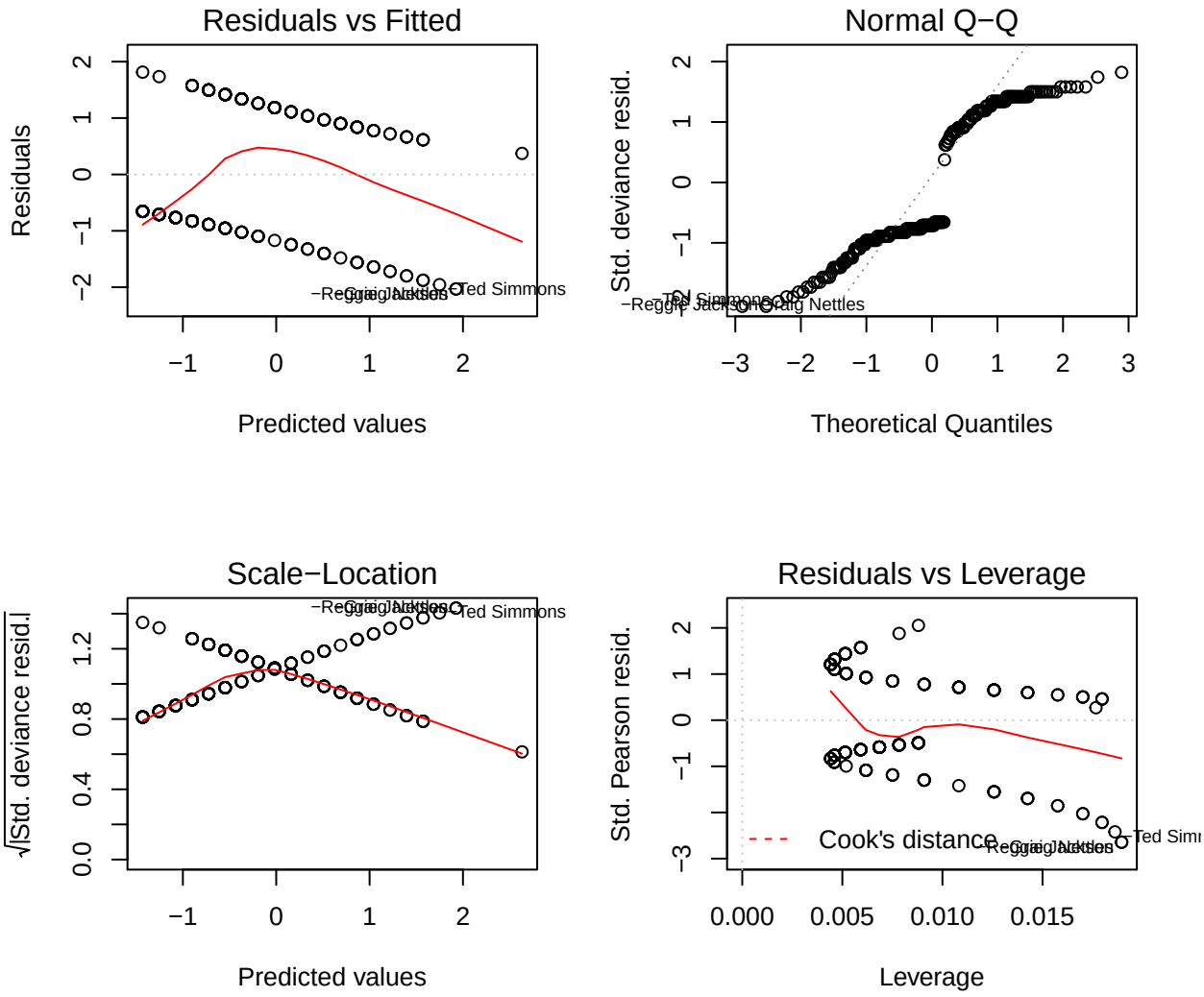
Stat 435 lecture notes 9: supplementary

Diagnosis for Logistic regression

```
library(ISLR)
Hitters = na.omit(Hitters)
Hitters$Salary1 = as.numeric(Hitters$Salary > 500)

fitRes = glm(Salary1 ~ Years, data = Hitters, family = "binomial")
par(mfrow = c(2, 2))
plot(fitRes)
```



