154Lab8

Jiyoon Clover Jeong

10/23/2017

```
library(ISLR)
library(ggplot2)
library(FactoMineR)

names(Default)
```

```
## [1] "default" "student" "balance" "income"
```

```
dim(Default)
```

```
## [1] 10000     4
```

```
summary(Default)
```

```
##  default    student       balance           income     
##  No :9667   No :7056   Min.   :   0.0   Min.   :  772  
##  Yes: 333   Yes:2944   1st Qu.: 481.7   1st Qu.:21340  
##                        Median : 823.6   Median :34553  
##                        Mean   : 835.4   Mean   :33517  
##                        3rd Qu.:1166.3   3rd Qu.:43808  
##                        Max.   :2654.3   Max.   :73554
```

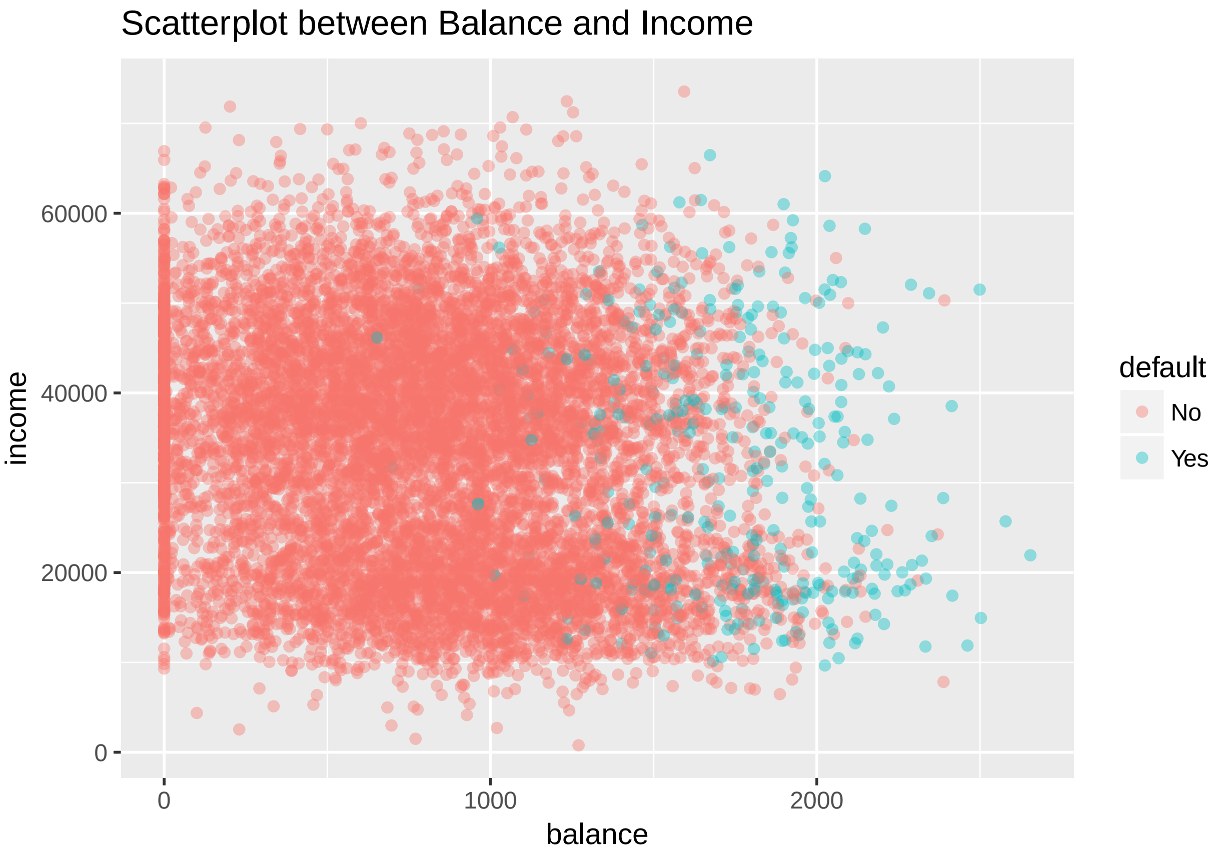
```
summary(subset(Default, default == 'Yes'))
```

```
##  default    student      balance           income     
##  No :  0   No :206   Min.   : 652.4   Min.   : 9664  
##  Yes:333   Yes:127   1st Qu.:1511.6   1st Qu.:19028  
##                      Median :1789.1   Median :31515  
##                      Mean   :1747.8   Mean   :32089  
##                      3rd Qu.:1988.9   3rd Qu.:43067  
##                      Max.   :2654.3   Max.   :66466
```

