Forest Fires Dataset Regression Analysis

Filippo Boni

Contents

1 Introduction **1**

1.1 Development Environment . . . 2

2 Dataset Overview **2**

2.1 Variables . . . 2

2.2 Exploratory Data Analysis . . . 4

2.3 Correlations . . . 5

2.4 Processing . . . 6

3 Regression Models **8**

3.1 General Linear Model . . . 8

3.1.1 OLS Regression . . . 8

3.1.2 Simple model . . . 9

3.1.3 Complete Model . . . 10

3.1.4 Smaller models . . . 12

3.1.5 Model Selection . . . 12

3.1.6 Validation of the model . . . 15

3.2 Generalized Linear Model . . . 16

3.2.1 Basic theory . . . 16

3.2.2 Logistic Regression Models . . . 16

4 Conclusions **21**

References **22**

1 Introduction

In this work we analyze a dataset of forest fires using the statistical approach of *regression*, focusing in the first part in predicting the scale of a woodland fireplace exploiting *general linear models* while in the second part in predicting the class of the scale of the fire exploiting *generalized linear models*. Nowadays forest fires are dealt by collecting the satellite images of forest and if there is an any emergency caused by the fires the authorities are notified to mitigate its effects. Unfortunately, by the time the fire is taken down, the environment is already damaged. The presented research offers a new preventive approach to the forest fires emergency, where geo-spatial and weather condition data can be used to predict the eventuality of a forest fire.

First we present the dataset and the attribute description, we study their distributions and correlations. After some processing, we dive into the linear regression analysis, where we build different models based on different subsets of predictors. Finally, the problem is modeled under a classification point of view to tackle the high skewness of the target variable and the same steps applied in the linear regression part are applied here.

1.1 Development Environment

All the project is developed with R programming language in a R Markdown notebook. The main libraries used are the following

```r
#imports
library(gridExtra) #combine plots in a grid
library(ggplot2) #visualization of results and data
library(tidyverse) #collection of R packages for data science
library(GGally) #extension of ggplot2
library(caret) #cross-validation
library(car) #VIF
library(caTools) #partition the dataset
library(MASS) #base package for statistics in R
library(regclass) #confusion matrix
library(bookdown)
options(max.print = 100) #limit some printing
```

2 Dataset Overview

The data we are using to predict the creation and the intensity of forest fires has been created by Cortez and Morais (2007) and acquired from the UCI Machine Learning Repo. It consists of 517 observations of forest fires and wildfires from Montesinho Park, located in Portugal, collected from January 2000 to December 2003. The dataset is composed by 13 variables including the output variable *area* (i.e. the area of the forest fire in hectares) and 12 explanatory variables consisting of spatial and temporal variables, FWI component variables and weather variables.

2.1 Variables

UCI Repository provides some useful attributes information.

Attribute	Description
X	x-axis spatial coordinate within the Montesinho park map: 1 to 9
Y	y-axis spatial coordinate within the Montesinho park map: 2 to 9
month	month of the year: ‘jan’ to ‘dec’
day	day of the week: ‘mon’ to ‘sun’
FFMC	FFMC index from the FWI system: 18.7 to 96.20
DMC	DMC index from the FWI system: 1.1 to 291.3
DC	DC index from the FWI system: 7.9 to 860.6

Attribute	Description
ISI	ISI index from the FWI system: 0.0 to 56.10
temp	temperature in Celsius degrees: 2.2 to 33.30
RH	relative humidity in %: 15.0 to 100
wind	wind speed in km/h: 0.40 to 9.40
rain	outside rain in mm/m2 : 0.0 to 6.4
area	the burned area of the forest (in ha): 0.00 to 1090.84

The following picture helps understanding how the four indexes belonging to Canadian FWI (Fire Weather System) are generated:

