Deep Learning HW1 – 2015129053 김형철

*** I also uploaded the file that has the written code that I used in R just in case.**

1. (a) Using the "stock.csv" data, we will construct the dummy variable matrix using the "model.matrix" function. The code is written as follows:

```
mydata=read.csv("stock.csv", sep=",", header=TRUE)
attach(mydata)
month.f=factor(Month)
dummy=model.matrix(~month.f)
dummy<-dummy[,-1]  #Dropped the first column to make the matrix 24x11 (as we
# are making 11 dummy variables except for January.
dim(dummy)
```

This gives us the 24x11 matrix of dummy variable.

```
> month.f=factor(Month)
> dummy=model.matrix(~month.f)
>
> dummy<-dummy[,-1]
> dim(dummy)
[1] 24 11
```

(b) We will now construct the design matrix X by using "cbind" to combine intercept, interest, unemployment, and month variables. The code is written as follows:

```
X<- cbind(1, Interest, Unemployment, dummy)
dim(X)
```

```
> X<- cbind(1, Interest, Unemployment, dummy)
> dim(X)
[1] 24 14
```

(c) Now we calculate the beta hat and standard error by using the following code:

beta.hat <-solve(t(X)%%X)%*%t(X)%*%y*
sigmasq.hat <- as.numeric(t(y-X%%beta.hat)%*%(y-X%*%beta.hat)/(24-14))*
standard_error=sqrt(diag(solve(t(X)%%X))*sigmasq.hat)*

```
> beta.hat
                    [,1]
            2891.2995444
Interest     182.9886105
Unemployment -389.5671982
month.f2       0.5216401
month.f3      35.6697039
month.f4     -19.2437358
month.f5      32.7129841
month.f6      38.3394077
month.f7     100.3177677
month.f8      92.8394077
month.f9      65.8826879
month.f10     31.9476082
month.f11     80.9259681
month.f12    130.6571754
```

```
> standard_error
                Interest Unemployment     month.f2     month.f3     month.f4     month.f5     month.f6     month.f7
  1918.29009   231.34463    248.82736     84.93946     84.35764     96.28695     87.03185     85.35262     87.18186
    month.f8    month.f9    month.f10    month.f11    month.f12
    85.35262    87.07803    101.73824     95.49031     91.85277
```

(d) Finally, we use the normal lm function to check whether the hand calculation is similar to the answer given by lm function.

model <- lm(Stock ~ Interest + Unemployment + as.factor(Month))
summary(model)

