LogisticRegression

Chase Enzweiler

10/23/2017

```
# credit default data
library(ISLR)
```

```
## Warning: package 'ISLR' was built under R version 3.3.2
```

```
library(FactoMineR)
```

```
## Warning: package 'FactoMineR' was built under R version 3.3.2
```

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.3.2
```

Run logistic regression

```
# logistic regression

n <- dim(Default)[1]

p <- dim(Default)[2]


# logistic regression
log_reg_default <- glm(default ~ balance, family = binomial, data = Default)

summary(log_reg_default)$coefficients
```

```
##                  Estimate   Std. Error   z value      Pr(>|z|)
## (Intercept) -10.651330614 0.3611573721 -29.49221 3.623124e-191
## balance       0.005498917 0.0002203702  24.95309 1.976602e-137
```

use predict() to obtain the probability of default for individuals with balance of 100, 200, . . . , 2000

```
# predictions

# new data the balances
new_data <- data.frame(seq(100, 2000, 100))
colnames(new_data) <- c("balance")

predict(log_reg_default, newdata = new_data, type = "response" )
```

```
##            1            2            3            4            5
## 4.101880e-05 7.108613e-05 1.231905e-04 2.134779e-04 3.699132e-04
##            6            7            8            9           10
## 6.409100e-04 1.110217e-03 1.922514e-03 3.327154e-03 5.752145e-03
##           11           12           13           14           15
## 9.926984e-03 1.707982e-02 2.923441e-02 4.960213e-02 8.294762e-02
##           16           17           18           19           20
## 1.355136e-01 2.136317e-01 3.201070e-01 4.493274e-01 5.857694e-01
```

Now fit a logistic regression model by regressing default on student. Intepret the coefficient estimate

```r
# default of student

log_reg_stu <- glm(default ~ student, family = binomial, data = Default)

summary(log_reg_stu)$coefficients
```

```
##              Estimate Std. Error    z value     Pr(>|z|)
## (Intercept) -3.5041278 0.07071301 -49.554219 0.0000000000
## studentYes   0.4048871 0.11501883   3.520181 0.0004312529
```

our coefficient estimate for studentYes is positive and therefore the students have a better chance at defaulting than non-students   Now regress default on student, balance, and income.

```r
# logistic regression on student balance and income

log_reg_bsi <- glm(default ~ balance + student + income, family = binomial, data = Default)