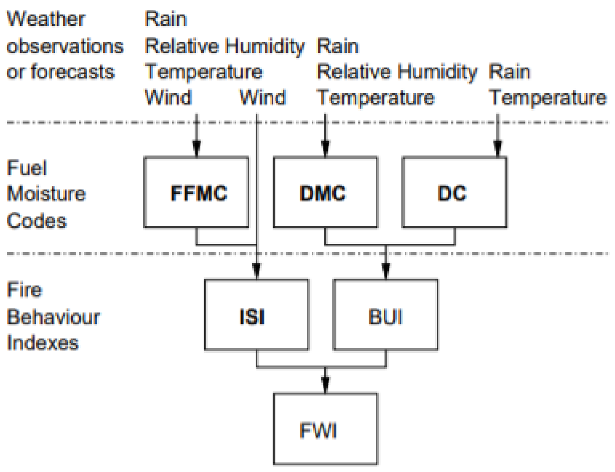


Figure 1: FWI indexes

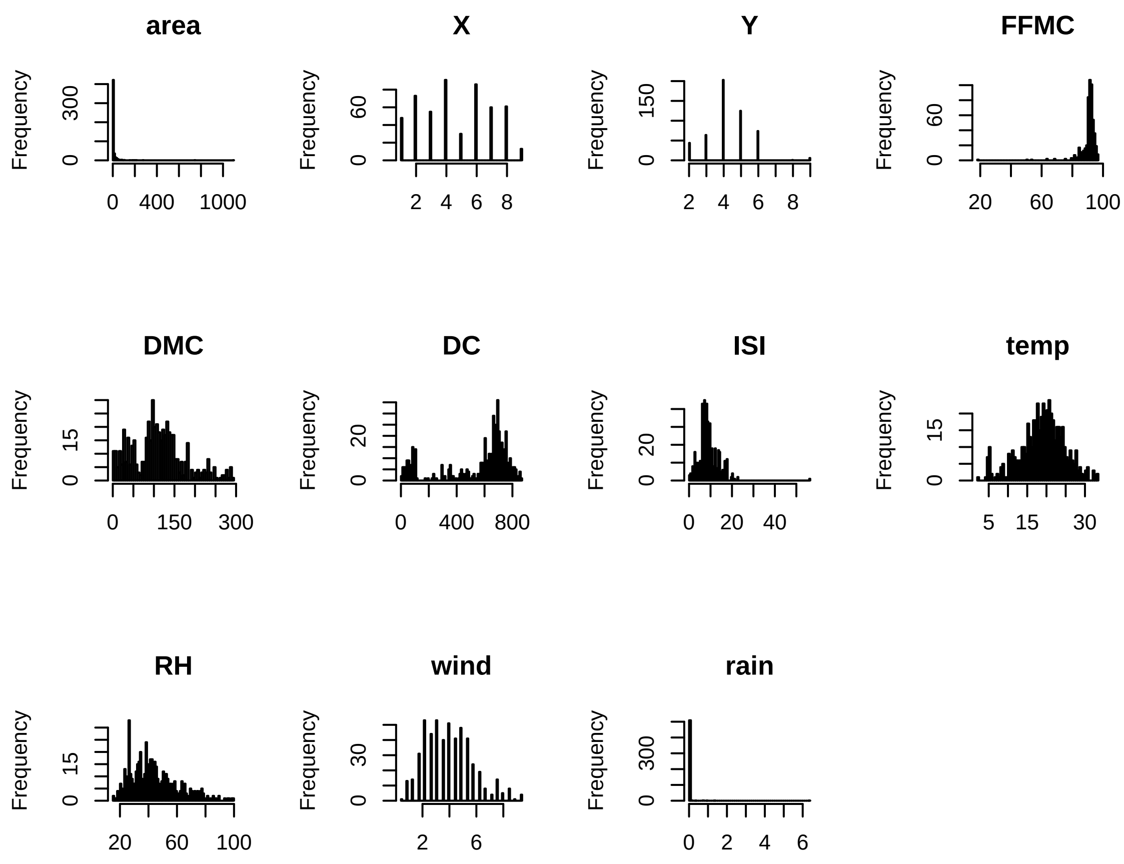
```r
#read data
fires <- read.csv(file = "forestfires.csv",header = TRUE)

#set the levels of month and day
fires$month <- factor(fires$month,levels =  c("jan", "feb", "mar", "apr",
                       "may", "jun", "jul", "aug", "sep", "oct", "nov", "dec"))
fires$day <- factor(fires$day,levels =  c("mon", "tue","wed", "thu",
                                          "fri", "sat", "sun"))

attach(fires)
head(fires)
```

```
##   X Y month day FFMC  DMC    DC  ISI temp RH wind rain area
## 1 7 5   mar fri 86.2 26.2  94.3  5.1  8.2 51  6.7  0.0    0
## 2 7 4   oct tue 90.6 35.4 669.1  6.7 18.0 33  0.9  0.0    0
## 3 7 4   oct sat 90.6 43.7 686.9  6.7 14.6 33  1.3  0.0    0
## 4 8 6   mar fri 91.7 33.3  77.5  9.0  8.3 97  4.0  0.2    0
## 5 8 6   mar sun 89.3 51.3 102.2  9.6 11.4 99  1.8  0.0    0
## 6 8 6   aug sun 92.3 85.3 488.0 14.7 22.2 29  5.4  0.0    0
```