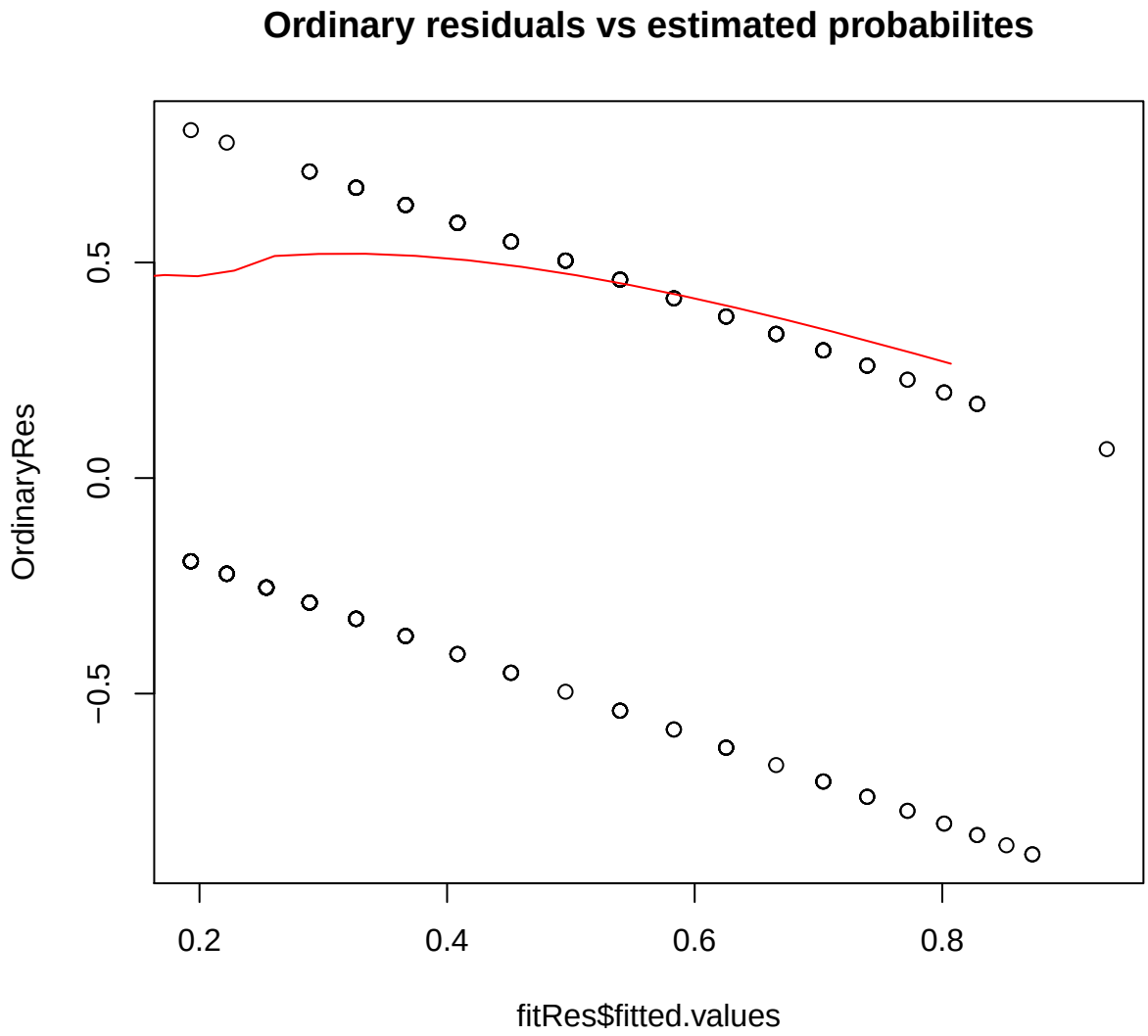
```r
# check residuals; caution: these residuals are not as
# desired
max(fitRes$residuals)
```

```
## [1] 5.181124
```

