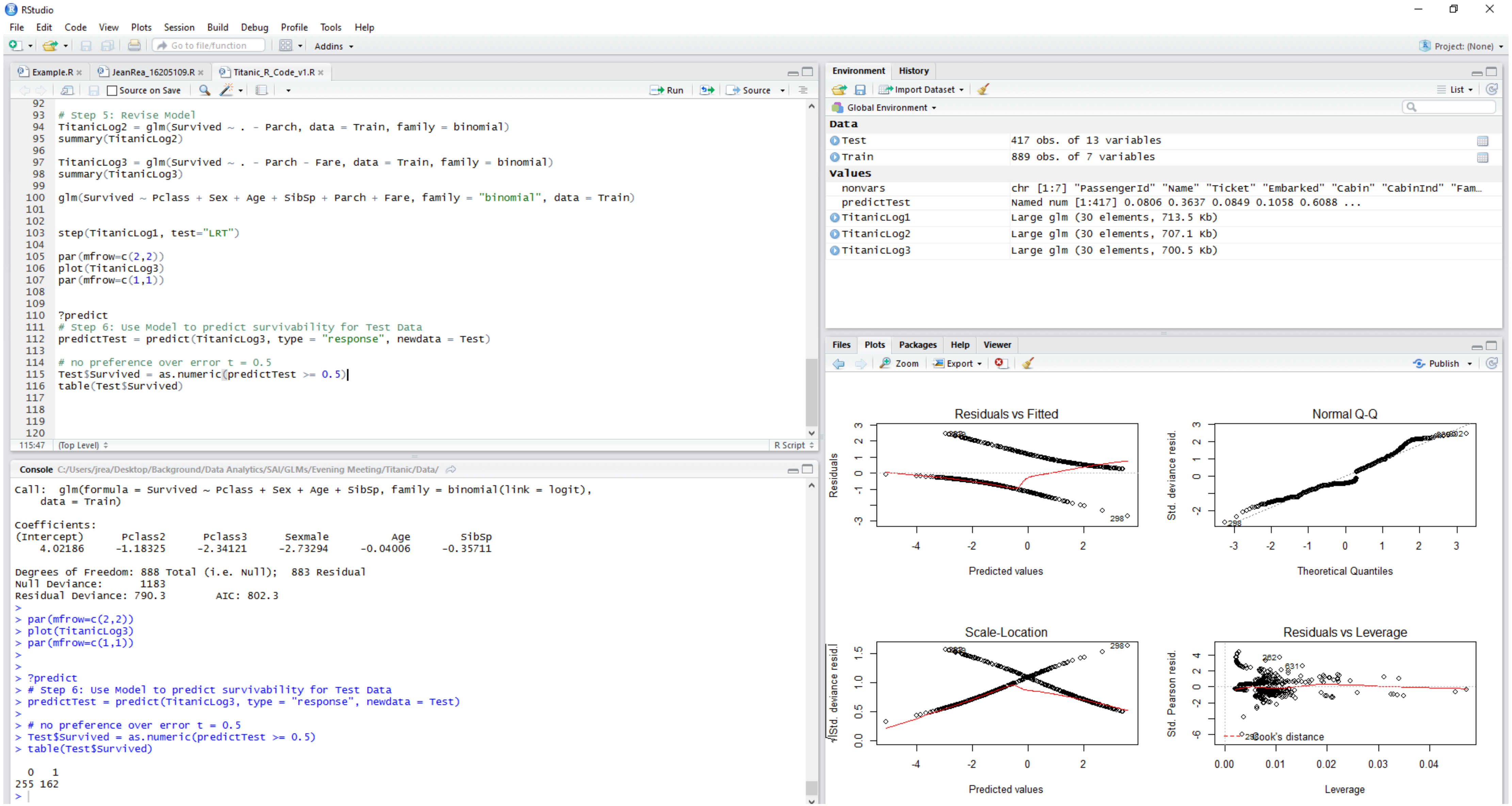
Example: Titanic kaggle competition

Variable	Definition	Level (if applicable)	Notes
survival	Survival	0 = No, 1 = Yes	
pclass	Ticket class	1 = 1st, 2 = 2nd, 3 = 3rd	Proxy for socio-economic status 1^{st} = upper etc.
sex	Sex		
Age	Age in years		Age is fractional if less than 1. If the age is estimated, is it in the form of xx.5
sibsp	# of siblings / spouses aboard the Titanic		Sibling = brother, sister, stepbrother, stepsister Spouse = husband, wife (mistresses and fiancés were ignored)
parch	# of parents / children aboard the Titanic		Parent = mother, father Child = daughter, son, stepdaughter, stepson Some children travelled only with a nanny, therefore parch=0 for them
ticket	Ticket number		
fare	Passenger fare		
cabin	Cabin number		
embarked	Port of Embarkation	C, Q, S	

Example: Titanic kaggle competition

- Will broadly follow the analysis below
- https://www.kaggle.com/jeremyd/titanic-logistic-regression-in-r
- This analysis is based on 7 steps
  1. Load and clean data
  2. Create data frame of variables
  3. Check for multicollinearity
  4. Build a logistic regression model
  5. Revise model
  6. Test accuracy of model on training data – not going to do this part
  7. Use model to predict survivability for test data

