Exam Chapter 4

Wenduo Wang

July 30, 2016

Problem 10

(a) Produce some numerical and graphical summaries of the Weekly data. Do there appear to be any patterns?

First load the data set from `ISLR` package. And make a summary on `Weekly`. Let's explore if there is a qualitative correlation between the previous X weeks return and the current week's return.

This is done by creating a new data frame called `co_move`, whose columns are `TRUE` if the product of the two returns is positive.

```
library(ISLR)
library(dplyr)
```

```
summary(Weekly)
co_move <- data.frame(lapply(Weekly[, 2:6], function(v) v*Weekly[, 8]>0))
cat("The output below is the probability that the current weekly percentage return is positive given tha
sapply(co_move, function(x) sum(x)/length(x))
```

```
>>>       Year          Lag1               Lag2               Lag3
>>>  Min.   :1990   Min.   :-18.1950   Min.   :-18.1950   Min.   :-18.1950
>>>  1st Qu.:1995   1st Qu.: -1.1540   1st Qu.: -1.1540   1st Qu.: -1.1580
>>>  Median :2000   Median :  0.2410   Median :  0.2410   Median :  0.2410
>>>  Mean   :2000   Mean   :  0.1506   Mean   :  0.1511   Mean   :  0.1472
>>>  3rd Qu.:2005   3rd Qu.:  1.4050   3rd Qu.:  1.4090   3rd Qu.:  1.4090
>>>  Max.   :2010   Max.   : 12.0260   Max.   : 12.0260   Max.   : 12.0260
>>>       Lag4               Lag5              Volume
>>>  Min.   :-18.1950   Min.   :-18.1950   Min.   :0.08747
>>>  1st Qu.: -1.1580   1st Qu.: -1.1660   1st Qu.:0.33202
>>>  Median :  0.2380   Median :  0.2340   Median :1.00268
>>>  Mean   :  0.1458   Mean   :  0.1399   Mean   :1.57462
>>>  3rd Qu.:  1.4090   3rd Qu.:  1.4050   3rd Qu.:2.05373
>>>  Max.   : 12.0260   Max.   : 12.0260   Max.   :9.32821
>>>      Today          Direction
>>>  Min.   :-18.1950   Down:484
>>>  1st Qu.: -1.1540   Up  :605
>>>  Median :  0.2410
>>>  Mean   :  0.1499
>>>  3rd Qu.:  1.4050
>>>  Max.   : 12.0260
>>> The output below is the probability that the current weekly percentage return is positive given that of th
>>> 0.4692378 0.5151515 0.5032140 0.5078053 0.4775023
```

From the result, it is difficult to draw a clear pattern, since it appears the current week's return is remotely related to historic weekly returns. Let's dig deeper by drawing some plots between them.

```
library(ggplot2)
library(gridExtra)
library(reshape2)
```

(b) Use the full data set to perform a logistic regression with Direction as the response and the five lag variables plus Volume as predictors. Use the summary function to print the results. Do any of the predictors appear to be statistically significant? If so, which ones?

A vanilla logistic regression is done on the dataset, and from the model summary, it is seen only Lag2 is statistically significant (if intercept is ignored), as determined by the *p-value*.

```
logistic_fit <- glm(Direction~.-Year-Today, data=Weekly, family=binomial)
cat("The fitted coefficients of the logistic model are listed below.")
coef(logistic_fit)
cat("The response signal created by R in place of Direction is like this.")
contrasts(Weekly$Direction)
cat("The summary of the fitted model is below.")
print(summary(logistic_fit))
prediction_logistic <- sapply(predict(logistic_fit, type="response"), function(x) ifelse(x>0.5, "Up", "I
```

```
>>> The fitted coefficients of the logistic model are listed below.(Intercept)      Lag1      Lag2      Lag3
>>>  0.26686414 -0.04126894  0.05844168 -0.01606114 -0.02779021 -0.01447206
>>>      Volume
>>> -0.02274153
>>> The response signal created by R in place of Direction is like this.     Up
>>> Down  0
>>> Up    1
>>> The summary of the fitted model is below.
>>> Call:
>>> glm(formula = Direction ~ . - Year - Today, family = binomial,
>>>     data = Weekly)
>>>
>>> Deviance Residuals:
>>>     Min       1Q   Median       3Q      Max
>>> -1.6949  -1.2565   0.9913   1.0849   1.4579
>>>
>>> Coefficients:
>>>             Estimate Std. Error z value Pr(>|z|)
>>> (Intercept)  0.26686    0.08593   3.106   0.0019 **
>>> Lag1        -0.04127    0.02641  -1.563   0.1181
>>> Lag2         0.05844    0.02686   2.175   0.0296 *
>>> Lag3        -0.01606    0.02666  -0.602   0.5469
>>> Lag4        -0.02779    0.02646  -1.050   0.2937
>>> Lag5        -0.01447    0.02638  -0.549   0.5833
>>> Volume      -0.02274    0.03690  -0.616   0.5377
>>> ---
>>> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
>>>
>>> (Dispersion parameter for binomial family taken to be 1)
>>>
>>>     Null deviance: 1496.2  on 1088  degrees of freedom
>>> Residual deviance: 1486.4  on 1082  degrees of freedom
>>> AIC: 1500.4
>>>
>>> Number of Fisher Scoring iterations: 4
```

(c) Compute the confusion matrix and overall fraction of correct predictions. Explain what the confusion matrix is telling you about the types of mistakes made by logistic regression.