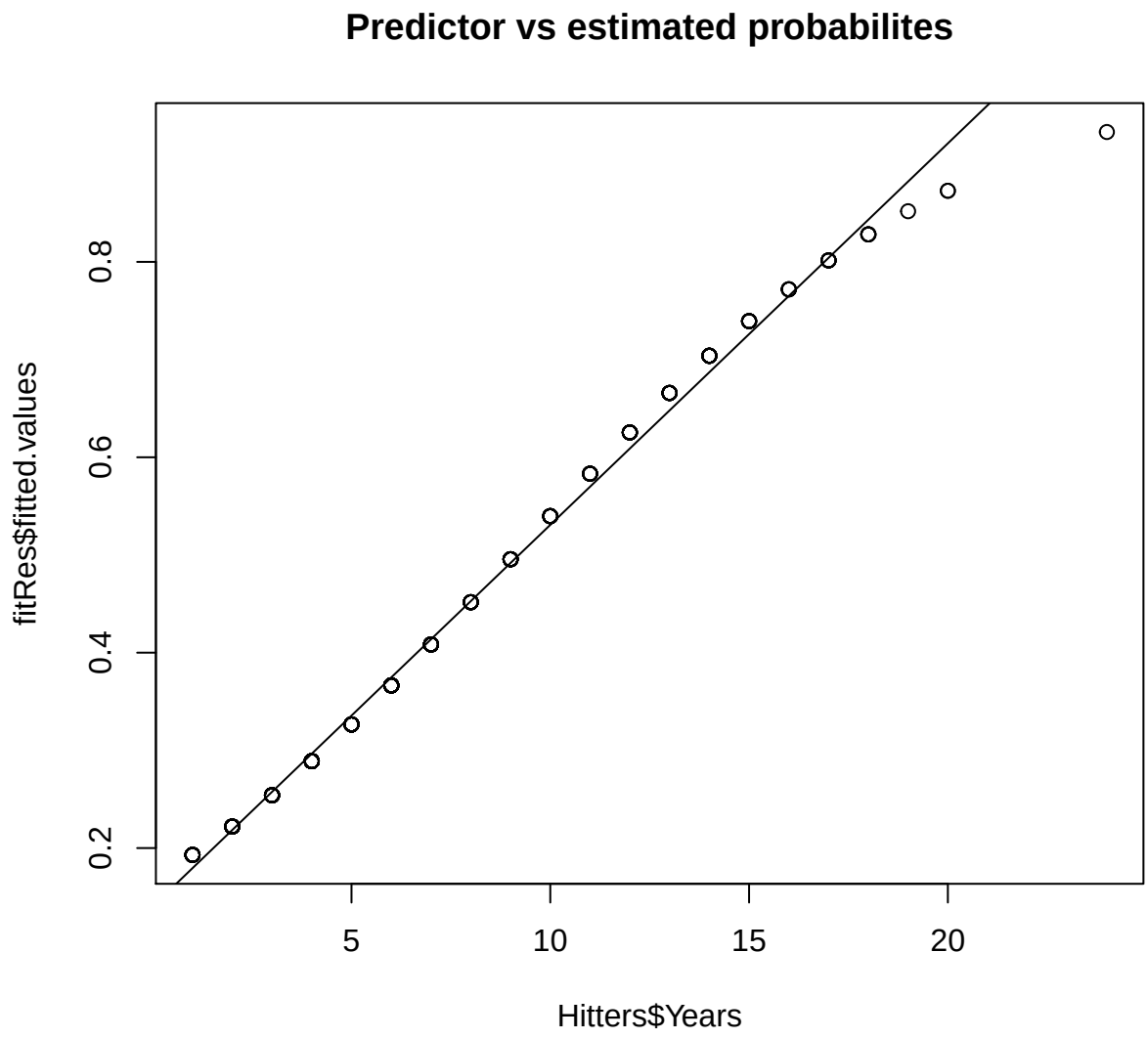
```r
min(fitRes$residuals)
```

```
## [1] -7.86094
```

```r
# plot residuals versus fitted values
par(mfrow = c(1, 1))  # row-column
OrdinaryRes = Hitters$Salary1 - fitRes$fitted.values
plot(fitRes$fitted.values, OrdinaryRes, main = "Ordinary residuals vs estimated probabilites")
# add a lowess fit
lines(lowess(OrdinaryRes, fitRes$fitted.values, f = 0.8), col = 2)
```



```r
# plot logit of estimated probabilities against predictors
plot(Hitters$Years, fitRes$fitted.values, main = "Predictor vs estimated probabilites")
abline(lm(fitRes$fitted.values ~ Hitters$Years))
```



Extract more statistics from `glm`

First, let us check what `groom::augment` applied to a `glm` object does by reading https://rdrr.io/cran/broom/man/augment.glm.html.

When newdata is not supplied, augment.lm returns one row for each observation, with seven columns added to the original data:

```
.hat
Diagonal of the hat matrix
```

```
.sigma
Estimate of residual standard deviation when corresponding observation is dropped from model

.cooksd
Cooks distance, cooks.distance()

.fitted
Fitted values of model

.se.fit
Standard errors of fitted values

.resid
Residuals

.std.resid
Standardised residuals
```

Some unusual lm objects, such as rlm from MASS, may omit .cooksd and .std.resid. gam from mgcv omits .sigma.