R Studio



RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Example.R
JeanRea_16205109.R
Titanic_R_Code_v1.R
Source on Save
Run
Source
# Step 5: Revise Model
TitanicLog2 = glm(Survived ~ . - Parch, data = Train, family = binomial)
summary(TitanicLog2)
TitanicLog3 = glm(Survived ~ . - Parch - Fare, data = Train, family = binomial)
summary(TitanicLog3)
glm(Survived ~ Pclass + Sex + Age + SibSp + Parch + Fare, family = "binomial", data = Train)
step(TitanicLog1, test="LRT")
par(mfrow=c(2,2))
plot(TitanicLog3)
par(mfrow=c(1,1))
?predict
# Step 6: Use Model to predict survivability for Test Data
predictTest = predict(TitanicLog3, type = "response", newdata = Test)
# no preference over error t = 0.5
Test$Survived = as.numeric(predictTest >= 0.5)
table(Test$Survived)
115:47
(Top Level)
R Script
Console C:/Users/jrea/Desktop/Background/Data Analytics/SAI/GLMs/Evening Meeting/Titanic/Data/
Call: glm(formula = Survived ~ Pclass + Sex + Age + SibSp, family = binomial(link = logit),
data = Train)
Coefficients:
(Intercept) Pclass2 Pclass3 Sexmale Age SibSp
4.02186 -1.18325 -2.34121 -2.73294 -0.04006 -0.35711
Degrees of Freedom: 888 Total (i.e. Null); 883 Residual
Null Deviance: 1183
Residual Deviance: 790.3 AIC: 802.3
> par(mfrow=c(2,2))
> plot(TitanicLog3)
> par(mfrow=c(1,1))
> ?predict
> # Step 6: Use Model to predict survivability for Test Data
> predictTest = predict(TitanicLog3, type = "response", newdata = Test)
> # no preference over error t = 0.5
> Test$Survived = as.numeric(predictTest >= 0.5)
> table(Test$Survived)
0 1
255 162
Environment
History
Import Dataset
List
Global Environment
Data
Test 417 obs. of 13 variables
Train 889 obs. of 7 variables
Values
nonvars chr [1:7] "PassengerId" "Name" "Ticket" "Embarked" "Cabin" "CabinInd" "Fam…
predictTest Named num [1:417] 0.0806 0.3637 0.0849 0.1058 0.6088 ...
TitanicLog1 Large glm (30 elements, 713.5 Kb)
TitanicLog2 Large glm (30 elements, 707.1 Kb)
TitanicLog3 Large glm (30 elements, 700.5 Kb)
Files Plots Packages Help Viewer
Zoom
Export
Publish
Residuals vs Fitted
Residuals
Predicted values
Normal Q-Q
Std. deviance resid.
Theoretical Quantiles
Scale-Location
√|Std. deviance resid.|
Predicted values
Residuals vs Leverage
Std. Pearson resid.
Cook's distance
Leverage

Step 1: Load and clean data

- Download csv files from website and save in your own directory
- Set working directory in R: Session→ Set working directory→…
- Then import the data into R, number of calls exist
- ?read.csv

```
read.csv(file, header = TRUE, sep = ",", quote = "\"",
         dec = ".", fill = TRUE, comment.char = "", ...)
```

- File – filename.
- Header - a logical value indicating whether the file contains the names of the variables as its first line.
- Sep - the field separator character. Values on each line of the file are separated by this character.

Step 1: Load and clean data

What it looks like in Excel:

Passenger	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	CabinInd
1	0	3	Braund, M	male	22	1	0	A/5 21171	7.25		S	0
2	1	1	Cumings, I	female	38	1	0	PC 17599	71.2833	C85	C	1
3	1	3	Heikkinen,	female	26	0	0	STON/O2.	7.925		S	0
4	1	1	Futrelle, M	female	35	1	0	113803	53.1	C123	S	1
5	0	3	Allen, Mr.	male	35	0	0	373450	8.05		S	0
6	0	3	Moran, M	male		0	0	330877	8.4583		Q	0

Importing into R:

```
Train=read.csv("train.csv",header=T,na.strings=c(""))
Test=read.csv("test.csv",header=T,na.strings=c(""))
str(Train)
str(Test)
```

Step 1: Load and clean data

- Data frame is a way of storing data – like a matrix except columns can have different data types
- Factors store categorical variables in R

```
> str(Train)
'data.frame':	891 obs. of  13 variables:
 $ PassengerId: int  1 2 3 4 5 6 7 8 9 10 ...
 $ Survived   : int  0 1 1 1 0 0 0 0 1 1 ...
 $ Pclass     : int  3 1 3 1 3 3 1 3 3 2 ...
 $ Name       : Factor w/ 891 levels "Abbing, Mr. Anthony",..: 109 191 358 277 16 559 520 629 41
7 581 ...
 $ Sex        : Factor w/ 2 levels "female","male": 2 1 1 1 2 2 2 2 1 1 ...
 $ Age        : num  22 38 26 35 35 NA 54 2 27 14 ...
 $ SibSp      : int  1 1 0 1 0 0 0 3 0 1 ...
 $ Parch      : int  0 0 0 0 0 0 0 1 2 0 ...
 $ Ticket     : Factor w/ 681 levels "110152","110413",..: 524 597 670 50 473 276 86 396 345 133
 ...
 $ Fare       : num  7.25 71.28 7.92 53.1 8.05 ...
 $ Cabin      : Factor w/ 147 levels "A10","A14","A16",..: NA 82 NA 56 NA NA 130 NA NA NA ...
 $ Embarked   : Factor w/ 3 levels "C","Q","S": 3 1 3 3 3 2 3 3 3 1 ...
 $ CabinInd   : int  0 1 0 1 0 0 1 0 0 0 ...
```

Step 1: Load and clean data

apply(Train,2,function(x) sum(is.na(x)))

```
> apply(Train,2,function(x) sum(is.na(x)))
PassengerId    Survived      Pclass        Name         Sex         Age       SibSp 
          0           0           0           0           0         177           0 
      Parch      Ticket        Fare       Cabin    Embarked    CabinInd 
          0           0           0         687           2           0 
```