2.2 Exploratory Data Analysis

Let's first visualize the frequencies of the quantitative variables by plotting the histograms with the number of occurrences on the y axis and the relative variable on the x axis

```
par(mfrow=c(3,4))
hist(fires$area,100,main = "area",xlab = "")
hist(fires$X,100,main = "X",xlab = "")
hist(fires$Y,100,main = "Y",xlab = "")
hist(fires$FFMC,100,main = "FFMC",xlab = "")
hist(fires$DMC,100,main = "DMC",xlab = "")
hist(fires$DC,100,main = "DC",xlab = "")
hist(fires$ISI,100,main = "ISI",xlab = "")
hist(fires$temp,100,main = "temp",xlab = "")
hist(fires$RH,100,main = "RH",xlab = "")
hist(fires$wind,100,main = "wind",xlab = "")
hist(fires$rain,70,main = "rain",xlab = "")
```



Let's visualize better the number of zero values of the *area* variable (representing a burned area smaller than $100m^2$) and *rain* variable with respect to the number of values greater that zero.

```r
big_fire <- rep(0,length(fires$X))
big_fire[fires$area>0] <-"Greater that zero"
rain_yes <- rep(0,length(fires$X))
rain_yes[fires$rain>0]<-"Greater than zero"

area_count <- ggplot(fires,aes(x=big_fire))+
  geom_bar(fill = "steelblue")+
  xlab("area") + theme_minimal(base_size = 10)
rain_count <-ggplot(fires,aes(x=rain_yes))+
  geom_bar(fill = "steelblue")+
  xlab("rain") + theme_minimal(base_size = 10)

grid.arrange(area_count,rain_count,ncol=2)
```