summary(log_reg_bsi)$coefficients
```

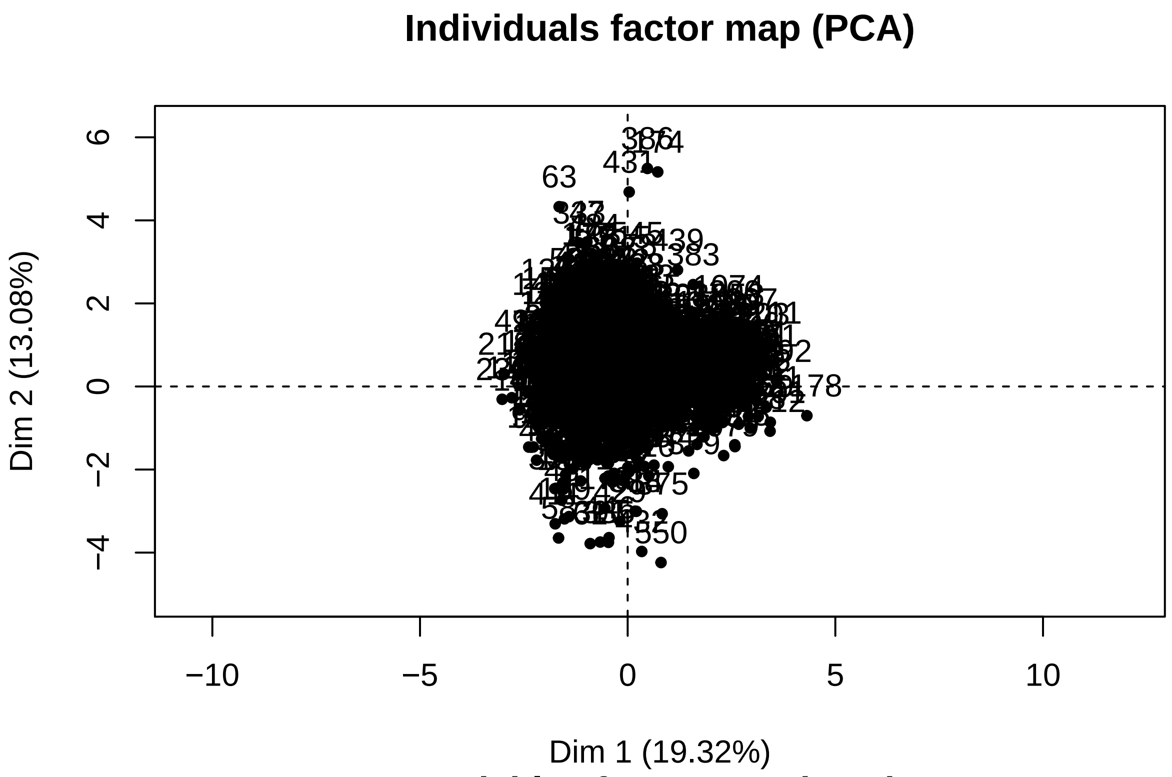
```
##                  Estimate   Std. Error    z value      Pr(>|z|)
## (Intercept) -1.086905e+01 4.922555e-01 -22.080088 4.911280e-108
## balance      5.736505e-03 2.318945e-04  24.737563 4.219578e-135
## studentYes  -6.467758e-01 2.362525e-01  -2.737646  6.188063e-03
## income       3.033450e-06 8.202615e-06   0.369815  7.115203e-01
```

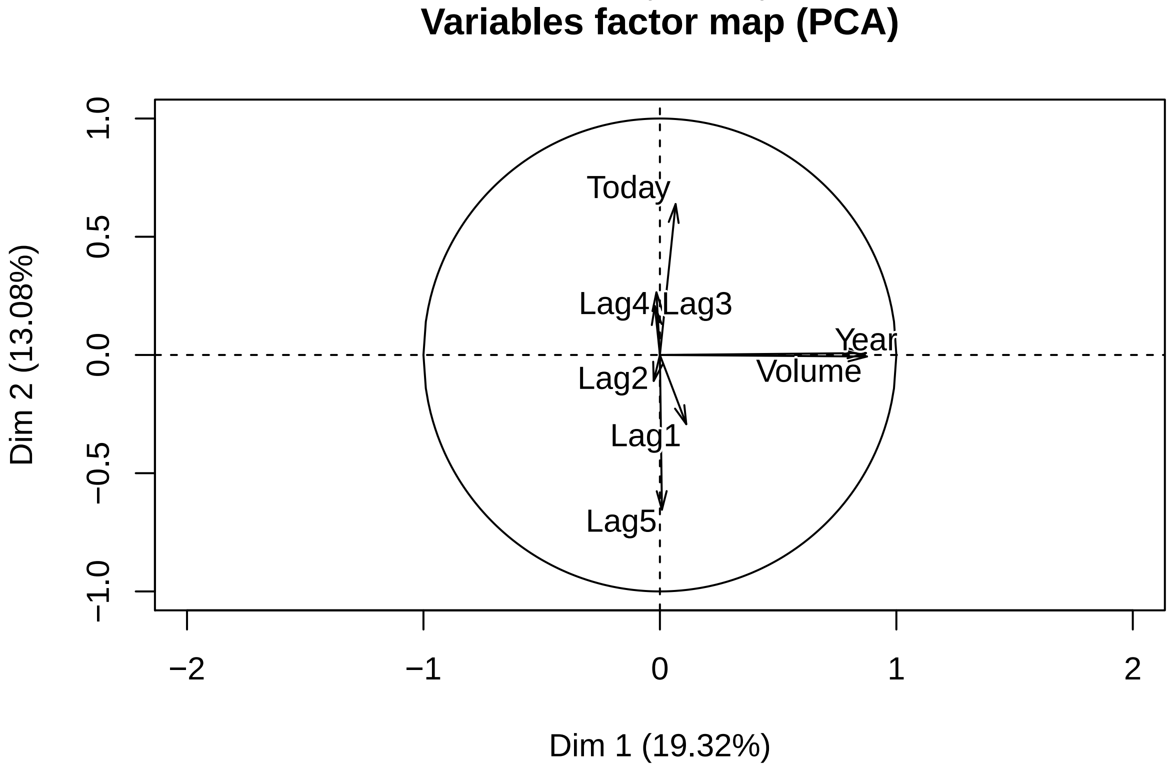
NO not all coefficients are significant, balance and income have large p values.   The contradiction between the coefficient for studentYes in this model and studentYes in the last model is that overall from the last model the students are more likely to default then the non-students. However in the new model, it tells us that given a paticular income and balance a student is less likely to default than a non-student. Confounding effect.

Stock Market data