When newdata is supplied, returns one row for each observation, with three columns added to the new data:

```
.fitted
Fitted values of model

.se.fit
Standard errors of fitted values

.resid
Residuals of fitted values on the new data
```

Illustration on extracting statistics for diagnosis

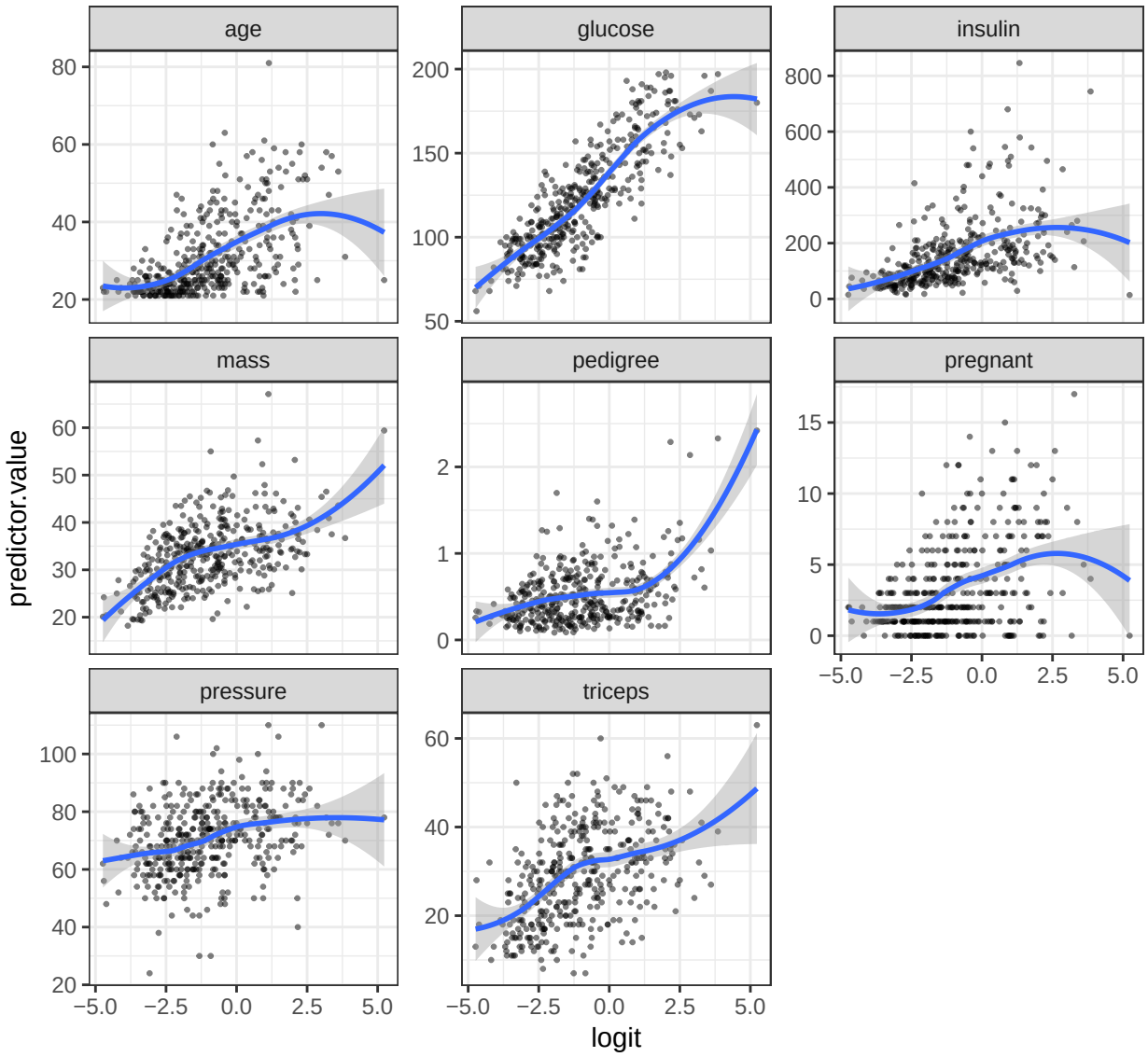
The following are adopted from: www.sthda.com/english/articles/36-classification-methods-essentials/148-logistic-regression-assumptions-and-diagnostics-in-r/

```r
library(dplyr)
library(tidyverse)
library(broom)
theme_set(theme_classic())

# Load the data
data("PimaIndiansDiabetes2", package = "mlbench")
PimaIndiansDiabetes2 <- na.omit(PimaIndiansDiabetes2)
# Fit the logistic regression model
model <- glm(diabetes ~ ., data = PimaIndiansDiabetes2, family = binomial)
```

```r
# Predict the probability (p) of diabete positivity
probabilities <- predict(model, type = "response")
predicted.classes <- ifelse(probabilities > 0.5, "pos", "neg")
head(predicted.classes)
```

```
##     4     5     7     9    14    15
## "neg" "pos" "neg" "pos" "pos" "pos"
```

```r
# Select only numeric predictors
mydata <- PimaIndiansDiabetes2 %>% dplyr::select_if(is.numeric)
predictors <- colnames(mydata)
# Bind the logit and tidying the data for plot
mydata <- mydata %>% mutate(logit = log(probabilities/(1 - probabilities))) %>%
    gather(key = "predictors", value = "predictor.value", -logit)

# example for 'gather':
# https://www.rdocumentation.org/packages/tidyr/versions/0.8.3/topics/gather

ggplot(mydata, aes(logit, predictor.value)) + geom_point(size = 0.5,
    alpha = 0.5) + geom_smooth(method = "loess") + theme_bw() +
    facet_wrap(~predictors, scales = "free_y")
```