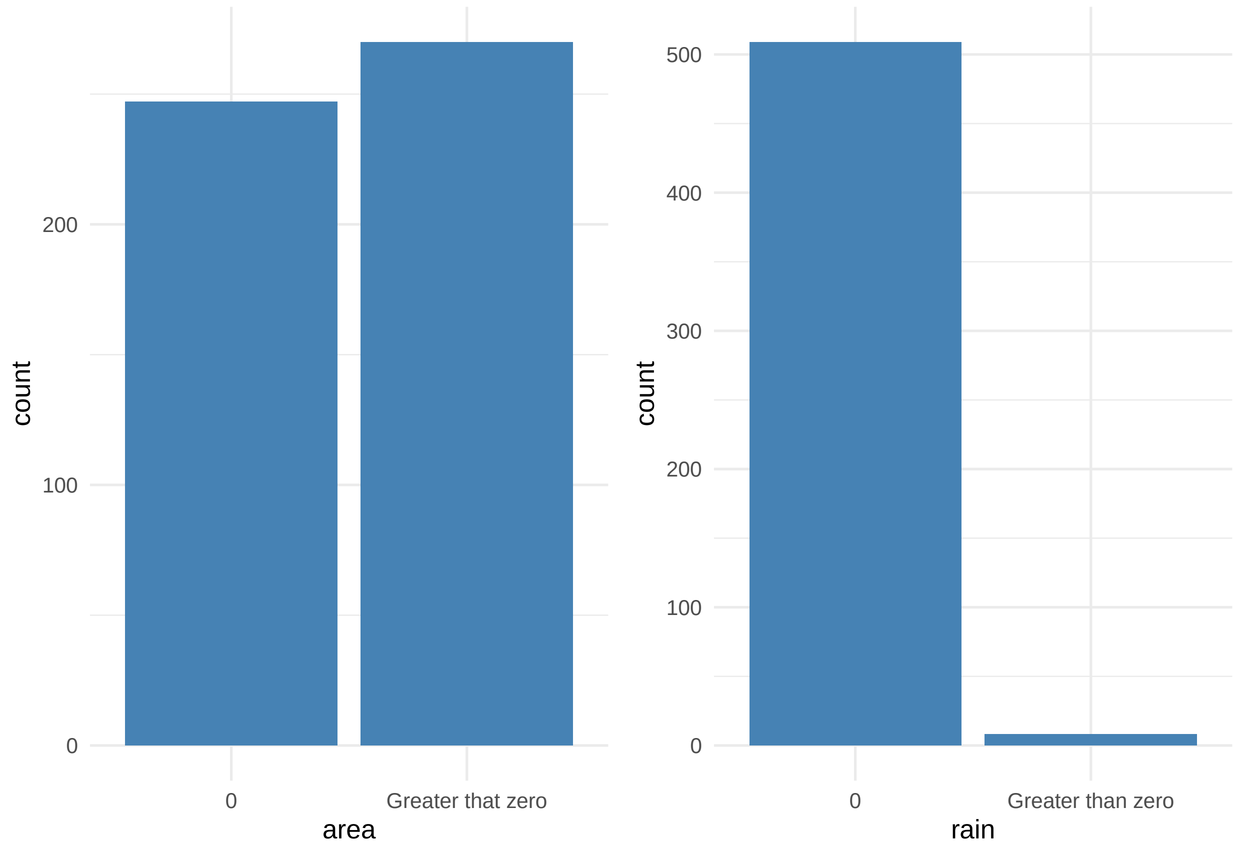


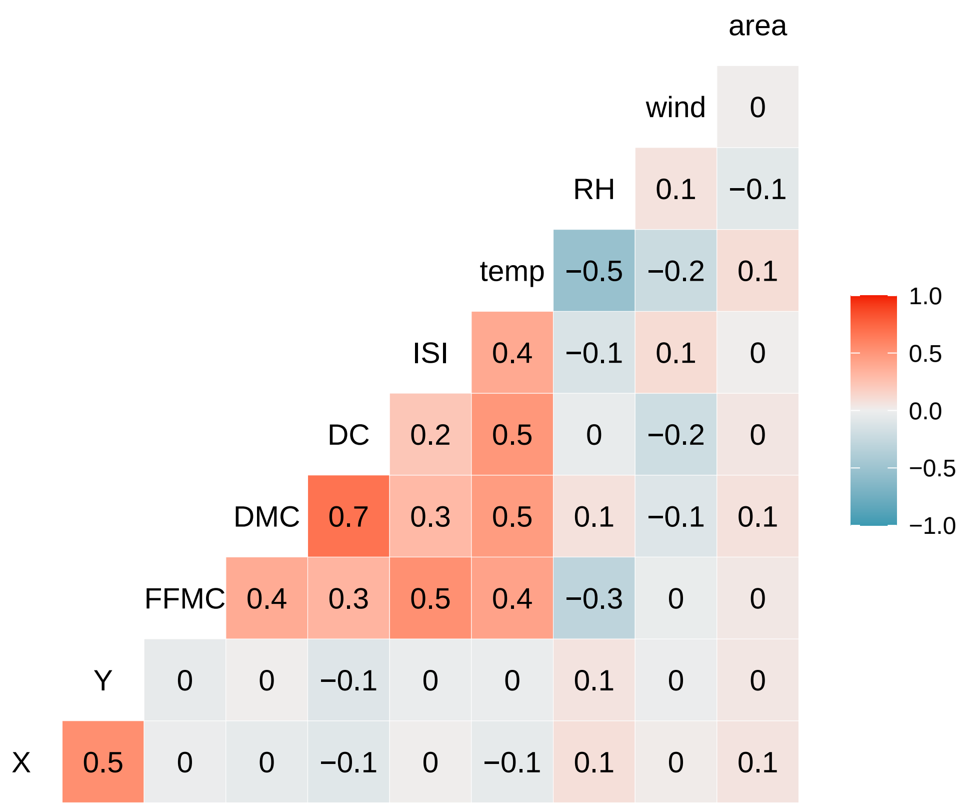
Figure 2: zero values histograms

The high skewness of these two variables and the high number of zero values will be tackled in the section 2.4 and again in the second part (3.2) when we will transform the analysis in a logistic regression task.

