ENVX2001 Practical Topic 10 Prediction and model quality

Willem Vervoort

Document generated on 2017-05-23

Objectives

- Making predictions using a developed model and understand confidence;
- Understand the difference between adjusted r^2 and r^2 to describe model quality;
- Test model quality using validation and calculation of quality measures.

DATA: *Data_Topic10_2017.xls*

EXERCISE 1 MAKING PREDICTIONS ON CALIFORNIAN STREAMFLOW

Data: *California streamflow worksheet*, *2017_Californiastreamflow.csv*

Use the best model with 2 variables identified last week:

```
# read in the data
s.data <- read.csv("2017_Californiastreamflow.csv")
names(s.data)
```

```
## [1] "L10APSAB" "L10OBPC"  "L10OPRC"  "L10BSAAM"
```

```
# best model
ML_Mod2 <- lm(L10BSAAM ~ L10OPRC + L10OBPC, data = s.data)
summary(ML_Mod2)
```

```
##
## Call:
## lm(formula = L10BSAAM ~ L10OPRC + L10OBPC, data = s.data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.09832 -0.02350  0.01076  0.03291  0.08568
##
```

```
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.35762    0.10547  31.835  < 2e-16 ***
## L10OPRC      0.44437    0.08925   4.979 1.26e-05 ***
## L10OBPC      0.21051    0.06861   3.068  0.00385 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.04937 on 40 degrees of freedom
## Multiple R-squared:  0.8749, Adjusted R-squared:  0.8686
## F-statistic: 139.8 on 2 and 40 DF,  p-value: < 2.2e-16
```

If we wish to predict y for a specific pair of values of x1 and x2, we can simply substitute these into the fitted model:

$\hat{y} = 3.35762 + 0.44437 * L10OPRC + 0.21051 * L10OBPC$

For example, if L10OPRC $= 2$ and L10OBPC $= 3$, then L10BSAAM $= \hat{y} = 4.87789$.

It is also convention to give a standard error (SE) for any prediction. The formula for the SE of a prediction from a 2 predictor linear regression model is complex (see page 215 of Med et al, 2003). However, in R it is simple to also return a corresponding SE value using `predict()` and specifying `se.fit=T`. In this case the output will then include an element called "se.fit". The tricky bit with `predict()` in R is that you need to specify `newdata` (see the help file), which has to be exactly the same structure as the original data. So to repeat the above example in R:

```
# create a new data frame with the variables to predict at
# Note that it does not matter what you put in for L10BSAAM
new.df <- data.frame(L10BSAAM = 0, L10OPRC = 2, L10OBPC = 3)
# now use predict() and specify se.fit=T
predLake <- predict(ML_Mod2, newdata = new.df, se.fit = T)
# the output now has two elements:
# the fit
predLake$fit
```

```
##        1
## 4.877918
```

```
# the se of the fit
predLake$se.fit
```

```
## [1] 0.07583833
```

This prediction is well within the original data set.
a. Why? You can use `range()` to figure this out, or just look at the original data.

More interesting is making prediction not part of the original data, but as we discussed in the lectures this means there is a different confidence interval. R allows you to define the interval using `interval = "prediction"`. The output will then include both the fitted and the prediction confidence interval and the default is to calculate the 95% confidence interval. If you want to calculate the actual se.fit from the output, you need to subtract the actual prediction and divide by the $t_{0.05}$ for `df = 40` (which is n - p - 1).

```
# create a new data frame with the variables to predict at
# Note that it again does not matter what you put in for L10BSAAM,
# I have put in five 0 values
new.df <- data.frame(L10BSAAM=rep(0,5),
                     L10OPRC = seq(3.0,4.0, length=5),
                     L10OBPC=seq(3.0,4.0, length=5))
```

b. Now use `predict()` and specify 'interval="prediction".

c. Inspect the output, the lwr and upr columns are the upper and lower confidence intervals. Note that the variation and prediction intervals are fairly small.

d. To calculate the true se.fit outside the confidence intervals, subtract column 1 from column 3 and divide by 2-tailed t for the interval at df 40 (n - p - 1) (which is 2.02).