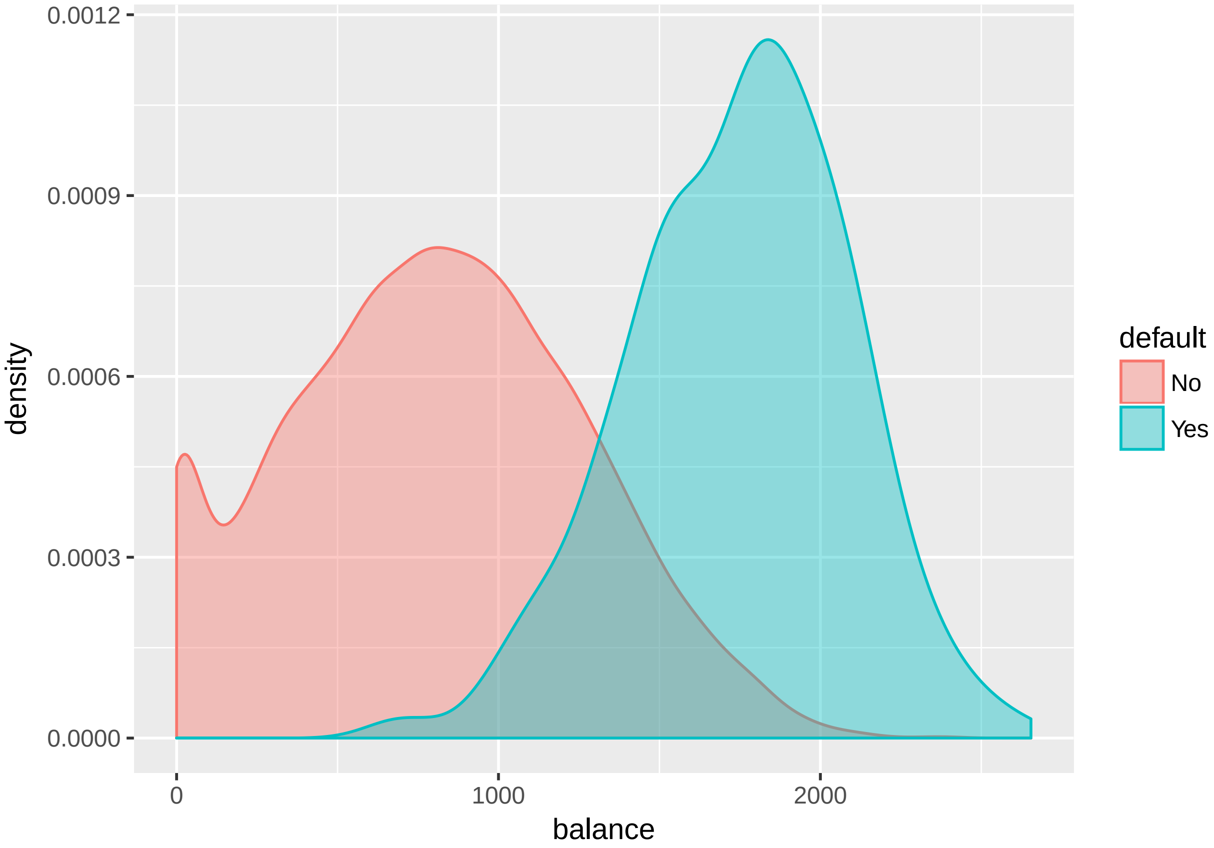
```
summary(subset(Default, default == 'No'))
```

```
##  default    student       balance           income     
##  No :9667   No :6850   Min.   :   0.0   Min.   :  772  
##  Yes:   0   Yes:2817   1st Qu.: 465.7   1st Qu.:21405  
##                        Median : 802.9   Median :34589  
##                        Mean   : 803.9   Mean   :33566  
##                        3rd Qu.:1128.2   3rd Qu.:43824  
##                        Max.   :2391.0   Max.   :73554
```