- Lots of missing ages and cabins, also 2 observations with missing Embarked

#lots of missing age values for each - replace with mean

Train$Age[is.na(Train$Age)] = mean(Train$Age,na.rm=T)

- Also deal with other missing observations – I have removed them

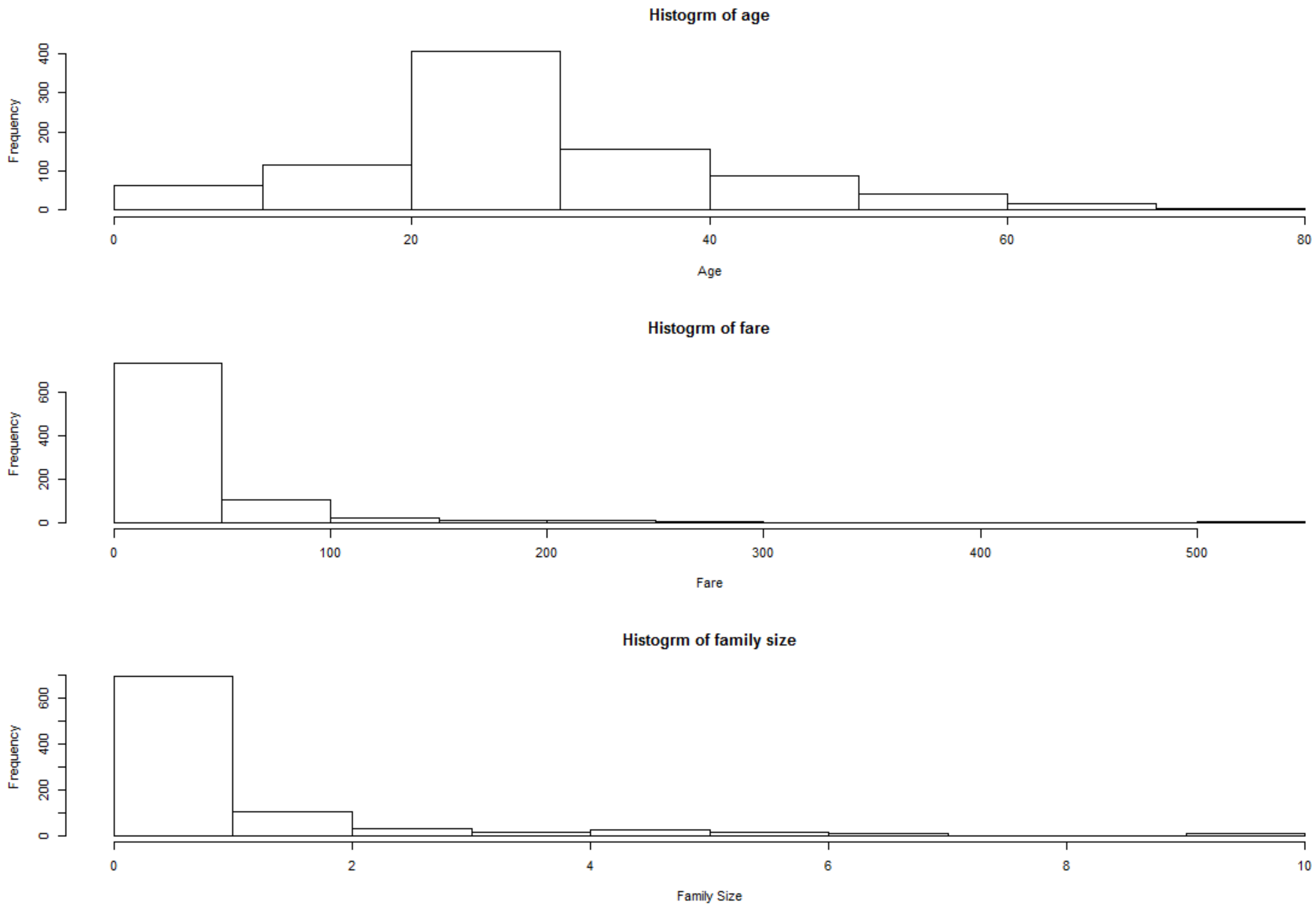
Step 1: Load and clean data

summary(Train)

```
> summary(Train)
  PassengerId       Survived          Pclass                                         Name    
 Min.   :  1    Min.   :0.0000   Min.   :1.000   Abbing, Mr. Anthony                  :  1  
 1st Qu.:224    1st Qu.:0.0000   1st Qu.:2.000   Abbott, Mr. Rossmore Edward          :  1  
 Median :446    Median :0.0000   Median :3.000   Abbott, Mrs. Stanton (Rosa Hunt)     :  1  
 Mean   :446    Mean   :0.3825   Mean   :2.312   Abelson, Mr. Samuel                  :  1  
 3rd Qu.:668    3rd Qu.:1.0000   3rd Qu.:3.000   Abelson, Mrs. Samuel (Hannah Wizosky):  1  
 Max.   :891    Max.   :1.0000   Max.   :3.000   Adahl, Mr. Mauritz Nils Martin       :  1  
                                                 (Other)                              :883  
     Sex           Age            SibSp            Parch             Ticket   
 female:312   Min.   : 0.42   Min.   :0.0000   Min.   :0.0000   1601    :  7  
 male  :577   1st Qu.:22.00   1st Qu.:0.0000   1st Qu.:0.0000   347082  :  7  
              Median :29.70   Median :0.0000   Median :0.0000   CA. 2343:  7  
              Mean   :29.65   Mean   :0.5242   Mean   :0.3825   3101295 :  6  
              3rd Qu.:35.00   3rd Qu.:1.0000   3rd Qu.:0.0000   347088  :  6  
              Max.   :80.00   Max.   :8.0000   Max.   :6.0000   CA 2144 :  6  
                                                                (Other) :850  
      Fare                    Cabin      Embarked    CabinInd     
 Min.   :  0.000   B96 B98    :  4   C:168      Min.   :0.0000  
 1st Qu.:  7.896   C23 C25 C27:  4   Q: 77      1st Qu.:0.0000  
 Median : 14.454   G6         :  4   S:644      Median :0.0000  
 Mean   : 32.097   C22 C26    :  3              Mean   :0.2272  
 3rd Qu.: 31.000   D          :  3              3rd Qu.:0.0000  
 Max.   :512.329   (Other)    :184              Max.   :1.0000  
                   NA's       :687                              
```