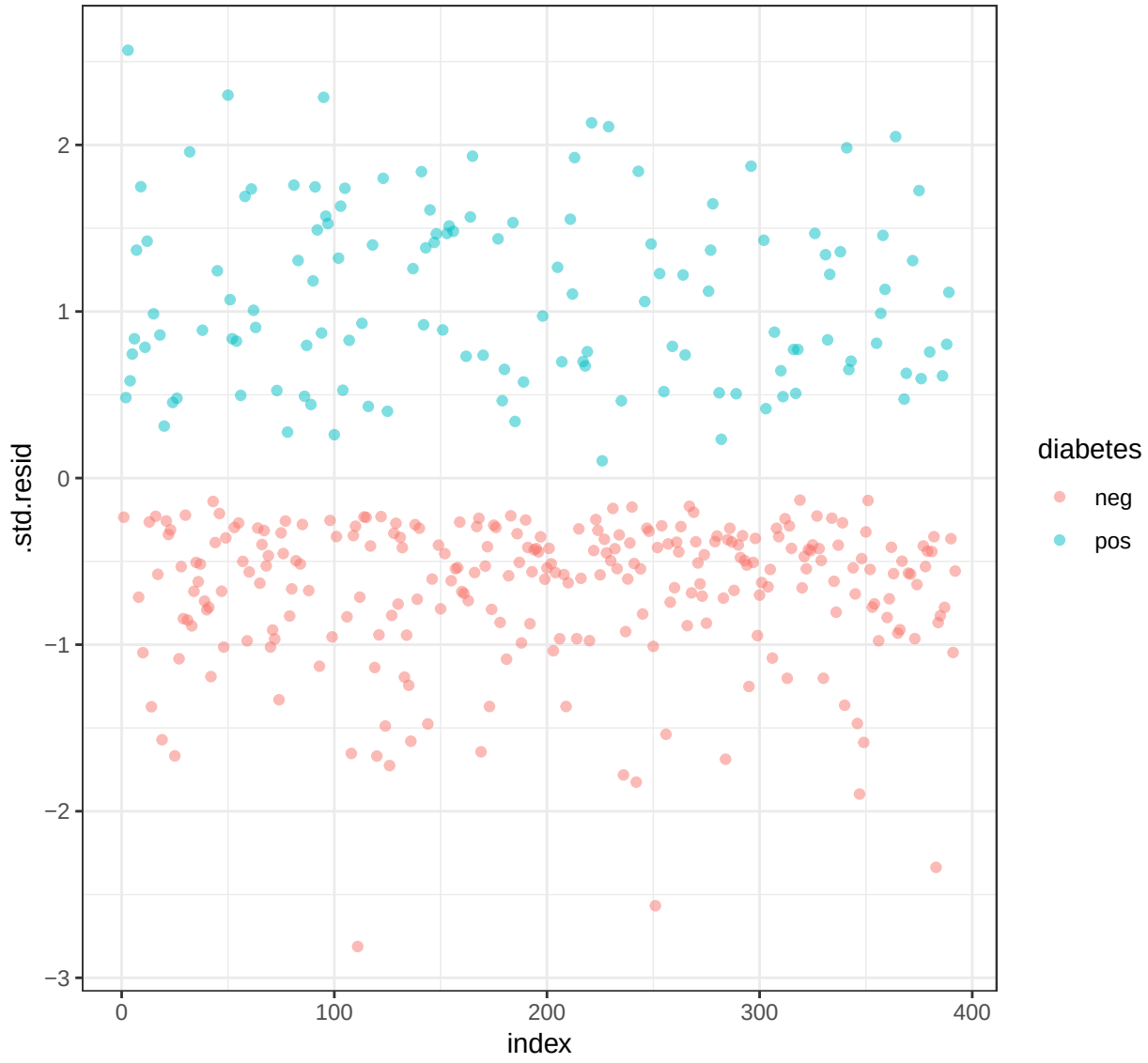



```
# Extract model results
model.data <- augment(model) %>% mutate(index = 1:n())

# augment: add columns to the original data that was modeled
# info on what is augmented:
# https://rdrr.io/cran/broom/man/augment.glm.html

model.data %>% top_n(3, .cooksd)
```

