Linear_Regression2

author: Dr. Marko Mitic

Problem Description: **DETECTING FLU EPIDEMICS VIA SEARCH ENGINE QUERY DATA**

Flu epidemics constitute a major public health concern causing respiratory illnesses, hospitalizations, and deaths. According to the National Vital Statistics Reports published in October 2012, influenza ranked as the eighth leading cause of death in 2011 in the United States. Each year, 250,000 to 500,000 deaths are attributed to influenza related diseases throughout the world.

The U.S. Centers for Disease Control and Prevention (CDC) and the European Influenza Surveillance Scheme (EISS) detect influenza activity through virologic and clinical data, including Influenza-like Illness (ILI) physician visits. Reporting national and regional data, however, are published with a 1-2 week lag.

The Google Flu Trends project, https://www.google.org/flutrends/us/#US, was initiated to see if faster reporting can be made possible by considering flu-related online search queries – data that is available almost immediately.

We would like to estimate influenza-like illness (ILI) activity using Google web search logs. Fortunately, one can easily access this data online:

ILI Data - The CDC publishes on its website the official regional and state-level percentage of patient visits to healthcare providers for ILI purposes on a weekly basis.

Google Search Queries - Google Trends allows public retrieval of weekly counts for every query searched by users around the world. For each location, the counts are normalized by dividing the count for each query in a particular week by the total number of online search queries submitted in that location during the week. Then, the values are adjusted to be between 0 and 1.