EXERCISE 2 - MEASURES OF MODEL QUALITY r^2 VERSUS ADJUSTED r^2

Data: *California Streamflow* worksheet, *2017_Californiastreamflow.csv*

The dataset is again the California Streamflow data used above. Import the data in R and generate a normally distributed random variable with a mean of 3 and variance of 2 using the following bit of Rcode.

```
set.seed(100) # to make sure everybody gets the same results
s.data$random_no <- rnorm(nrow(s.data),3,2)
# this generates the random number into the dataset
```

We will see the impact of including a totally useless variable, such as this random variable, has on measures of model quality, r^2 and adjusted r^2 values.

Task
Create two regression models:

1. model L10BSAAM with L10OPRC + L10OPBC
2. model L10BSAAM with L10OPRC + L10OPBC + random_no
   a. Compare each in terms of their r^2 and adjusted r^2 values. Which performance measure (r^2 or adj r^2) would you use to identify which predictors to use in your model?

EXERCISE 3 - VALIDATION AND CHECKING MODEL PREDICTION QUALITY

Data: *California Streamflow* worksheet, *2017_Californiastreamflow.csv*

In this exercise we will test the quality of the developed model, but doing it formally using a comparison on a validation data set. We have to once again `set.seed()` to make sure your results are the same across the class. The first step is to sample 25% of the data as a validation data set from the overall data set. We are doing this by using the function `sample()` to pick a random number of rows. I am using `dim()` to check the dimensions of the data sets.

```
#Only use so we are all get same random numbers -
# otherwise R uses computer time to get random number
set.seed(10)

#Sample 20% of the rows, find row numbers
indexes <- sample(1:nrow(s.data), size = 0.20*nrow(s.data))
#Split data
valid <- s.data[indexes,]
dim(valid)
```

```
## [1] 8 5
```

```
calib <- s.data[-indexes,]
dim(calib)
```

```
## [1] 35  5
```

Rather than rerunning the calibration we are going to reuse the two models from last week with different data and compare the results. We will use the model with 2 variables and the model with 3 variables.

```
# use model 2 and model 3 from topic 9 practical (last week)
# and test which one is the best model, but use calib data
```

```r
ML_Mod2 <- lm(L10BSAAM ~ L10OPRC + L10OBPC, data = calib)
ML_Mod3 <- lm(L10BSAAM ~ L10OPRC + L10OBPC + L10APSAB, data = calib)

# compare the models
summary(ML_Mod2)
```

```
##
## Call:
## lm(formula = L10BSAAM ~ L10OPRC + L10OBPC, data = calib)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.097141 -0.023042  0.003415  0.035688  0.082055
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.30629    0.11686  28.292  < 2e-16 ***
## L10OPRC      0.50956    0.09906   5.144 1.31e-05 ***
## L10OBPC      0.16684    0.07861   2.122   0.0416 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.04941 on 32 degrees of freedom
## Multiple R-squared:  0.8736, Adjusted R-squared:  0.8657
## F-statistic: 110.5 on 2 and 32 DF,  p-value: 4.265e-15
```

```r
summary(ML_Mod3)
```

```
##
## Call:
## lm(formula = L10BSAAM ~ L10OPRC + L10OBPC + L10APSAB, data = calib)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.090336 -0.034301  0.006501  0.030779  0.093848
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.17307    0.14163  22.404  < 2e-16 ***
```

```
## L10OPRC      0.51066    0.09678   5.277 9.72e-06 ***
## L10OBPC      0.16612    0.07680   2.163   0.0384 *
## L10APSAB     0.06527    0.04106   1.590   0.1221
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.04828 on 31 degrees of freedom
## Multiple R-squared:  0.8831, Adjusted R-squared:  0.8718
## F-statistic: 78.06 on 3 and 31 DF,  p-value: 1.528e-14
```