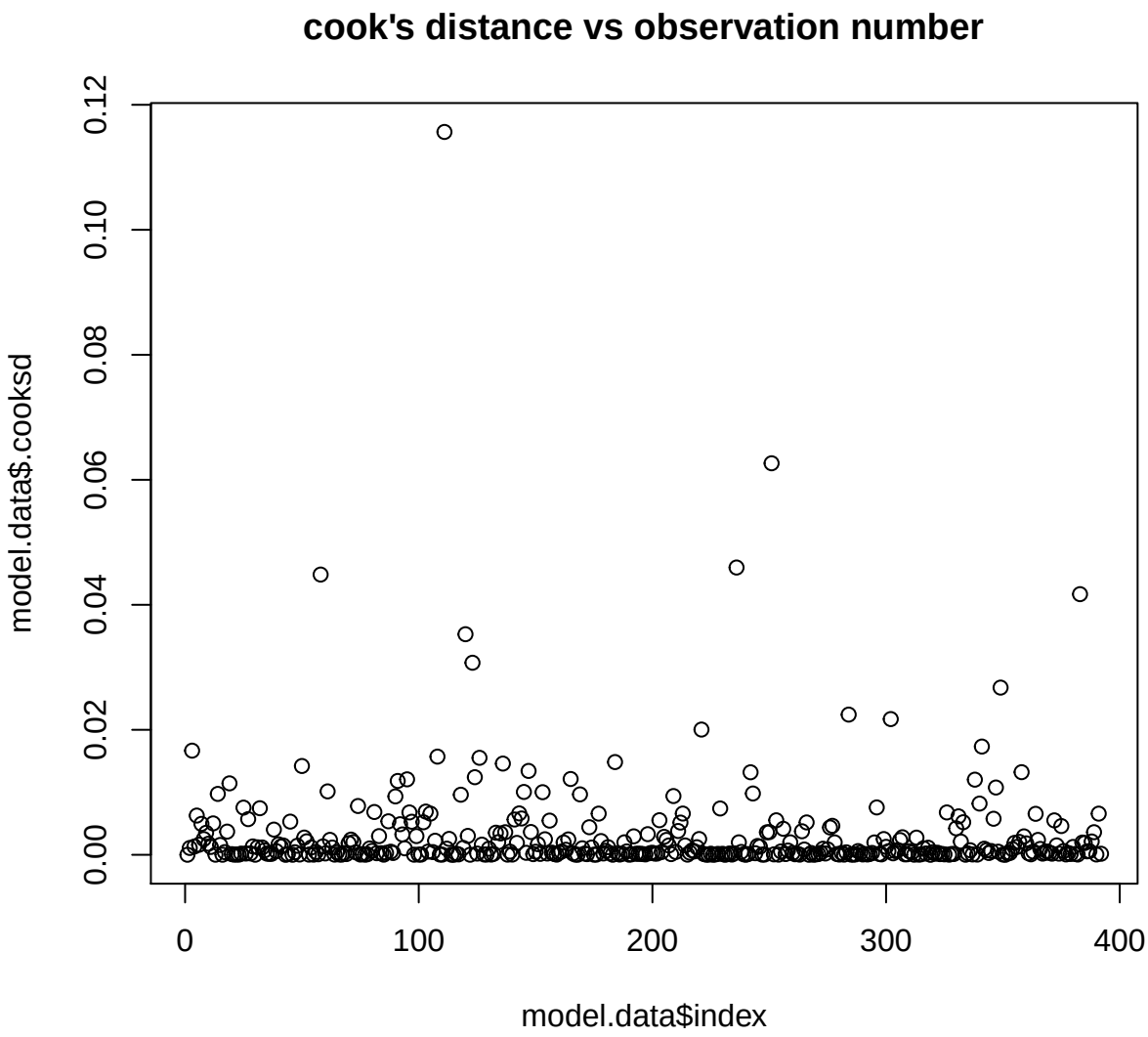
```
## # A tibble: 3 x 18
##   .rownames diabetes pregnant glucose pressure triceps insulin  mass
##   <chr>     <fct>       <dbl>   <dbl>    <dbl>   <dbl>   <dbl> <dbl>
## 1 229       neg             4     197       70      39     744  36.7
## 2 460       neg             9     134       74      33      60  25.9
## 3 488       neg             0     173       78      32     265  46.5
```

```
## # ... with 10 more variables: pedigree <dbl>, age <dbl>, .fitted <dbl>,
## #   .se.fit <dbl>, .resid <dbl>, .hat <dbl>, .sigma <dbl>, .cooksd <dbl>,
## #   .std.resid <dbl>, index <int>
```

```
ggplot(model.data, aes(index, .std.resid)) + geom_point(aes(color = diabetes),
    alpha = 0.5) + theme_bw()
```