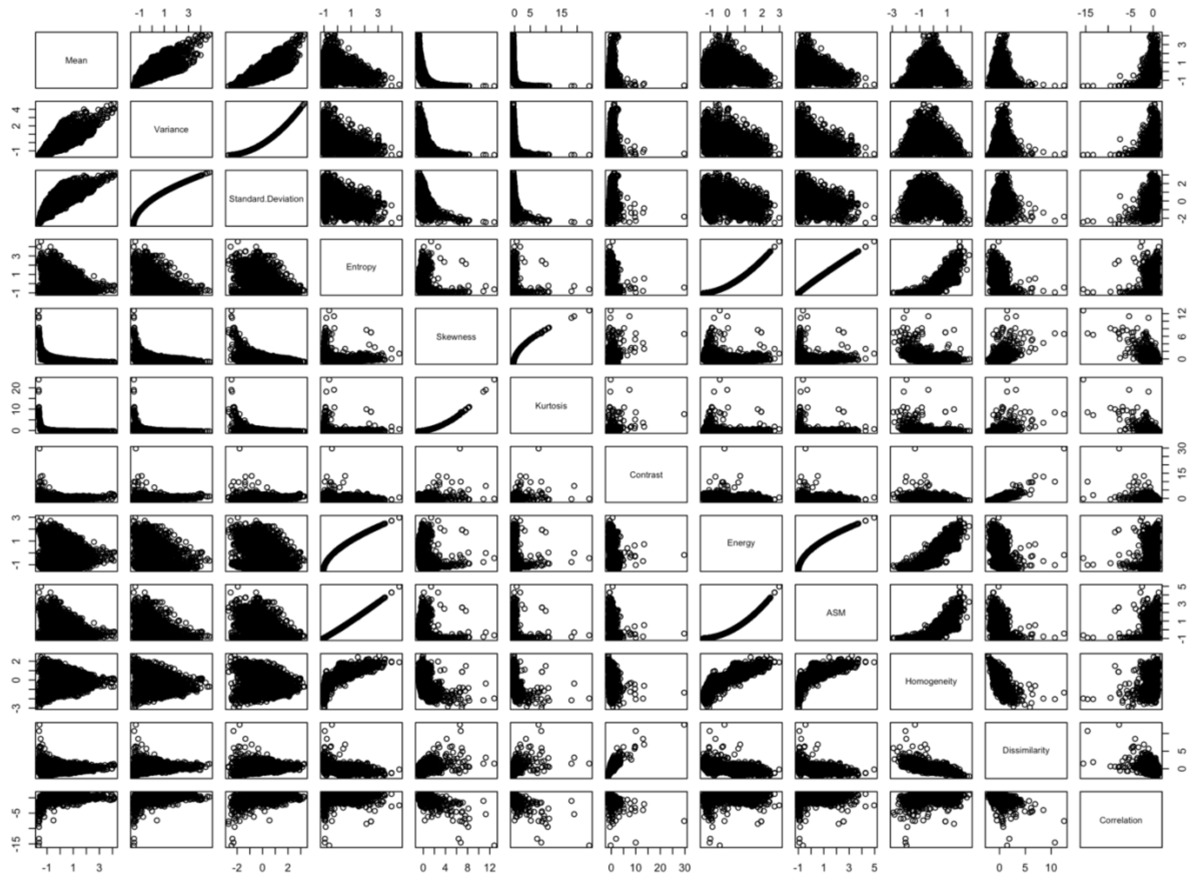
```
Call:
lm(formula = Stock ~ Interest + Unemployment + as.factor(Month))

Residuals:
    Min      1Q  Median      3Q     Max 
-116.17  -20.63    0.00   20.63  116.17 

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)        2891.2995  1918.2901   1.507    0.163
Interest            182.9886   231.3446   0.791    0.447
Unemployment       -389.5672   248.8274  -1.566    0.149
as.factor(Month)2     0.5216    84.9395   0.006    0.995
as.factor(Month)3    35.6697    84.3576   0.423    0.681
as.factor(Month)4   -19.2437    96.2869  -0.200    0.846
as.factor(Month)5    32.7130    87.0319   0.376    0.715
as.factor(Month)6    38.3394    85.3526   0.449    0.663
as.factor(Month)7   100.3178    87.1819   1.151    0.277
as.factor(Month)8    92.8394    85.3526   1.088    0.302
as.factor(Month)9    65.8827    87.0780   0.757    0.467
as.factor(Month)10   31.9476   101.7382   0.314    0.760
as.factor(Month)11   80.9260    95.4903   0.847    0.417
as.factor(Month)12  130.6572    91.8528   1.422    0.185

Residual standard error: 84.02 on 10 degrees of freedom
Multiple R-squared:  0.9309,	Adjusted R-squared:  0.841 
F-statistic: 10.36 on 13 and 10 DF,  p-value: 0.0003926
```

We can easily check that both ways of calculation are similar.

2. (a) After reading the csv file, I used the following codes to get the scatter plots for Each 12 variables:

mydata=read.csv("BrainTumor.csv", sep=",", header=TRUE)
mydata12=mydata[,-1]
scaled=scale(mydata2, center=TRUE, scale=TRUE)
pairs(scaled)

This gives us the following scatter plot for the 12 scaled variables:



Some of the relationships between variables seems to be: 1) there seems to be some positive correlation between “variance” and “standard deviation”. 2) Also there seems to be positive relations between Skewness and Kurtosis although there seems to be some jump in the middle. 3) Also there is positive relations between “Entropy” and “Energy”, “Entropy” and “ASM”. 4) There also seems to be some weak negative relations between variables such as “Mean” and “Entropy” but it is too weak to say for sure.

(b) I divided the data into training and test sets using the following code that does it randomly. **(I wasn’t sure if I was to do regression with the original variables or the scaled ones. Thus, I did both but I will just write here the case in which the variables were not scaled as the code is exactly the same and the answer seems to be okay for both of them. I will post the code using scaled data in the code file that I will separately provide.)**

```
set.seed(100)
sample=sample.int(n = nrow(mydata), size = floor(nrow(mydata)*0.70), replace = FALSE)
train=mydata[sample, ]
test=mydata[-sample, ]
```

This separated the data into two groups: 70% for training group and 30% for testing group.