The confusion matrix is created with `table()` function. Yet before creating the matrix, an extra procedure is necessary to translate the response signal into *Up* or *Down*.

The confusion matrix has four cells. Mistakes are in the cells where the row index and column index are different. So in this case, the most common mistake is when the regression model predicts *Up* while the actual return went *Down* (more than 400 occurrences). In fact, the prediction model is highly biased towards *Up* vs *Down* , where in reality both sides are almost equally likely.

```
cat("The prediction vs actual Direction is tabulated below.")
table(prediction_logistic, Weekly$Direction)
```

```
>>> The prediction vs actual Direction is tabulated below.
>>> prediction_logistic Down  Up
>>>                Down   54  48
>>>                Up    430 557
```

(d) Now fit the logistic regression model using a training data period from 1990 to 2008, with Lag2 as the only predictor. Compute the confusion matrix and the overall fraction of correct predictions for the held out data (that is, the data from 2009 and 2010).

For this problem, the `Weekly` dataset is split into a training set composed of record on and prior to 2008, while the fitted model will be tested on the records from 2009 onwards. When modeling solely on `Lag2`, the prediction quality is similar to the previous example. While in reality the *Up/Down* ratio is $\frac{56+5}{34+9} \approx 1.4$, the prediction gives $\frac{34+56}{9+5} \approx 6.4$, which is equivalent to an overall fraction of correct predictions of $\frac{56+9}{56+9+5+34} = 62.5\%$.

```
training_set <- Weekly[Weekly$Year<=2008, ]
test_set <- Weekly[Weekly$Year>2008, ]
logistic_fit <- glm(Direction~Lag2, data=training_set, family=binomial)
prediction_logistic <- sapply(predict(logistic_fit, newdata=test_set, type="response"), function(x) ifel
table(prediction_logistic, test_set$Direction)
```

```
>>>
>>> prediction_logistic Down Up
>>>                Down    9  5
>>>                Up     34 56
```

(g), (h) Repeat (d) using KNN with K = 1.

First load the `class` library which is necessary for `knn()` method.

```
library(class)
```

To use `knn()` method, the training data and test data are converted to matrix. Yet unsimilar to normal logistic regression, `knn()` does fitting and prediction in the same function. The returned result, moreover, is recorded as factors which eliminates the need of conversion.

The prediciton is printed in a confusion matrix as in (d). In this case, the prediction is less biased toward *Up*, but comparatively the accuracy is lower $\frac{21+32}{21+32+22+29} \approx 51.0\%$ than the vanilla logistic regression.

In comparison, the vanilla logistic regression returns higher prediction accuracy than the knn method used.

```
prediction_knn <- knn(train=matrix(training_set$Lag2), test=matrix(test_set$Lag2), cl=matrix(training_s
table(prediction_knn, test_set$Direction)
```

```
>>>
>>> prediction_knn Down Up
>>>           Down   21 30
>>>           Up     22 31
```

(i) Experiment with different combinations of predictors, including possible transformations and interactions, for each of the methods. Report the variables, method, and associated

confusion matrix that appears to provide the best results on the held out data. Note that you should also experiment with values for K in the KNN classifier.

Let's first dig a little bit deeper into the KNN method, by varying `k`- number of neighbours used in estimating new data.

In the previous problem, only `k=1` was used, here `k=2:10` is used to explore how the accuracy changes with `k`.