Based on the results, ML_Mod3 is better based on a tiny bit better adj r^2. So we should do a more thorough investigation which model is better

You might want to check the residual plots of the predictions. Do you observe anything suspicious?

```
par(mfrow=c(2,2))
plot(ML_Mod2,which=c(1,2,5))
hist(rstandard(ML_Mod2))
```

```
par(mfrow=c(1,1))
```

```
par(mfrow=c(2,2))
plot(ML_Mod3,which=c(1,2,5))
hist(rstandard(ML_Mod3))
```

```
par(mfrow=c(1,1))
```

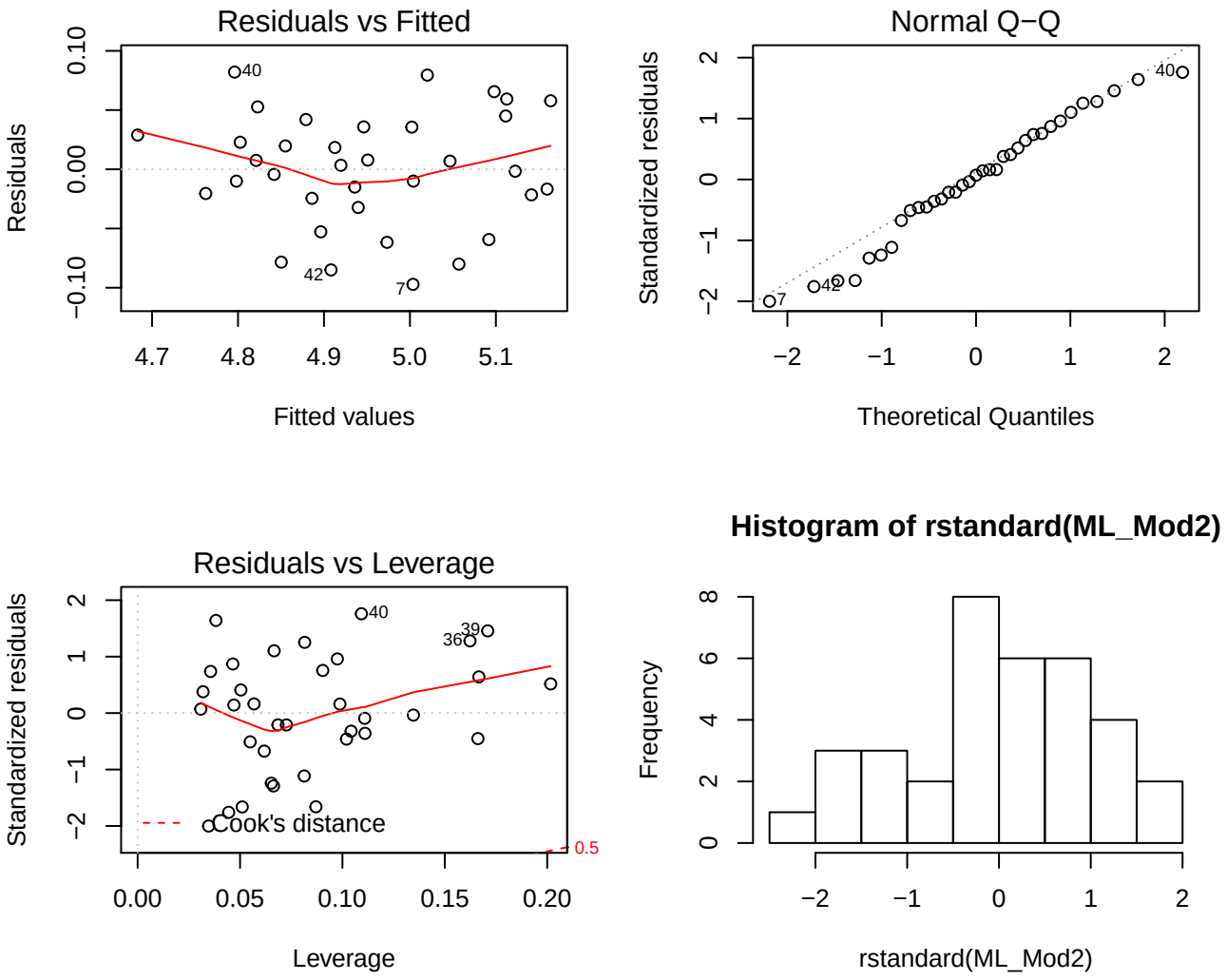


Figure 1: Residual plots for Model with 2 parameters

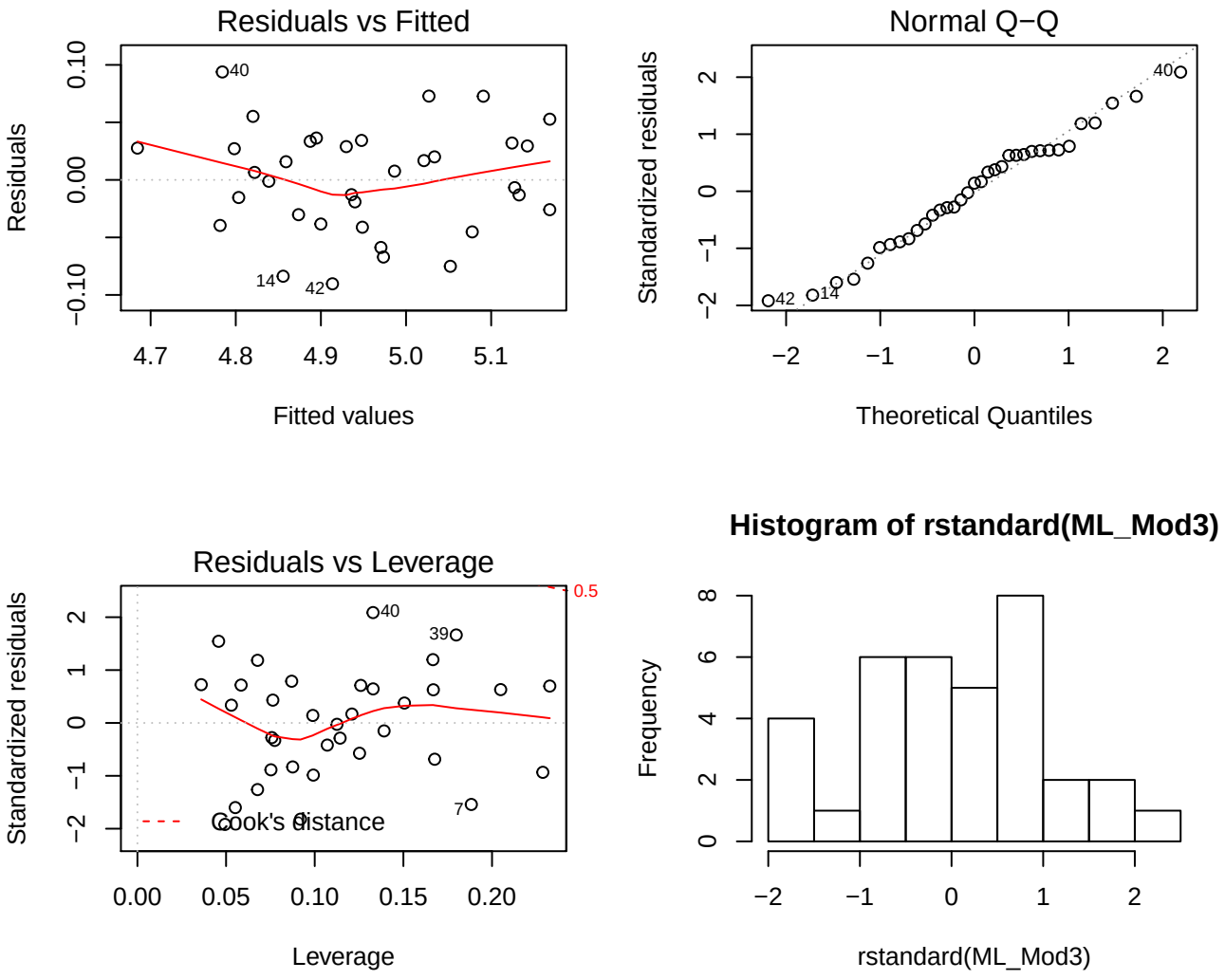


Figure 2: Residual plots for Model with 3 parameters