2.3 Correlations

In the regression analysis one main goal is to isolate the relationship between each independent variable and the dependent one. If two or more predictors are strongly correlated (multicollinearity), it becomes difficult to estimate their effects independently on the response variable.

```
#plotting correlation of pairs
drop_vars <- names(fires) %in% c("month","day","rain")
ggcorr(fires[!drop_vars],label = TRUE, label_size=4, label_color='black')
```



As we can see from the correlation plot, some variables are positively correlated (like *DC*-*DMC* and *X*-*Y*) and *temp*-*RH* are negatively correlated. By computing **VIF (Variable Inflation Factors)** we can determine the strength of the correlation between the independent variables and ensure no multicollinearity is present among the attributes. VIF exceeding 5 or 10 indicates high multicollinearity between that independent variable and the others and it should be removed from the model.

```
#VIF computation
round(vif(lm(log(area+1) ~ X + Y + FFMC + DMC + DC + ISI + temp + RH + wind,
             data = fires)),1)
```

```
##    X    Y FFMC  DMC   DC  ISI temp   RH wind
##  1.4  1.4  1.7  2.4  2.1  1.6  2.6  1.9  1.1
```

No VIF exceeding the threshold so we can safely assume that variables are independent.

2.4 Processing

In the previous histograms 2 we saw that *area* and *rain* are highly positively skewed with quite a lot of zero values. As suggested from the paper of Cortez and Morais (2007), to reduce the skewness and and improve symmetry, the $log(x + 1)$ transformation is applied to the *area* attribute.

```r
#skewness of the target variable
area_plot <-ggplot(data = fires)+
  geom_histogram(aes(area),binwidth = 40)+
  theme_minimal(base_size = 11)+
  xlab("Burned area (in hectares)")

#applying log transformation to get a more gaussian like shape and adding 1 to account
#for zero values
log_area_plot<-ggplot(data = fires)+
  geom_histogram(aes(log(area+1)),binwidth = 0.3,fill = "steelblue")+
  theme_minimal(base_size = 11)+
  xlab("Ln(area +1)")

grid.arrange(area_plot,log_area_plot,ncol=2)
```