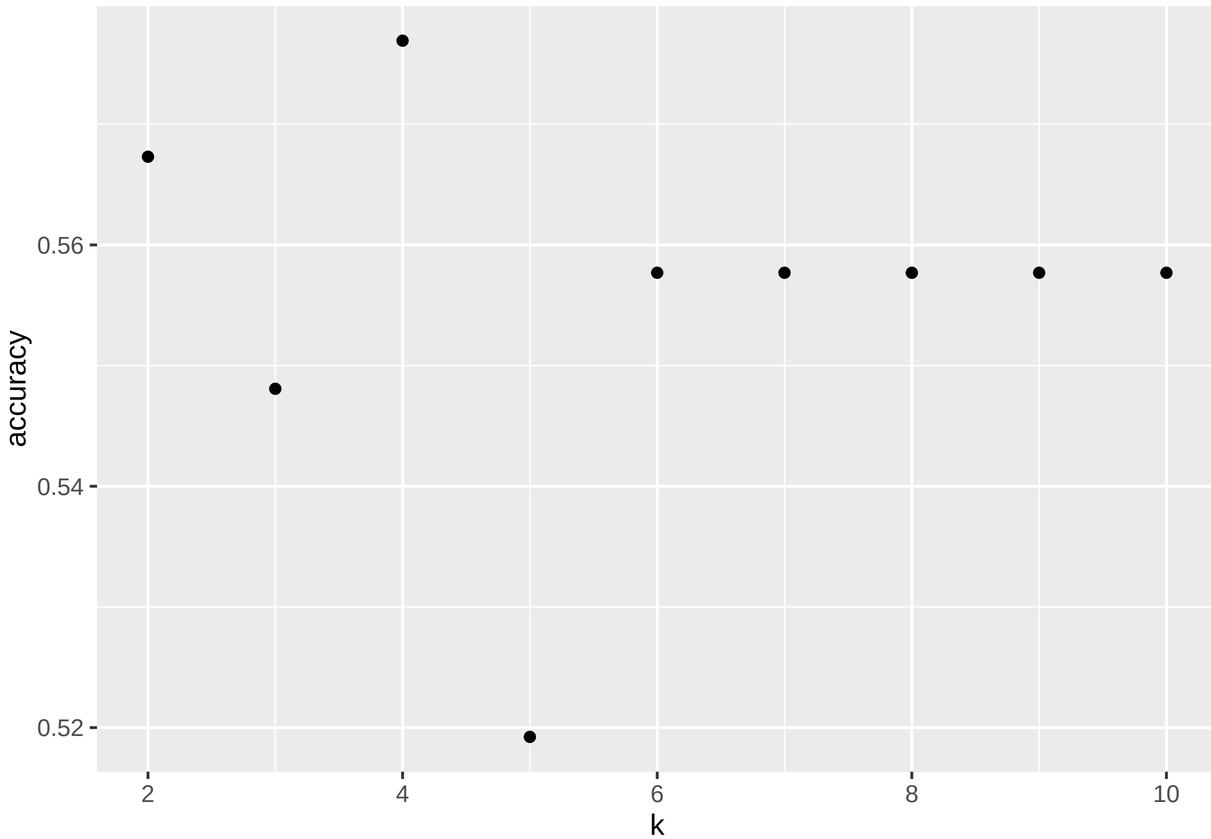
```
knn_k <- function(nk){
    prediction_tmp <- knn(train=matrix(training_set$Lag2), test=matrix(test_set$Lag2), cl=matrix(trainin
    t_tmp <- table(prediction_tmp, test_set$Direction)
    accuracy <- sum(t_tmp[1, 1], t_tmp[2, 2])/sum(t_tmp)
    return(accuracy)
}

k_accuracy <- sapply(2:10, knn_k)

k_df <- data.frame(k=2:10, accuracy=k_accuracy)

p2 <- qplot(x=k, y=accuracy, data=k_df, geom="auto")

p2
```




```
cat("When k=", which.max(k_accuracy)+1, "the prediction is the most accurate.")
cat("Max accuracy is:", k_accuracy[which.max(k_accuracy)]*100, "%")
```

```
>>> When k= 4 the prediction is the most accurate.Max accuracy is: 57.69231 %
```

Then let's trying improving the prediction using logistic regression. The baseline was established on

`Direction~Lag2` because when all `Lag` variables are included together with `Volume`, only `Lag2` was statistically significant. To improve the model's capability, now let's try adding more features into the formula, and creating interactions between some of them.

A function is defined to help choose the right combination of predictors, which is expressed below. It takes a list of predictors and add the predictors to the baseline formula and train a new model. Then it prints out the confusion matrix of the prediction and returns the predicion accuracy.

An example is given below. By calling `glm_n` function with `Lag3`, `Lag3` is added into the baseline model, and the confusion matrix of the new model `Direction~Lag2+Lag3` is returned.

```
glm_n <- function(names){
    formula <- as.formula(paste("Direction~Lag2", sapply(names, function(s) paste("+", s))))
    logistic_fit <- glm(formula, data=training_set, family=binomial)
    prediction_logistic <- sapply(predict(logistic_fit, newdata=test_set, type="response"), function(x)
    tabl <- table(prediction_logistic, test_set$Direction)
    glm_accuracy <- (tabl[1, 1]+tabl[2, 2])/sum(tabl)
    print(tabl)
    return(glm_accuracy)
}

glm_n("Lag3")
```

```
>>>
>>> prediction_logistic Down Up
>>>                Down    8  4
>>>                Up     35 57
>>> [1] 0.625
```

After trying out different combinations, it is found that with `Direction~Lag2+Lag3*Lag4` the model gives best prediction results, as seen below.

```
glm_n("Lag3*Lag4")
```

```
>>>
>>> prediction_logistic Down Up
>>>                Down   10  5
>>>                Up     33 56
>>> [1] 0.6346154
```