The csv file FluTrain.csv aggregates this data from January 1, 2004 until December 31, 2011 as follows:

-**Week** - The range of dates represented by this observation, in year/month/day format.

-**ILI** - This column lists the percentage of ILI-related physician visits for the corresponding week.

-**Queries** - This column lists the fraction of queries that are ILI-related for the corresponding week, adjusted to be between 0 and 1 (higher values correspond to more ILI-related search queries).

Structure of the data can be observed using `str`:

```r
FluTrain=read.csv("FluTrain.csv")
str(FluTrain)
```

```
## 'data.frame':    417 obs. of  3 variables:
##  $ Week   : Factor w/ 417 levels "2004-01-04 - 2004-01-10",..: 1 2 3 4 5 6 7 8 9 10 ...
##  $ ILI    : num  2.42 1.81 1.71 1.54 1.44 ...
##  $ Queries: num  0.238 0.22 0.226 0.238 0.224 ...
```

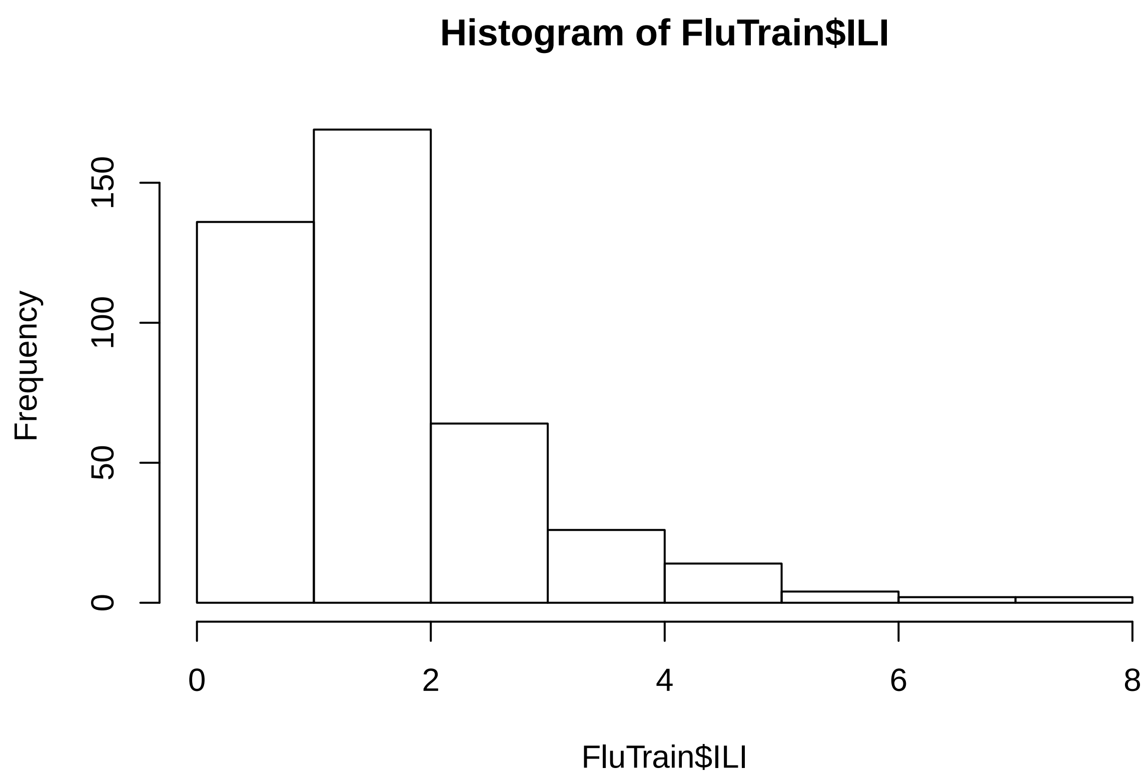
Summary statistics is available with `summary` command:

```r
summary(FluTrain)
```

```
##                       Week           ILI              Queries
##  2004-01-04 - 2004-01-10:  1   Min.   :0.5341   Min.   :0.04117
##  2004-01-11 - 2004-01-17:  1   1st Qu.:0.9025   1st Qu.:0.15671
##  2004-01-18 - 2004-01-24:  1   Median :1.2526   Median :0.28154
##  2004-01-25 - 2004-01-31:  1   Mean   :1.6769   Mean   :0.28603
##  2004-02-01 - 2004-02-07:  1   3rd Qu.:2.0587   3rd Qu.:0.37849
##  2004-02-08 - 2004-02-14:  1   Max.   :7.6189   Max.   :1.00000
##  (Other)                :411
```