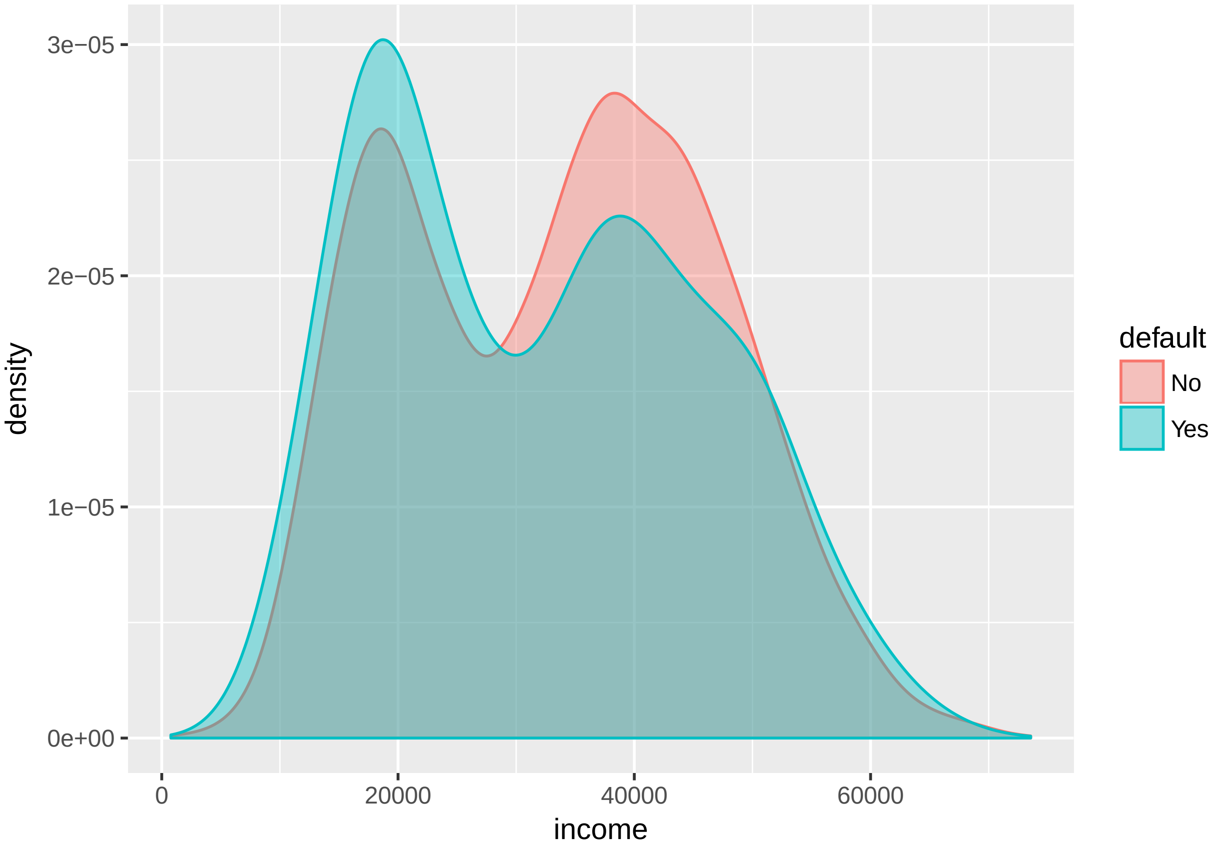
```
ggplot(data = Default, aes(x = balance, y = income, color = default)) + geom_point(alpha = 0.4) + labs(
```



```
ggplot(data = Default, aes(x = balance, fill = default, color = default)) +  geom_density(alpha = 0.4)
```



```
ggplot(data = Default, aes(x = income, fill = default, color = default)) + geom_density(alpha = 0.4)
```