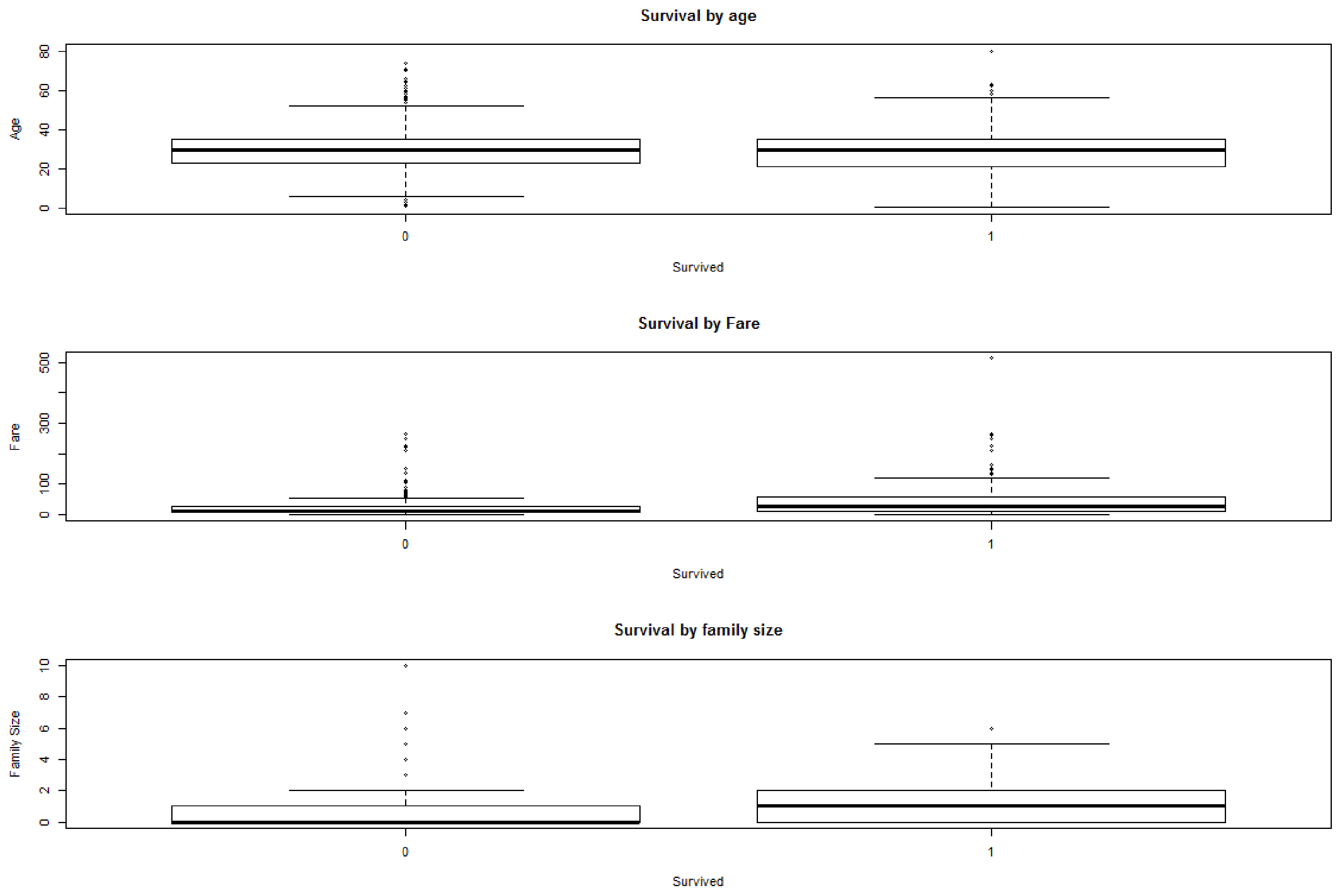
Step 1: Load and clean data

Histograms



Step 1: Load and clean data

Boxplots



Step 1: Load and clean data

Also consider categorical variables

```
> table(Train$Sex, Train$Survived)

           0   1
  female  81 231
  male   468 109
> prop.table(table(Train$Sex, Train$Survived),1)

                 0         1
  female 0.2596154 0.7403846
  male   0.8110919 0.1889081
> prop.table(table(Train$Sex, Train$Survived),2)

                 0         1
  female 0.1475410 0.6794118
  male   0.8524590 0.3205882
```