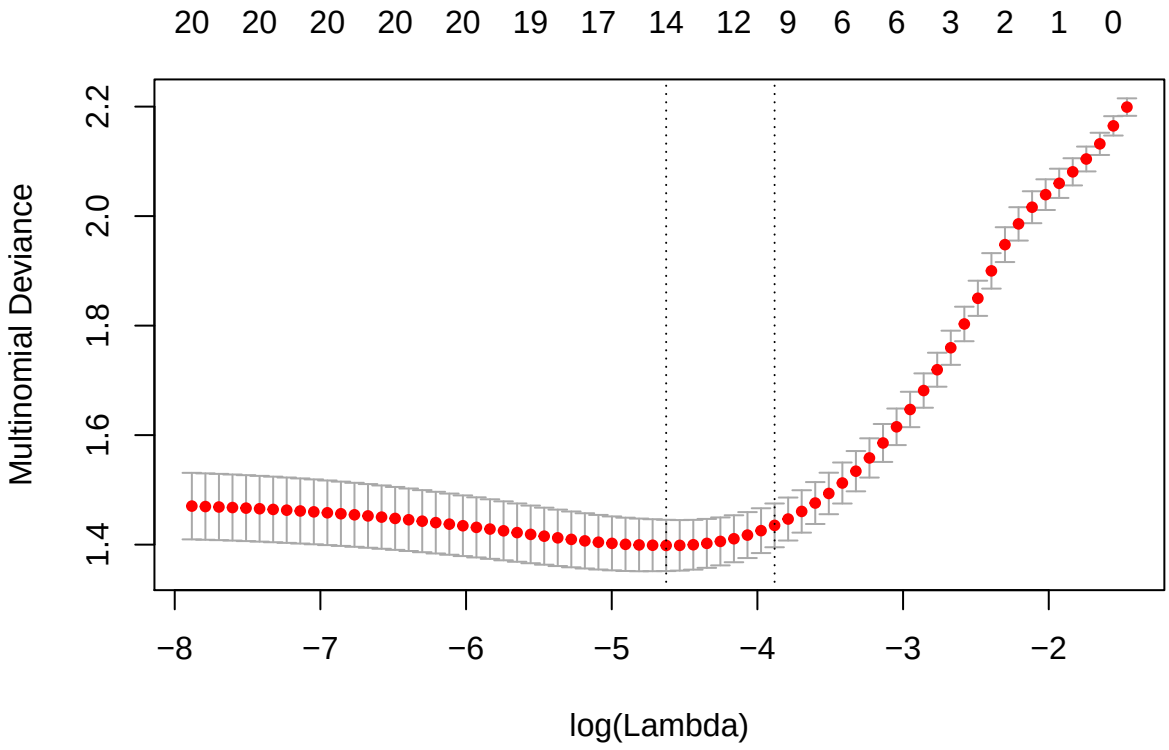



```
## plot cook's distance vs estimated probabilities
plot(model.data$index, model.data$.cooksd, main = "cook's distance vs observation number")
```



Diagnostics for multinomial logistic regression

```
library(glmnet)
par(mfrow = c(1, 1))
load("MultinomialExample.RData")

# find the best lambda for lasso
cvfit = cv.glmnet(x, y, family = "multinomial", alpha = 1, parallel = TRUE)
plot(cvfit)
```




```
# extract fitted model
cvmd = cvfit$glmnet.fit

# extract estimated ceofficients
tmp_coeffs <- coef(cvfit, s = "lambda.min")
tmp_coeffs[[1]]@x
```

```
##  [1]  0.02718177 -0.16249288  0.59576506 -0.16746565 -0.23380570
##  [6]  0.34036004  0.30535554  0.19276380 -0.04516063  0.11571243
## [11]  0.03428813  0.10523164  0.04696864 -0.00718863
```

```
# obtain predictions Type ''class'' applies only to
# ''binomial'' or ''multinomial'' models, and produces the
# class label corresponding to the maximum probability.

prdProb = predict(cvfit, newx = x, s = "lambda.min", type = "response")
prdClass = predict(cvfit, newx = x, s = "lambda.min", type = "class")
```

```
library(glmnet)
library(ggplot2)
head(iris)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
```

```
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