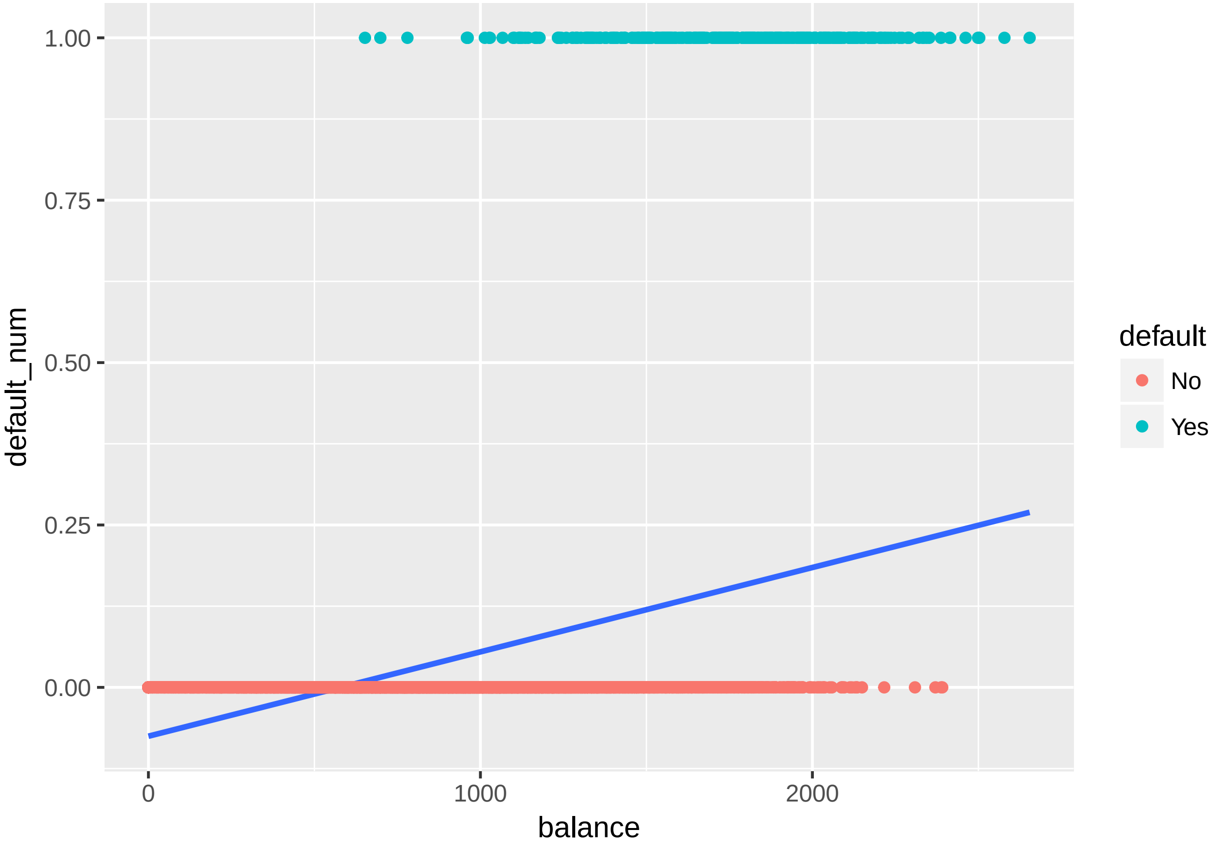
OLS Regression

```
default_numeric <- rep(0, nrow(Default))
default_numeric[Default$default == 'Yes'] <- 1
Default$default_num <- default_numeric
ols_reg <- lm(default_num ~ balance, data = Default)

summary(ols_reg)
```

```
##
## Call:
## lm(formula = default_num ~ balance, data = Default)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.23533 -0.06939 -0.02628  0.02004  0.99046
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -7.519e-02  3.354e-03  -22.42   <2e-16 ***
## balance      1.299e-04  3.475e-06   37.37   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1681 on 9998 degrees of freedom
## Multiple R-squared:  0.1226, Adjusted R-squared:  0.1225
## F-statistic:  1397 on 1 and 9998 DF,  p-value: < 2.2e-16
```

```
# Q
ggplot(data = Default, aes(x = balance, y = default_num)) + geom_smooth(method = "lm", se = F) + geom_po
```



```
#aes(x = Default$balance, y = Default$default_num)
```

Logistic Regression

```
logreg_default <- glm(default ~ balance, family = binomial, data = Default)
summary(logreg_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
```

```
logreg_default
```

```
##
## Call:  glm(formula = default ~ balance, family = binomial, data = Default)
##
## Coefficients:
## (Intercept)      balance
##  -10.651331     0.005499
##
## Degrees of Freedom: 9999 Total (i.e. Null);  9998 Residual
## Null Deviance:       2921
## Residual Deviance: 1596  AIC: 1600
```

```
newdata = data.frame(balance = seq(100,2000,100))
predict(logreg_default, newdata, 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
```

```
logreg_default <- glm(default ~ student, family = binomial, data = Default)
summary(logreg_default)
```

```
##
## Call:
## glm(formula = default ~ student, family = binomial, data = Default)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.2970  -0.2970  -0.2434  -0.2434   2.6585
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.50413    0.07071  -49.55  < 2e-16 ***
## studentYes   0.40489    0.11502    3.52 0.000431 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2920.6  on 9999  degrees of freedom
## Residual deviance: 2908.7  on 9998  degrees of freedom
## AIC: 2912.7
##
## Number of Fisher Scoring iterations: 6
```