Step 2: Create data frame of variables

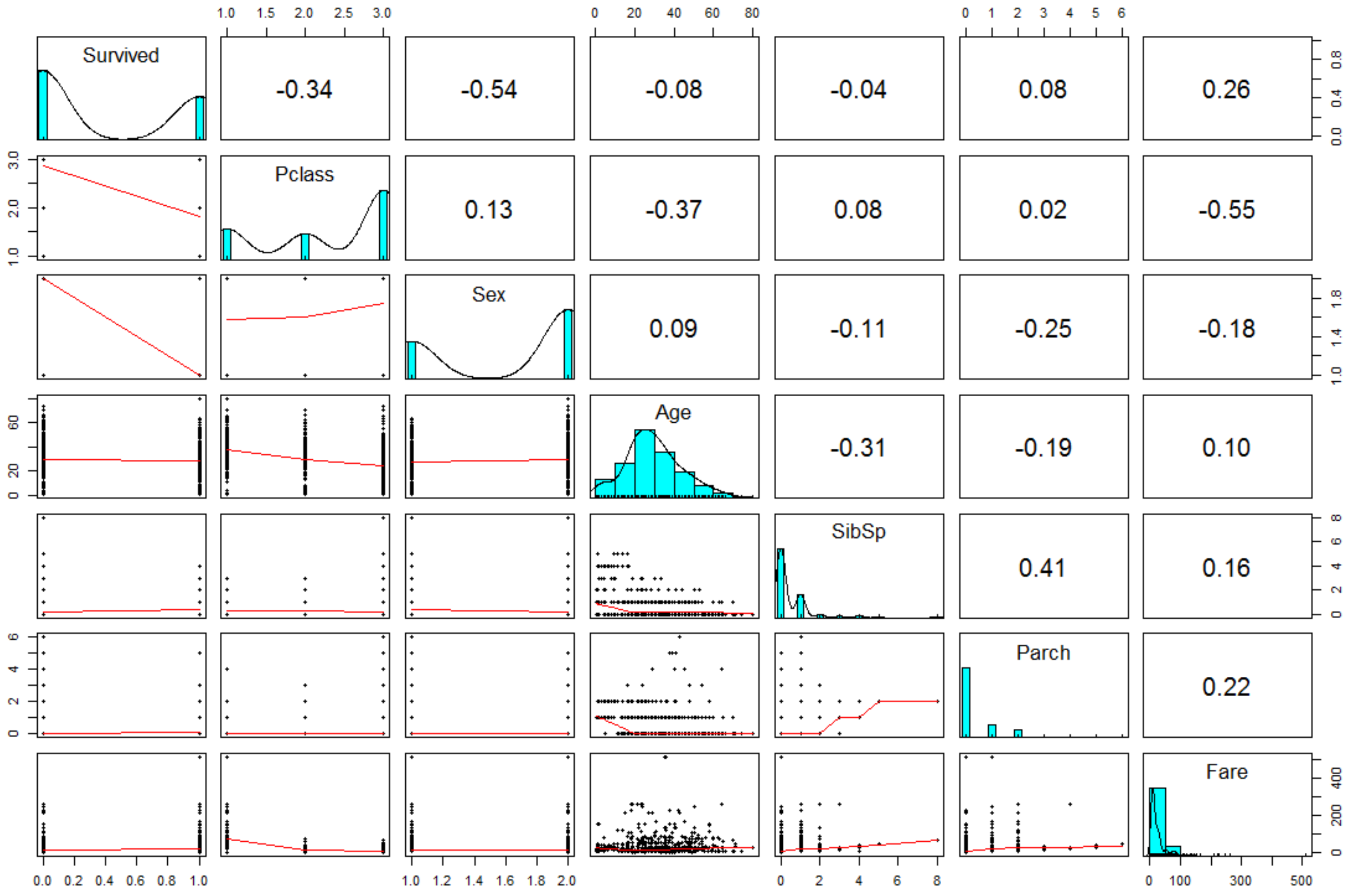
```
# Step 2: Create DF of independent/dependent variables
nonvars = c("PassengerId","Name","Ticket","Embarked","Cabin","CabinInd")
Train = Train[,!(names(Train) %in% nonvars)]
str(Train)
```

```
> str(Train)
'data.frame':	889 obs. of  7 variables:
 $ Survived: int  0 1 1 1 0 0 0 0 1 1 ...
 $ Pclass  : int  3 1 3 1 3 3 1 3 3 2 ...
 $ Sex     : Factor w/ 2 levels "female","male": 2 1 1 1 2 2 2 2 1 1 ...
 $ Age     : num  22 38 26 35 35 ...
 $ SibSp   : int  1 1 0 1 0 0 0 3 0 1 ...
 $ Parch   : int  0 0 0 0 0 0 0 1 2 0 ...
 $ Fare    : num  7.25 71.28 7.92 53.1 8.05 ...
```

```
Train$Pclass = as.factor(Train$Pclass)
```

Step 3: Collinearity

Step 3: Check for MultiCollinearity



Step 4: Build a logistic regression model

```
# Step 4: Build a Logistic Regression Model
TitanicLog1 = glm(Survived~., data = Train, family = binomial(link=logit))
summary(TitanicLog1)
```

Step 4: Build a logistic regression model

```
Call:
glm(formula = Survived ~ ., family = binomial(link = logit), 
    data = Train)

Deviance Residuals: 
    Min       1Q   Median       3Q      Max  
-2.7055  -0.6098  -0.4271   0.6147   2.4188  

Coefficients:
             Estimate Std. Error z value Pr(>|z|)    
(Intercept)  3.836254   0.446497   8.592  < 2e-16 ***
Pclass2     -1.017868   0.293867  -3.464 0.000533 ***
Pclass3     -2.144843   0.289561  -7.407 1.29e-13 ***
Sexmale     -2.753512   0.199454 -13.805  < 2e-16 ***
Age         -0.039687   0.007858  -5.051 4.40e-07 ***
SibSp       -0.349212   0.109498  -3.189 0.001427 ** 
Parch       -0.111842   0.117598  -0.951 0.341579    
Fare         0.002969   0.002441   1.216 0.223854    
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1182.82  on 888  degrees of freedom
Residual deviance:  788.16  on 881  degrees of freedom
AIC: 804.16

Number of Fisher Scoring iterations: 5
```