Now I fit the logistic regression using training set with glm function as follows:

```
fit=glm(Class~Mean+Variance+Standard.Deviation+Entropy+Skewness+Kurtosis+Contrast+Energy+ASM+Homogeneity+Dissimilarity+Correlation,family="binomial",data=train)
summary(fit)
```

```
> summary(fit)

Call:
glm(formula = Class ~ Mean + Variance + Standard.Deviation +
    Entropy + Skewness + Kurtosis + Contrast + Energy + ASM +
    Homogeneity + Dissimilarity + Correlation, family = "binomial",
    data = train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.9322  -0.0320  -0.0032   0.0001   3.9329

Coefficients:
                    Estimate Std. Error z value Pr(>|z|)
(Intercept)       -6.325e+01  1.631e+01  -3.879 0.000105 ***
Mean               8.549e-02  1.115e-01   0.766 0.443412
Variance          -9.148e-03  3.192e-03  -2.866 0.004161 **
Standard.Deviation  6.709e-01  1.820e-01   3.686 0.000228 ***
Entropy           -5.406e+02  7.803e+02  -0.693 0.488411
Skewness           3.765e+00  6.353e-01   5.926 3.11e-09 ***
Kurtosis          -6.093e-02  1.478e-02  -4.124 3.73e-05 ***
Contrast           4.056e-02  7.687e-03   5.277 1.31e-07 ***
Energy            -7.637e+01  8.050e+01  -0.949 0.342754
ASM                7.077e+02  7.816e+02   0.905 0.365215
Homogeneity       -4.219e+01  1.045e+01  -4.038 5.40e-05 ***
Dissimilarity     -3.021e+00  5.465e-01  -5.528 3.24e-08 ***
Correlation        8.823e+01  1.718e+01   5.135 2.82e-07 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 3617.44  on 2632  degrees of freedom
Residual deviance:  197.28  on 2620  degrees of freedom
AIC: 223.28

Number of Fisher Scoring iterations: 11
```

According to the glm summary, it seems Variance, Standard.Deviation, Skewness, Kurtosis, Contrast, Homogeneity, Dissimilarity, Correlation are significant variables.

It seems the variables Variance, Kurtosis, Homogeneity, and Dissimilarity seems to be negatively correlated with the chance of the MRI image being a tumor. So, if these variables become larger in value, the chance of the MRI image being a tumor goes down.

It seems the variables Standard Deviation, Skewness, Contrast, and Correlation is positively correlated with the chance of the MRI image being a tumor. Thus, if these variables become larger, the chance of the MRI image being a tumor goes up.

I am not exactly sure what these variables mean in the MRI image, but perhaps feature variables like "Contrast" might mean that there is stronger contrast of color in the MRI image because of the existence of tumor.

(c) I used the following code to predict the response for the test datasets:

```
testnew=test[,-1] # Using this code to get a matrix that has information of the 12
# variables in the test data(I removed the Class variable to predict).

eta3=predict(fit,newdata=testnew,type="response") # I put the predicted mean
# probability in the variable eta3.
```

I then used the following code to assign 1 if predicted mean probability is greater than equal to 0.5 (otherwise assign 0).

```
tumor_class3=eta3 #This variable is used to hold the values of only 0 and 1.
for (i in 1:1129) {
    if (eta3[i]>=0.5) {
        tumor_class3[i]=1
    } else {
        tumor_class3[i]=0
    }
}
```

Tumor_class3 variable holds the predicted value(0 or 1) made by the fitted glm function. Now we will calculate the prediction accuracy by checking how much of the test data was predicted accurately.

```
real=test[,1]
prediction=tumor_class3

accuracy=real-prediction

accuracy=unname(accuracy)  #This was used to unname the variable.
check=0 # This variable was made to check the number of wrong predictions.

for (i in 1:1129) {
    if (accuracy[i] != 0) {
```

```
    check=check+1
    }
}

calculator=(1129-check)/1129
calculator
```

The calculator variable gives us “0.9840567”. This is because out of 1129 test data, there were only 18 wrong predictions.

Thus, the prediction accuracy of this test data is around 98%.