```r
# PCA on stockmarket data
cor_mat <- cor(Smarket[,-9])

PCA(Smarket[,-9])
```

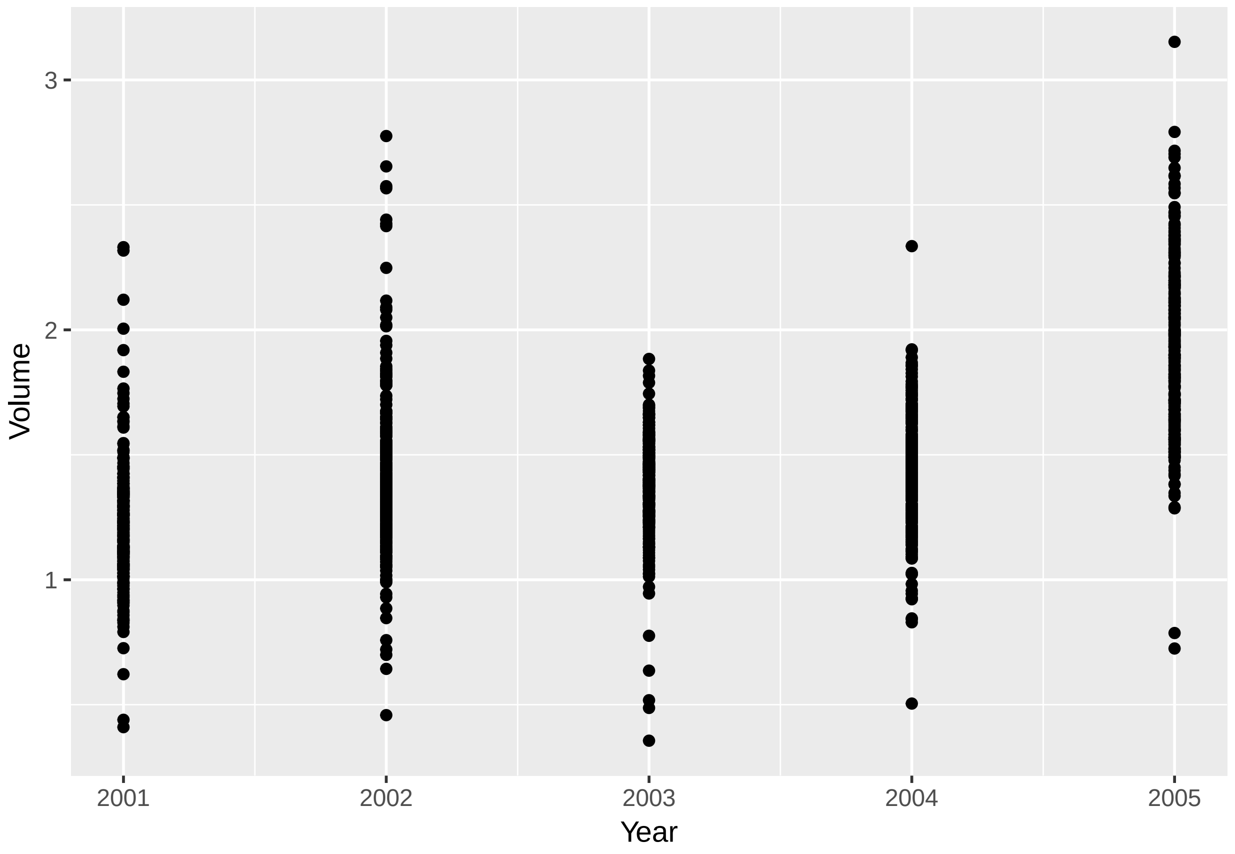




```
## **Results for the Principal Component Analysis (PCA)**
## The analysis was performed on 1250 individuals, described by 8 variables
## *The results are available in the following objects:
##
##    name               description
## 1  "$eig"             "eigenvalues"
## 2  "$var"             "results for the variables"
## 3  "$var$coord"       "coord. for the variables"
## 4  "$var$cor"         "correlations variables - dimensions"
## 5  "$var$cos2"        "cos2 for the variables"
## 6  "$var$contrib"     "contributions of the variables"
## 7  "$ind"             "results for the individuals"
## 8  "$ind$coord"       "coord. for the individuals"
## 9  "$ind$cos2"        "cos2 for the individuals"
## 10 "$ind$contrib"     "contributions of the individuals"
## 11 "$call"            "summary statistics"
## 12 "$call$centre"     "mean of the variables"
## 13 "$call$ecart.type" "standard error of the variables"
## 14 "$call$row.w"      "weights for the individuals"
## 15 "$call$col.w"      "weights for the variables"
```

the lag variables are not very well correlated with the today variable and they have correlations close to zero. The previous days returns are not well correlated with todays returns.

```r
# make a scatter plot of the year and the volume
ggplot(data = Smarket, aes(Year, Volume)) + geom_point()
```



Logistic Regression

```r
# logistic regression direction on lags and volume
```

```
market_log_reg <- glm(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume, family = binomial, data = S

summary(market_log_reg)
```

```
##
## Call:
## glm(formula = Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 +
##     Volume, family = binomial, data = Smarket)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.446  -1.203   1.065   1.145   1.326
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.126000   0.240736  -0.523    0.601
## Lag1        -0.073074   0.050167  -1.457    0.145
## Lag2        -0.042301   0.050086  -0.845    0.398
## Lag3         0.011085   0.049939   0.222    0.824
## Lag4         0.009359   0.049974   0.187    0.851
## Lag5         0.010313   0.049511   0.208    0.835
## Volume       0.135441   0.158360   0.855    0.392
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1731.2  on 1249  degrees of freedom
## Residual deviance: 1727.6  on 1243  degrees of freedom
## AIC: 1741.6
##
## Number of Fisher Scoring iterations: 3
```

Looking at the p values of the coefficients we can see that none of the coefficients are statistically significant. The coefficient of Lag1 is -0.073074, we interpret this as the market is more likely to go down when Lag1 gets good returns???