. means to include first order terms of all variables

Other options:
0 to exclude the intercept
- to exclude terms
: to include interactions

Alternative is to explicitly state them in the formula:
glm(Survived ~ Pclass + Sex + Age + SibSp + Parch + Fare, family = binomial, data = Train)

Step 4: Build a logistic regression model

```
Call:
glm(formula = Survived ~ ., family = binomial(link = logit),
    data = Train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.7055  -0.6098  -0.4271   0.6147   2.4188

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  3.836254   0.446497   8.592  < 2e-16 ***
Pclass2     -1.017868   0.293867  -3.464 0.000533 ***
Pclass3     -2.144843   0.289561  -7.407 1.29e-13 ***
Sexmale     -2.753512   0.199454 -13.805  < 2e-16 ***
Age         -0.039687   0.007858  -5.051 4.40e-07 ***
SibSp       -0.349212   0.109498  -3.189 0.001427 **
Parch       -0.111842   0.117598  -0.951 0.341579
Fare         0.002969   0.002441   1.216 0.223854
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1182.82  on 888  degrees of freedom
Residual deviance:  788.16  on 881  degrees of freedom
AIC: 804.16

Number of Fisher Scoring iterations: 5
```

Step 4: Build a logistic regression model

```
Call:
glm(formula = Survived ~ ., family = binomial(link = logit),
    data = Train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.7055  -0.6098  -0.4271   0.6147   2.4188

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  3.836254   0.446497   8.592  < 2e-16 ***
Pclass2     -1.017868   0.293867  -3.464 0.000533 ***
Pclass3     -2.144843   0.289561  -7.407 1.29e-13 ***
Sexmale     -2.753512   0.199454 -13.805  < 2e-16 ***
Age         -0.039687   0.007858  -5.051 4.40e-07 ***
SibSp       -0.349212   0.109498  -3.189 0.001427 **
Parch       -0.111842   0.117598  -0.951 0.341579
Fare         0.002969   0.002441   1.216 0.223854
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1182.82  on 888  degrees of freedom
Residual deviance:  788.16  on 881  degrees of freedom
AIC: 804.16

Number of Fisher Scoring iterations: 5
```

Note factors / categorical variables are relative to a baseline

Step 4: Build a logistic regression model

```
Call:
glm(formula = Survived ~ ., family = binomial(link = logit), 
    data = Train)

Deviance Residuals: 
    Min       1Q   Median       3Q      Max  
-2.7055  -0.6098  -0.4271   0.6147   2.4188  

Coefficients:
             Estimate Std. Error z value Pr(>|z|)    
(Intercept)  3.836254   0.446497   8.592  < 2e-16 ***
Pclass2     -1.017868   0.293867  -3.464 0.000533 ***
Pclass3     -2.144843   0.289561  -7.407 1.29e-13 ***
Sexmale     -2.753512   0.199454 -13.805  < 2e-16 ***
Age         -0.039687   0.007858  -5.051 4.40e-07 ***
SibSp       -0.349212   0.109498  -3.189 0.001427 ** 
Parch       -0.111842   0.117598  -0.951 0.341579    
Fare         0.002969   0.002441   1.216 0.223854    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1182.82  on 888  degrees of freedom
Residual deviance:  788.16  on 881  degrees of freedom
AIC: 804.16

Number of Fisher Scoring iterations: 5
```

Hypothesis testing individual coefficients.

Null hypothesis is that parameter is zero.

Test statistic z-statistic is the estimate divided by the standard error e.g. Pclass2 -1.02/.29=-3.51

Note here scale parameter is known.

z-statistic is asymptotically standard normal when Ho is true and the sample size is fairly large.

Step 4: Build a logistic regression model

```
Call:
glm(formula = Survived ~ ., family = binomial(link = logit),
    data = Train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.7055  -0.6098  -0.4271   0.6147   2.4188

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  3.836254   0.446497   8.592  < 2e-16 ***
Pclass2     -1.017868   0.293867  -3.464 0.000533 ***
Pclass3     -2.144843   0.289561  -7.407 1.29e-13 ***
Sexmale     -2.753512   0.199454 -13.805  < 2e-16 ***
Age         -0.039687   0.007858  -5.051 4.40e-07 ***
SibSp       -0.349212   0.109498  -3.189 0.001427 **
Parch       -0.111842   0.117598  -0.951 0.341579
Fare         0.002969   0.002441   1.216 0.223854
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1182.82  on 888  degrees of freedom
Residual deviance:  788.16  on 881  degrees of freedom
AIC: 804.16

Number of Fisher Scoring iterations: 5
```

Step 4: Build a logistic regression model

```
Call:
glm(formula = Survived ~ ., family = binomial(link = logit),
    data = Train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.7055  -0.6098  -0.4271   0.6147   2.4188

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  3.836254   0.446497   8.592  < 2e-16 ***
Pclass2     -1.017868   0.293867  -3.464 0.000533 ***
Pclass3     -2.144843   0.289561  -7.407 1.29e-13 ***
Sexmale     -2.753512   0.199454 -13.805  < 2e-16 ***
Age         -0.039687   0.007858  -5.051 4.40e-07 ***
SibSp       -0.349212   0.109498  -3.189 0.001427 **
Parch       -0.111842   0.117598  -0.951 0.341579
Fare         0.002969   0.002441   1.216 0.223854
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1182.82  on 888  degrees of freedom
Residual deviance:  788.16  on 881  degrees of freedom
AIC: 804.16

Number of Fisher Scoring iterations: 5
```

Check on overall model fit / appropriateness – Probability of a Chi-Squared 881 random variable being as large as 788. The probability is approximately 99% which is high and supports that it does follow a Chi-Squared distribution with 881 df.