Let us plot a histogram on `ILI` variable. We can see that the values are most of the values are small, with a relatively small number of much larger values:

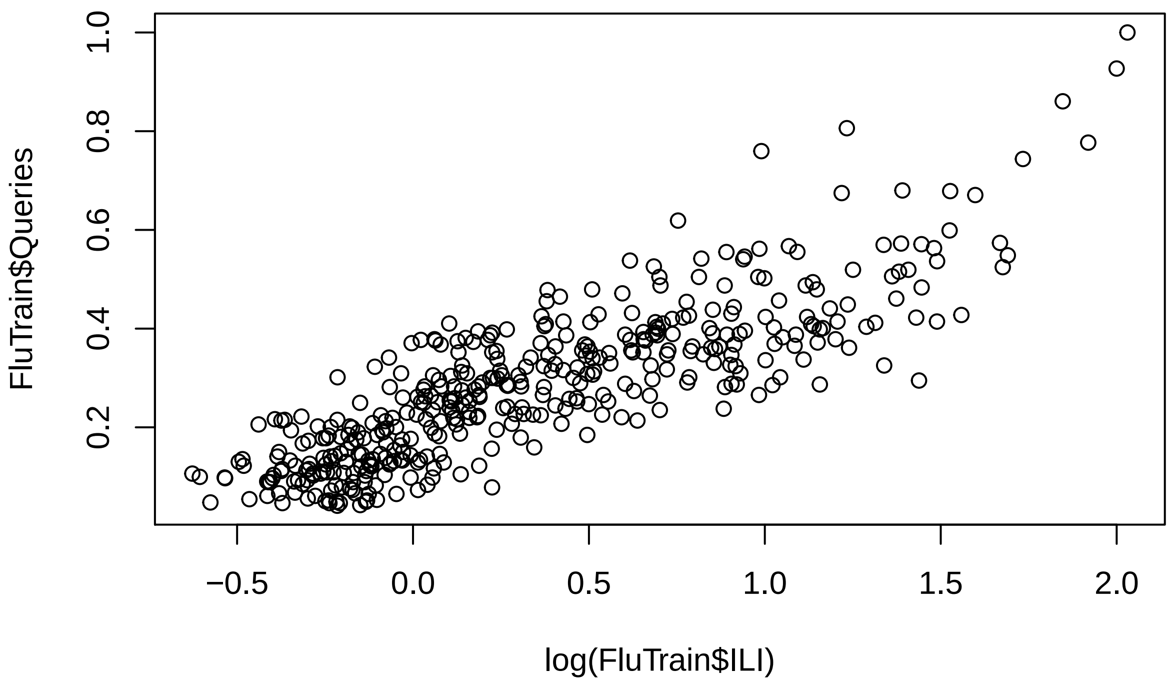
```r
hist(FluTrain$ILI)
```



The values are "skewed right", so we must modify the `ILI` variable. In this situation, it is useful to predict the logarithm of the dependent variable instead of the dependent variable itself – this prevents the small number of unusually large or small observations from having an undue influence on the sum of squared errors of predictive models. In this problem, we will predict the natural log of the ILI variable, which can be computed in R using the log() function.

Plotting the `log(ILI)` vs `Queries`, the linear relationship between these variable becomes evident.

```r
plot(log(FluTrain$ILI), FluTrain$Queries)
```



Next, we can build linear model as follows:

```r
FluTrend1 = lm(log(ILI)~Queries, data=FluTrain)
summary(FluTrend1)
```

```
##
## Call:
## lm(formula = log(ILI) ~ Queries, data = FluTrain)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.76003 -0.19696 -0.01657  0.18685  1.06450
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.49934    0.03041  -16.42   <2e-16 ***
## Queries      2.96129    0.09312   31.80   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2995 on 415 degrees of freedom
## Multiple R-squared:  0.709,  Adjusted R-squared:  0.7083
## F-statistic:  1011 on 1 and 415 DF,  p-value: < 2.2e-16
```

The Multiple R-squared value is 0.709 which is very solid result. To validate the model, we can use test dataset `FluTest`. However, we should bare in mind that log(ILI) instead of ILI was set as dependent variable, so our predict function will look as follows:

```
FluTest = read.csv("FluTest.csv")
PredTest1 = exp(predict(FluTrend1, newdata=FluTest))
```

Relative error between predictive and actual values are calculated as:

```
Rel = (FluTest$ILI-PredTest1)/FluTest$ILI
```

RMSE on the test set can be determined next:

```
SSE = sum((PredTest1-FluTest$ILI)^2)
RMSE=sqrt(SSE/nrow(FluTest))
RMSE
```

```
## [1] 0.7490645
```

Often, statistical models in **time series** datasets like this can be improved by predicting the current value of the dependent variable using the value of the dependent variable from earlier weeks. In our models, this means we will predict the ILI variable in the current week using values of the ILI variable from previous weeks.

First, we need to decide the amount of time to lag the observations. Because the ILI variable is reported with a 1- or 2-week lag, a decision maker cannot rely on the previous week's ILI value to predict the current week's value. Instead, the decision maker will only have data available from 2 or more weeks ago. We will build a variable called ILILag2 that contains the ILI value from 2 weeks before the current observation.