```
# predict
predict(market_log_reg, newdata = Smarket, type = "response")[1:10]
```

```
##         1         2         3         4         5         6         7
## 0.5070841 0.4814679 0.4811388 0.5152224 0.5107812 0.5069565 0.4926509
##         8         9        10
## 0.5092292 0.5176135 0.4888378
```

### Estimation of Parameters

```
# Newton Raphson Method
library(MASS)
```

```
## Warning: package 'MASS' was built under R version 3.3.2
```

```
Newt_Raph <- function(formula, response , data){

  # get column vector of the response
  y <- response
```

```r
  # create design matrix (n x p+1)
  X <- model.matrix(formula, data = data)

  n <- dim(X)[1]

  p <- dim(X)[2]

  # old coefficient estimates vector originally set to zero
  B_old <- rep(1, p)

  # create variable of vector of new coefficients
  B_new <- rep(0, p)


  # iterate until B_old and B_new are "close"
  while (!identical(round(B_old, digits = 2), round(B_new, digits = 2))){

    # set B_old to B_new

    B_old <- B_new

    # compute p, the n vector of fitted probabilites with ith element
    # of p(xi, B_old), transposes in equation dont translate

    # fitted probabilities vector length n
    P <- c()

    # elements of diagonal of W, vector length n
    W_elem <- c()

    for (i in 1 : n){

      P[i] <- exp( X[i,] %*% B_old ) / ( 1 + exp( X[i,] %*% B_old ))

      # compute the i elements for the diagonal of matrix W

      W_elem[i] <- P[i] * (1 - P[i])

    }

    # calculate z

    z <- (X %*% B_old) + solve(diag(W_elem)) %*% (y - P)

    # calculate our new estimated coefficients
    B_new <- solve(t(X) %*% diag(W_elem) %*% X) %*% t(X) %*% diag(W_elem) %*% z

  }

  return(list("coefficients" = B_new))

}
```

```
market <- Smarket

# convert UP/Down to 1's and 0's
market$Direction <- as.numeric(Smarket$Direction)-1

Newt_Raph(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume, response = market$Direction, data = ma
```

```
## $coefficients
##                     [,1]
## (Intercept) -0.126000259
## Lag1        -0.073073747
## Lag2        -0.042301345
## Lag3         0.011085108
## Lag4         0.009358938
## Lag5         0.010313069
## Volume       0.135440661
```

Simple Newton Raphson

```
# simple newton raphson
Simple_Newt_Raph <- function(formula, response , data){

  # get column vector of the response
  y <- response

  # create design matrix (n x p+1)
  X <- model.matrix(formula, data = data)

  n <- dim(X)[1]

  p <- dim(X)[2]

  # old coefficient estimates vector originally set to zero
  B_old <- rep(1, p )

  # create variable of vector of new coefficients
  B_new <- rep(0, p )

  X_tilda <- matrix(0, ncol = p, nrow = n)

  # iterate until B_old and B_new are "close"
  while (!identical(round(B_old, digits = 2), round(B_new, digits = 2))){

    # set B_old to B_new

    B_old <- B_new

    # compute p, the n vector of fitted probabilites with ith element
    # of p(xi, B_old), transposes in equation dont translate

    # fitted probabilities vector length n
    P <- c()
    # elements of diagonal of W, vector length n
    W_elem <- c()

    for (i in 1 : n){

      P[i] <- exp( X[i,] %*% B_old ) / ( 1 + exp( X[i,] %*% B_old ))

      # compute the i elements for the diagonal of matrix W

      W_elem[i] <- P[i] * (1 - P[i])

      X_tilda[i, ] <- X[i, ] * W_elem[i]

    }




    # calculate our new estimated coefficients
    B_new <- B_old + solve(t(X) %*% X_tilda) %*% t(X) %*% (y - P)

   }

  return(list("coefficients" = B_new))

}

# says t(X) %*% X is computationally singular

Simple_Newt_Raph(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume, response = market$Direction, da
```

```
## $coefficients
##                [,1]
## [1,] -0.126000259
## [2,] -0.073073747
## [3,] -0.042301345
## [4,]  0.011085108
## [5,]  0.009358938
## [6,]  0.010313069
## [7,]  0.135440661
```