The Null deviance is the deviance for a model with just a constant term

Residual deviance is the deviance of the fitted model.

These can be combined to give the proportion deviance explained, a generalization of R Squared, as follows:

```
> (1182.82-788.16)/1182.82 [1]
0.3336602
```

Step 5: Revise model

```
Call:
glm(formula = Survived ~ . - Parch, family = binomial, data = Train)

Deviance Residuals: 
    Min       1Q   Median       3Q      Max  
-2.7394  -0.6023  -0.4206   0.6102   2.4498  

Coefficients:
             Estimate Std. Error z value Pr(>|z|)    
(Intercept)  3.811224   0.443929   8.585  < 2e-16 ***
Pclass2     -1.043488   0.291840  -3.576 0.000349 ***
Pclass3     -2.176099   0.286569  -7.594 3.11e-14 ***
Sexmale     -2.717378   0.195045 -13.932  < 2e-16 ***
Age         -0.039417   0.007838  -5.029 4.93e-07 ***
SibSp       -0.377069   0.106138  -3.553 0.000381 ***
Fare         0.002462   0.002319   1.062 0.288339    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1182.82  on 888  degrees of freedom
Residual deviance:  789.08  on 882  degrees of freedom
AIC: 803.08

Number of Fisher Scoring iterations: 5
```

-Parch
Means to remove Parch

Step 5: Revise model

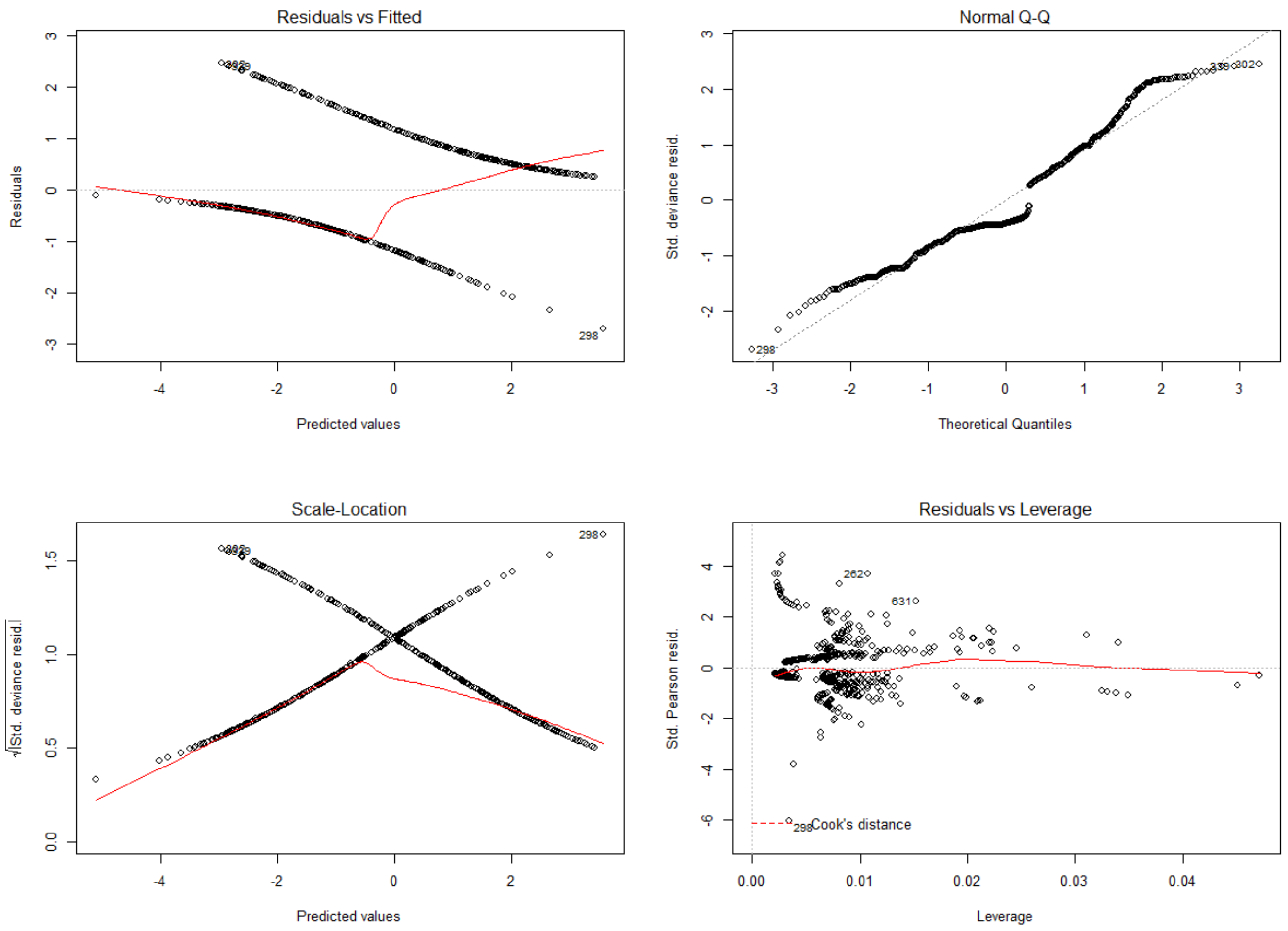
```
Call:
glm(formula = Survived ~ . - Parch - Fare, family = binomial,
    data = Train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.6877  -0.6028  -0.4219   0.6116   2.4523

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  4.021856   0.399906  10.057  < 2e-16 ***
Pclass2     -1.183245   0.261950  -4.517 6.27e-06 ***
Pclass3     -2.341213   0.242938  -9.637  < 2e-16 ***
Sexmale     -2.732937   0.194376 -14.060  < 2e-16 ***
Age         -0.040059   0.007813  -5.128 2.94e-07 ***
SibSp       -0.357112   0.104111  -3.430 0.000603 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1182.8  on 888  degrees of freedom
Residual deviance:  790.3  on 883  degrees of freedom
AIC: 802.3

Number of Fisher Scoring iterations: 5
```