a. Accuracy: check RMSE and bias of the calibrated models. Use the equation for RMSE: $RMSE = \sqrt{(\frac{1}{n}(\sum(y - \hat{y})^2)}$ and for bias $Bias = \frac{1}{n}(\sum(y - \hat{y})$. Check both the models and for both calibration and validation. To derive the validation data, use (for example for Model 2):

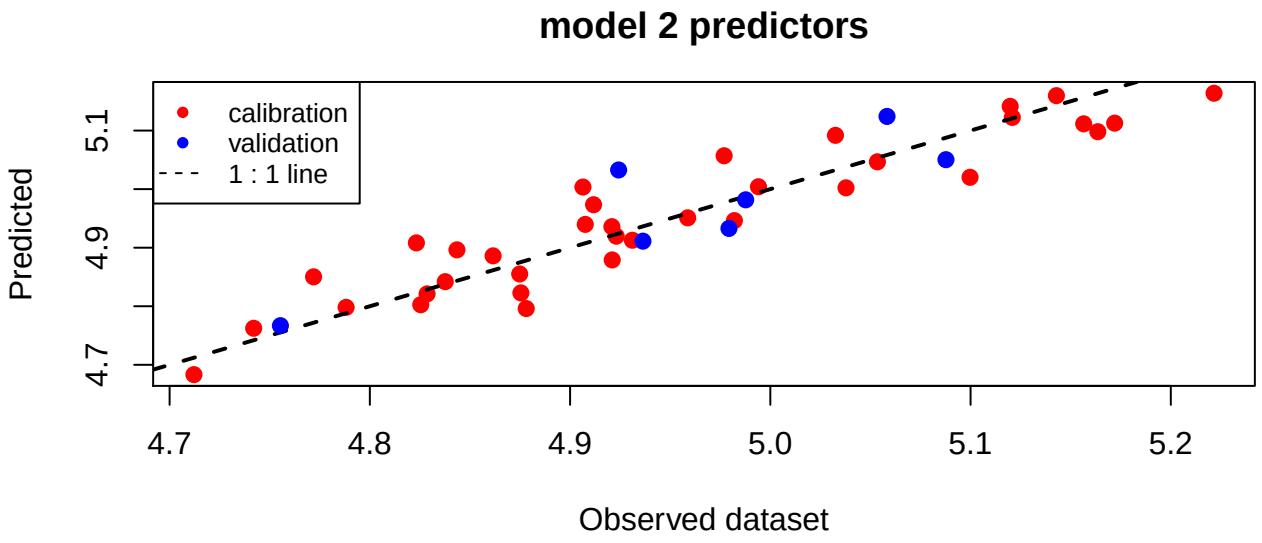
```
predict(ML_Mod2, newdata = valid)
```