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

```
##
## Call:
## glm(formula = default ~ balance + student + income, family = binomial,
##     data = Default)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4691  -0.1418  -0.0557  -0.0203   3.7383
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.087e+01  4.923e-01 -22.080  < 2e-16 ***
## balance      5.737e-03  2.319e-04  24.738  < 2e-16 ***
## studentYes  -6.468e-01  2.363e-01  -2.738  0.00619 **
## income       3.033e-06  8.203e-06   0.370  0.71152
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2920.6  on 9999  degrees of freedom
## Residual deviance: 1571.5  on 9996  degrees of freedom
## AIC: 1579.5
##
## Number of Fisher Scoring iterations: 8
```

```
print("income coefficient is not significant")
```

```
## [1] "income coefficient is not significant"
```

The Stock Market Smarket Data

```
names(Smarket)
```

```
## [1] "Year"      "Lag1"      "Lag2"      "Lag3"      "Lag4"      "Lag5"
## [7] "Volume"    "Today"     "Direction"
```

```
dim(Smarket)
```

```
## [1] 1250    9
```

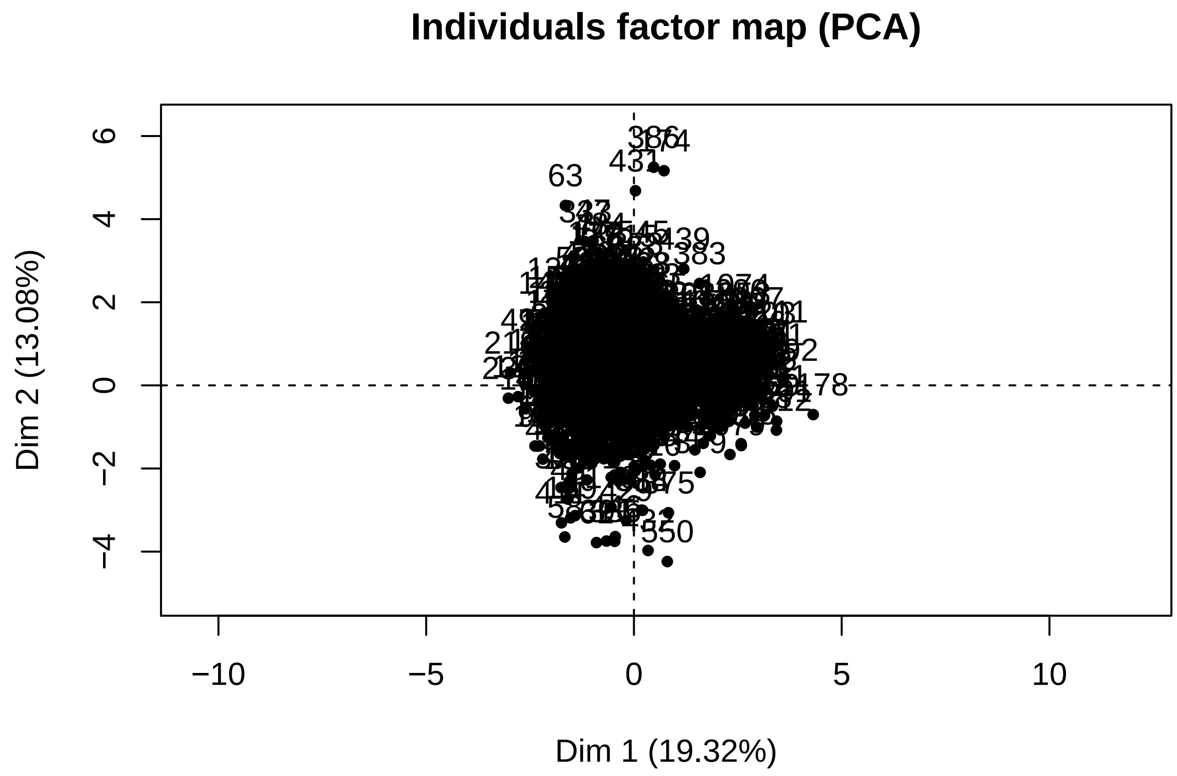
```
summary(Smarket)
```

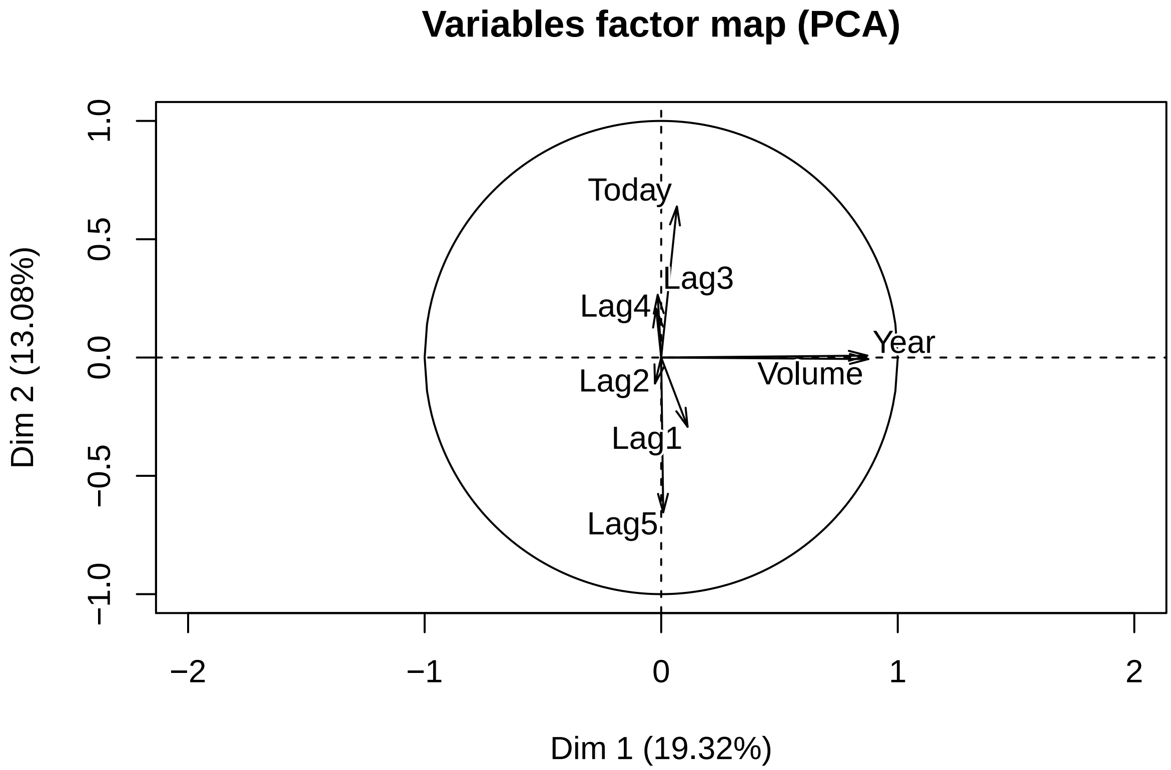
```
##       Year           Lag1                Lag2
##  Min.   :2001   Min.   :-4.922000   Min.   :-4.922000
##  1st Qu.:2002   1st Qu.:-0.639500   1st Qu.:-0.639500
##  Median :2003   Median : 0.039000   Median : 0.039000
##  Mean   :2003   Mean   : 0.003834   Mean   : 0.003919
##  3rd Qu.:2004   3rd Qu.: 0.596750   3rd Qu.: 0.596750
##  Max.   :2005   Max.   : 5.733000   Max.   : 5.733000
##       Lag3                Lag4                Lag5
##  Min.   :-4.922000   Min.   :-4.922000   Min.   :-4.92200
##  1st Qu.:-0.640000   1st Qu.:-0.640000   1st Qu.:-0.64000
##  Median : 0.038500   Median : 0.038500   Median : 0.03850
##  Mean   : 0.001716   Mean   : 0.001636   Mean   : 0.00561
##  3rd Qu.: 0.596750   3rd Qu.: 0.596750   3rd Qu.: 0.59700
##  Max.   : 5.733000   Max.   : 5.733000   Max.   : 5.73300
##      Volume           Today             Direction
##  Min.   :0.3561   Min.   :-4.922000   Down:602
##  1st Qu.:1.2574   1st Qu.:-0.639500   Up  :648
##  Median :1.4229   Median : 0.038500
##  Mean   :1.4783   Mean   : 0.003138
##  3rd Qu.:1.6417   3rd Qu.: 0.596750
##  Max.   :3.1525   Max.   : 5.733000
```

```
cor <- cor(Smarket[,-9])
cor
```

```
##              Year         Lag1         Lag2         Lag3         Lag4
## Year   1.00000000  0.029699649  0.030596422  0.033194581  0.035688718
## Lag1   0.02969965  1.000000000 -0.026294328 -0.010803402 -0.002985911
## Lag2   0.03059642 -0.026294328  1.000000000 -0.025896670 -0.010853533
## Lag3   0.03319458 -0.010803402 -0.025896670  1.000000000 -0.024051036
## Lag4   0.03568872 -0.002985911 -0.010853533 -0.024051036  1.000000000
## Lag5   0.02978799 -0.005674606 -0.003557949 -0.018808338 -0.027083641
## Volume 0.53900647  0.040909908 -0.043383215 -0.041823686 -0.048414246
## Today  0.03009523 -0.026155045 -0.010250033 -0.002447647 -0.006899527
##                Lag5       Volume        Today
## Year    0.029787995  0.53900647  0.030095229
## Lag1   -0.005674606  0.04090991 -0.026155045
## Lag2   -0.003557949 -0.04338321 -0.010250033
## Lag3   -0.018808338 -0.04182369 -0.002447647
## Lag4   -0.027083641 -0.04841425 -0.006899527
## Lag5    1.000000000 -0.02200231 -0.034860083
## Volume -0.022002315  1.00000000  0.014591823
## Today  -0.034860083  0.01459182  1.000000000
```