We can do this in an automated way:
step(TitanicLog1, test="LRT")

Step 5: Revise model

- Check model diagnostics – not very informative for logistic regression

plot(TitanicLog3)



Step 7: Use model to predict survival on test data

- # Step 7: Use Model to predict survivability for Test Data
- predictTest = predict(TitanicLog3, type = "response", newdata = Test)
- Setting type to response means that you get the predicted probabilities otherwise the default for a binomial are predictions on the logit / log odds scale
- If we had the survival indicator for the test data we could also calculate a misclassification rate

Interpretation

```
Call:
glm(formula = Survived ~ . - Parch - Fare, family = binomial,
    data = Train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.6877  -0.6028  -0.4219   0.6116   2.4523

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  4.021856   0.399906  10.057  < 2e-16 ***
Pclass2     -1.183245   0.261950  -4.517 6.27e-06 ***
Pclass3     -2.341213   0.242938  -9.637  < 2e-16 ***
Sexmale     -2.732937   0.194376 -14.060  < 2e-16 ***
Age         -0.040059   0.007813  -5.128 2.94e-07 ***
SibSp       -0.357112   0.104111  -3.430 0.000603 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1182.8  on 888  degrees of freedom
Residual deviance:  790.3  on 883  degrees of freedom
AIC: 802.3

Number of Fisher Scoring iterations: 5
```

- Logodds, odds and probabilities are variations of each other
- The estimates are on the scale of the linear predictor (logodds for binomial), can convert
- Factors need to be interpreted relative to the baseline
- For continuous it relates to a one unit increase e.g. a one unit increase in age changes the log odds of surviving by an estimated -0.04

Other considerations

- Over dispersion
- Use of offset
  - Paper "Applications of the Offset in Property-Casualty Predictive Modeling"

Other considerations - overdispersion

- Illustrative glm call for quasibinomial - not saying it is present here

```
Call:
glm(formula = Survived ~ . - Parch - Fare, family = quasibinomial,
    data = Train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.6877  -0.6028  -0.4219   0.6116   2.4523

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  4.021856   0.407074   9.880  < 2e-16 ***
Pclass2     -1.183245   0.266645  -4.438 1.02e-05 ***
Pclass3     -2.341213   0.247292  -9.467  < 2e-16 ***
Sexmale     -2.732937   0.197860 -13.813  < 2e-16 ***
Age         -0.040059   0.007953  -5.037 5.73e-07 ***
SibSp       -0.357112   0.105977  -3.370 0.000785 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for quasibinomial family taken to be 1.036169)

    Null deviance: 1182.8  on 888  degrees of freedom
Residual deviance:  790.3  on 883  degrees of freedom
AIC: NA

Number of Fisher Scoring iterations: 5
```

Other datasets

Third party motor insurance claims in Sweden in 1977

Description

In Sweden all motor insurance companies apply identical risk arguments to classify customers, and thus their portfolios and their claims statistics can be combined. The data were compiled by a Swedish Committee on the Analysis of Risk Premium in Motor Insurance. The Committee was asked to look into the problem of analyzing the real influence on claims of the risk arguments and to compare this structure with the actual tariff.

Usage

```
data(motorins)
```

Format

A data frame with 1797 observations on the following 8 variables.

Kilometres

: an ordered factor representing kilomoters per year with levels 1: < 1000, 2: 1000-15000, 3: 15000-20000, 4: 20000-25000, 5: > 25000

Zone

: a factor representing geographical area with levels 1: Stockholm, Goteborg, Malmo with surroundings 2: Other large cities with surroundings 3: Smaller cities with surroundings in southern Sweden 4: Rural areas in southern Sweden 5: Smaller cities with surroundings in northern Sweden 6: Rural areas in northern Sweden 7: Gotland

Bonus

: No claims bonus. Equal to the number of years, plus one, since last claim

Make

: A factor representing eight different common car models. All other models are combined in class 9

Insured

: Number of insured in policy-years

Claims

: Number of claims

Payment

: Total value of payments in Skr

perd

: payment per claim

Other datasets

- install.packages("faraway")
- library(faraway)
- data(motorins)
- str(motorins)

```
> str(motorins)
'data.frame':   1797 obs. of  8 variables:
 $ Kilometres: Ord.factor w/ 5 levels "1"<"2"<"3"<"4"<..: 1 1 1 1 1 1 1 1 1 1 ...
 $ Zone      : Factor w/ 7 levels "1","2","3","4",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ Bonus     : int  1 1 1 1 1 1 1 1 1 2 ...
 $ Make      : Factor w/ 9 levels "1","2","3","4",..: 1 2 3 4 5 6 7 8 9 1 ...
 $ Insured   : num  455.1 69.2 72.9 1292.4 191 ...
 $ Claims    : int  108 19 13 124 40 57 23 14 1704 45 ...
 $ Payment   : int  392491 46221 15694 422201 119373 170913 56940 77487 6805992 214011 ...
 $ perd      : num  3634 2433 1207 3405 2984 ...
```

Uses

- Non Life pricing
- Non Life reserving – see paper “STOCHASTIC LOSS RESERVING USING GENERALIZED LINEAR MODELS” by *Greg Taylor and Gráinne McGuire*
- Life modelling

R sessions – coming soon

Questions

Thank you