```
tempcv <- cv.glmnet(x = as.matrix(iris[, -5]), y = iris[, 5],
    family = "multinomial", nfolds = 20, alpha = 1)
coefsMin <- coef(tempcv, s = "lambda.min")
# show coefficients
coefsMin
```

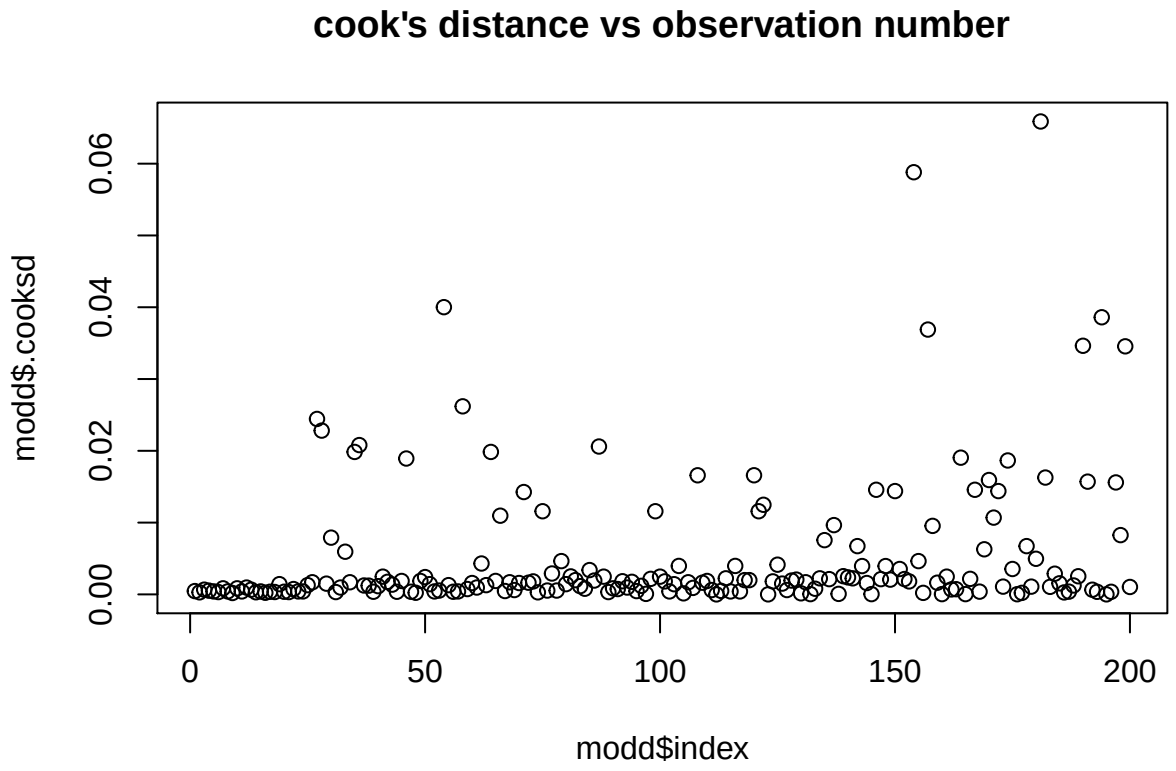
```
## $setosa
## 5 x 1 sparse Matrix of class "dgCMatrix"
##                       1
## (Intercept)  12.336906
## Sepal.Length  .
## Sepal.Width   3.636588
## Petal.Length -2.822105
## Petal.Width  -1.856026
##
## $versicolor
## 5 x 1 sparse Matrix of class "dgCMatrix"
##                      1
## (Intercept)  6.118728
## Sepal.Length 1.535396
## Sepal.Width  .
## Petal.Length .
## Petal.Width  .
##
## $virginica
## 5 x 1 sparse Matrix of class "dgCMatrix"
##                        1
## (Intercept)  -18.455633
## Sepal.Length   .
## Sepal.Width   -3.551571
## Petal.Length   5.424302
## Petal.Width   10.603935
```

Poisson regression

```
require(ggplot2)

p <- read.csv("poisson_sim.csv")
p$prog <- factor(p$prog, levels = 1:3, labels = c("General",
    "Academic", "Vocational"))
summary(p)
```

```
##        id           num_awards          prog          math      
##  Min.   :  1.00   Min.   :0.00   General   : 45   Min.   :33.00  
##  1st Qu.: 50.75   1st Qu.:0.00   Academic  :105   1st Qu.:45.00  
##  Median :100.50   Median :0.00   Vocational: 50   Median :52.00  
##  Mean   :100.50   Mean   :0.63                    Mean   :52.65  
##  3rd Qu.:150.25   3rd Qu.:1.00                    3rd Qu.:59.00  
##  Max.   :200.00   Max.   :6.00                    Max.   :75.00
```

```r
m1 <- glm(num_awards ~ prog + math, family = "poisson", data = p)



# Extract model results
modd <- augment(m1) %>% mutate(index = 1:n())

modd$phat <- predict(m1, type = "response")


par(mfrow = c(1, 1))  # row-column
## plot cook's distance vs estimated probabilities
plot(modd$index, modd$.cooksd, main = "cook's distance vs observation number")
```



```
# plot residuals versus fitted values add a lowess fit

# plot logit of estimated probabilities against predictors
```