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

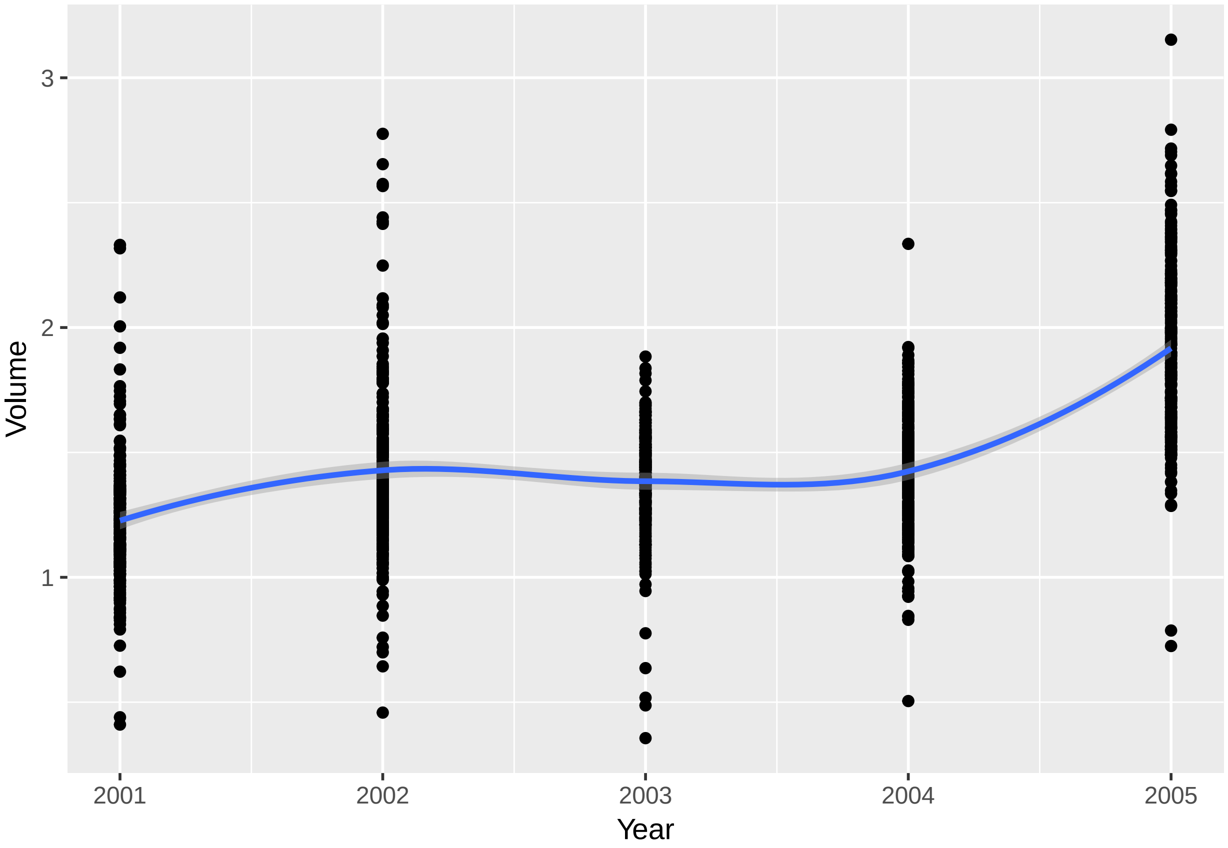




```
## **Results for the Principal Component Analysis (PCA)**
## The analysis was performed on 1250 individuals, described by 8 variables
## *The results are available in the following objects:
##
##    name               description
## 1  "$eig"             "eigenvalues"
## 2  "$var"             "results 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"
```

```
# plot(x = Smarket$Year, y = Smarket$Volume)
# lines(lowess(Year,Volume), col="blue")


ggplot(data = Smarket, aes(x = Year, y = Volume)) + geom_point() + geom_smooth(method = loess)
```