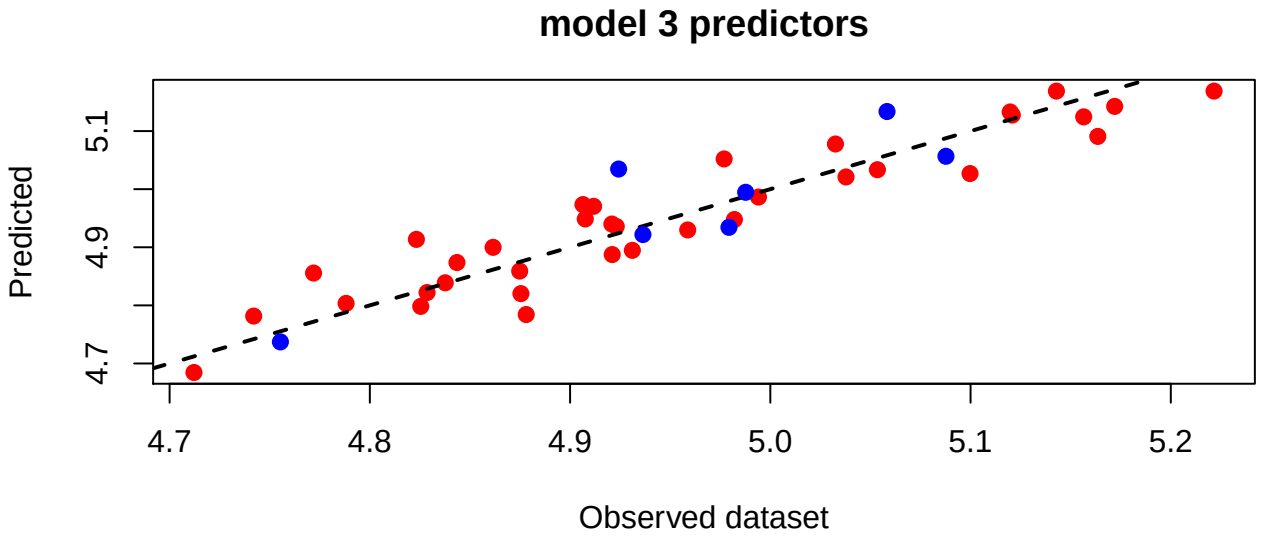
```
##       22       13       18       28        4        9       11       10
## 5.240699 4.766814 4.981750 4.932881 5.032698 5.124130 5.050281 4.911155
```

We can subsequently make a plot of the calibration and validation data sets for both observed data and the predicted data and compare to the 1:1 line.

```
# plot predicted versus observed
par(mfrow = c(2,1), mar=c(4,4,3,2))
# model 2
plot(calib$L10BSAAM, predict(ML_Mod2),
     # colour = red, type = "16" and size is 20% larger
     pch = 16, col = "red", cex = 1.2,
     # add titles for axes and main
     xlab = "Observed dataset", ylab = "Predicted",
     main="model 2 predictors")
# insert a 1:1 line, dashed line, width = 2
abline(0, 1, lty = 2, lwd = 2)
# add the validation data
points(valid$L10BSAAM, predict(ML_Mod2, newdata = valid),
       # colour = blue, type = "16" and size is 20% larger
       col = "blue", pch = 16, cex = 1.2)
# add a legend to the first plot
legend("topleft", c("calibration", "validation", "1 : 1 line"),
       pch = c(16, 16, NA), lty = c(NA, NA, 2), col = c("red", "blue", 1),
       # 20% smaller
       cex=0.8)
# model 3
plot(calib$L10BSAAM, predict(ML_Mod3),
     # colour = red, type = "16" and size is 20% larger
     pch = 16, col = "red", cex = 1.2,
     # add titles for axes and main
     xlab = "Observed dataset", ylab = "Predicted",
     main="model 3 predictors")
```

```
# insert a 1:1 line, dashed line, width = 2
abline(0, 1, lty = 2, lwd = 2)
# add the validation data
points(valid$L10BSAAM, predict(ML_Mod3, newdata = valid),
       # colour = blue, type = "16" and size is 20% larger
       col = "blue", pch = 16, cex = 1.2)
```





This gives the opportunity for a visual inspection, but we can also calculate the correlation (to echo the model derivation) for the calibration and validation data sets

b. Calculate correlation using `cor()`

c. Now calculate Lin's coefficient of concordance, using the lecture slides, don't forget to call `library(epiR)` and maybe `install.packages("epiR")` if the package is not installed.

d. Draw conclusions about which model is the best model to predict L10BSAAM from the other variables, use the results to support your argument.

END OF PRACTICAL