To do so, we will use the "zoo" package, which provides a number of helpful methods for time series models. While many functions are built into R, you need to add new packages to use some functions. New packages can be installed and loaded easily in R, and we will do this many times in this class.

```
#install.packages("zoo")
library(zoo)
```

```
## Warning: package 'zoo' was built under R version 3.2.1
```

```
##
## Attaching package: 'zoo'
##
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

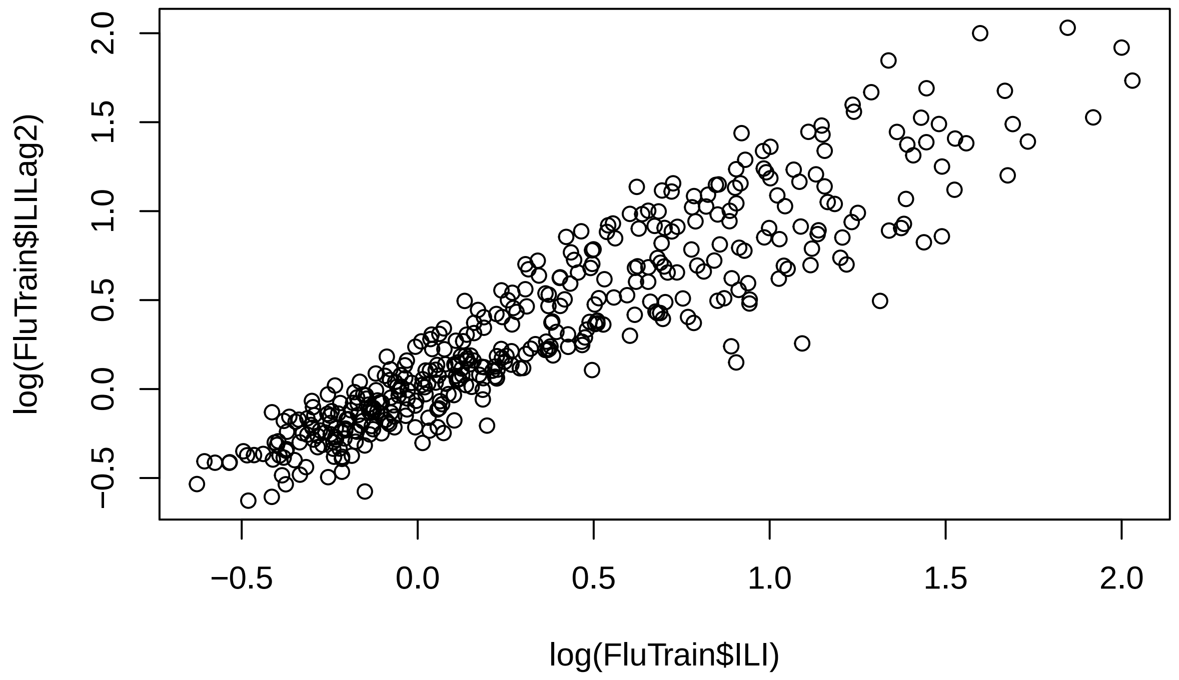
After this, new variable can be generated as follows:

```
ILILag2 = lag(zoo(FluTrain$ILI), -2, na.pad=TRUE)
FluTrain$ILILag2 = coredata(ILILag2)
summary(FluTrain$ILILag2)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##  0.5341  0.9010  1.2520  1.6750  2.0580  7.6190       2
```

We can observe the strong linear relationship between this new and old ILI variable:

```
plot(log(FluTrain$ILI),log(FluTrain$ILILag2))
```



Corelation between ILI variables can also be easily determined:

```
cor(log(FluTrain$ILI),log(FluTrain$ILILag2), use="complete")
```

```
## [1] 0.9214355
```

Next, the new linear regression FluTrend2 model can be built:

```
FluTrend2 = lm(log(ILI) ~ Queries + log(ILILag2), data=FluTrain)
summary(FluTrend2)
```

```
##
## Call:
## lm(formula = log(ILI) ~ Queries + log(ILILag2), data = FluTrain)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.52209 -0.11082 -0.01819  0.08143  0.76785
##
## Coefficients:
```

```
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.24064    0.01953  -12.32   <2e-16 ***
## Queries       1.25578    0.07910   15.88   <2e-16 ***
## log(ILILag2)  0.65569    0.02251   29.14   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1703 on 412 degrees of freedom
##   (2 observations deleted due to missingness)
## Multiple R-squared:  0.9063, Adjusted R-squared:  0.9059
## F-statistic:  1993 on 2 and 412 DF,  p-value: < 2.2e-16
```

Since all the variables are significant, there is no evidence of overfitting, and FluTrend2 is superior to FluTrend1 on the training set.

According to this, `ILILag2` variable should be added to test set also:

```r
ILILag2 = lag(zoo(FluTest$ILI), -2, na.pad=TRUE)
FluTest$ILILag2 = coredata(ILILag2)
summary(FluTest$ILILag2)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##  0.9018  1.1360  1.3410  1.5190  1.7610  3.6000       2
```

The training and testing sets are split sequentially – the training set contains all observations from 2004-2011 and the testing set contains all observations from 2012. Therefore, missing test values can be take from training dateset: the appropriate value 2 weeks before the missing value should be taken.

```r
FluTest$ILILag2[1]=FluTrain$ILI[416]
FluTest$ILILag2[2]=FluTrain$ILI[417]
```

Finally, with new RMSE test on FluTrend2 we can validate our conclusion abot both developed linear models:

```r
PredTest2 = exp(predict(FluTrend2, newdata = FluTest))
RMSE2=sqrt(sum((FluTest$ILI-PredTest2)^2)/nrow(FluTest))
RMSE2
```

```
## [1] 0.2942029
```