Logistic Regression

```
formula <- paste0("Lag", 1:5, collapse = " + ")
formula <- paste("Direction ~", formula, "+ Volume")
fit <- glm(formula, family = binomial, data = Smarket)

summary(fit)
```

```
##
## Call:
## glm(formula = formula, 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
```

```
# Q : if new data ommited, fitted value will be used
head(predict(fit, type = "response"))
```

```
##         1         2         3         4         5         6
## 0.5070841 0.4814679 0.4811388 0.5152224 0.5107812 0.5069565
```

Lag1 seems to be the most significant. The coefficient of Lag1 is -0.073074 and has nagative sign.

Estimation of Parameters

• Let y be the column vector of response Y • Let X be the n × (p + 1) input (design) matrix • Let p be the n-vector of fitted probabilities with the i-th element $p(xi, \beta^{old})$ • Let W be an n×n diagonal matrix of weights with i-th element $p(xi, \beta^{old})(1 - p(xi, \beta^{old}))$

Newton-Raphson algorithm

```
newdirect <- rep(0, nrow(Smarket))
newdirect[Smarket$Direction == "Up"] <- 1
Smarket$newdirect <- newdirect
n = nrow(Smarket)



y <- as.matrix(Smarket[,10, drop = F])
X <- as.matrix(cbind(1, Smarket[, c(2:7), drop = F]))
p <- c(0)
W <- diag(0,n)
b_old <- matrix(0, ncol(X), 1)
b_new <- matrix(0, ncol(X), 1)
diff <- 10^10

while(diff >10^(-7)){
   b_old <- b_new
  for(j in 1:n){
    p[j] <- exp( X[j,, drop = F] %*% b_old )   /
      (1 + exp( X[j, , drop = F] %*% b_old ) )

    W[j,j] = p[j] * (1- p[j])

  }

  z <- X %*% b_old + solve(W) %*% (y - p)
  b_new <- solve(t(X) %*% W %*% X ) %*% t(X) %*% W %*% z
  diff <- sqrt(sum((b_new - b_old)^2))

}

b_new
```

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

Simplified Algorithm

```
b_old <- matrix(0, ncol(X), 1)
y <- as.matrix(Smarket[,10, drop = F])
X <- as.matrix(cbind(1, Smarket[, c(2:7), drop = F]))
p <- c(0)
diff <- 10^10


while(diff >10^(-7)){
  b_old <- b_new
  for(j in 1:n){
    p[j] <- exp( X[j, , drop = F] %*% b_old )   /
      (1 + exp( X[j, , drop = F] %*% b_old ) )

  }
  X_hat <- sweep(X, MARGIN = 1, p * (1-p), FUN = '*')

  b_new <- b_old + solve(t(X) %*% X_hat) %*% t(X) %*% (y-p)

  diff <- sqrt(sum((b_new - b_old)^2))

}

b_new
```

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