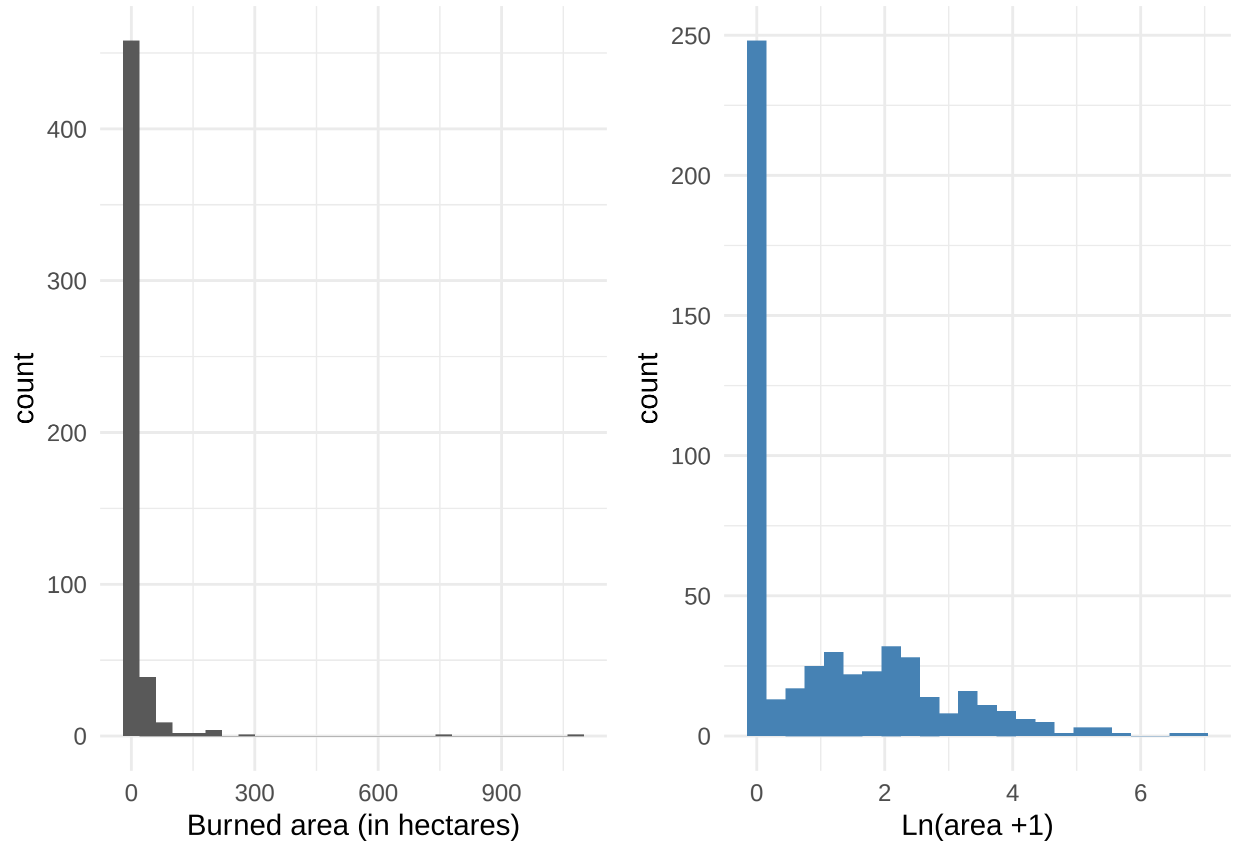


Figure 3: log-transformation of area

As we can see from the right histogram, the area is now less left-skewed and has a more gaussian shape, allowing us to stick with the normality assumption of the response variable necessary for linear regression. During the later experiments, *area* will always be $log(x + 1)$ transformed.

Before applying any transformation to *rain* let's observe that only 8 are non zero values and consequently the variance of this variable will be very low.

```
## rain values greater that zero =  8
## var[rain] =  0.0875918
```

I strongly believe that this variable will have a very low predictive power so it make sense to remove it and simplify the model.

Finally I decided to encode the spatial variables X and Y as categorical variables ranging from 1-9 and 2-9.

```r
fires$X <- as.factor(fires$X)
fires$Y <- as.factor(fires$Y)
#remove rain var
drop_cols <- names(fires) %in% c("rain")
fires_reg <- fires[!drop_cols]
```

3 Regression Models

As already anticipated in the Introduction, we will start by predicting *area* as a quantitative variable exploiting **General Linear Models** then we will transform *area* into a categorical response and build distinct **Generalized Linear Models**. In both the settings we will try to choose the best one using different methods.

3.1 General Linear Model

In the general linear model the main focus is to describe the probabilistic behavior of a set of a quantitative responses $y_1...y_n$, considered realization of normal random variables $Y_1...Y_n$, in terms of a set of predictors collected in a matrix X.

Mathematically, a general linear model is represented by the following linear combination

$$Y = X\beta + \epsilon$$

where X is a $n \times p$ matrix containing a first column of 1 (that represent the intercept of the regression line) followed by the values of the $p-1$ predictors for all the n samples, β is the column vector of dimension $p-1$ of parameters main object of the statistical inference and finally the ϵ n-dimensional column vector is a set of unobservable random variables (also called errors) which account for natural variability and other sources of uncertainty. This vector is assumed to represent a situation of normally distributed i.i.d errors added to the signal $X\beta$ with $\mu = 0$ and $var = \sigma^2 I_{n\times n}$.

The normality assumption of ϵ implies that Y will be normally distributed too with each element having the same variance

$$Y = X\beta + \epsilon \sim N_n(X\beta, \sigma^2 I_{n\times n})$$

3.1.1 OLS Regression

Thanks to the assumption of Gaussian noise we can find the line for which the probability of the data is highest by solving the following optimization problem:

$$min_\beta \sum_{i=1} (y_i - X_i\beta)^2$$

where min_β just means "find the value of β that minimize the following", X_i refers to the row i of the matrix X and y_i is the i-th element of the column vector of the realizations of Y.

By using some basic linear algebra to solve this minimization problem and assuming X^TX is invertible, we can find the optimal estimates of β

$$\hat{\beta}(X^TX)^{-1}X^Ty$$

and the corresponding estimator, normally distributed being a linear combination of Y,

$$\hat{\beta}(X^TX)^{-1}X^TY \sim N_p(\beta, \sigma^2(X^TX)^{-1})$$

.

To properly interpret the coefficients of the OLS model, the following statistical assumptions must be satisfied:

- **Normality:** For fixed values of the independent variables, the dependent variable is normally distributed
- **Independence:** The observations are independent of each other (already verified with VIF in 2.3)
- **Linearity:** The true relationship between the dependent variable and the independent variables is linear
- **Homoscedasticity:** The variance of the dependent variable doesn't vary with the levels of the independent variables.

In the following sections, we will select different subset of explanatory variables, compute the OLS estimate and, after choosing the best performing model, assess the satisfaction of the statistical assumptions to validate it.

3.1.2 Simple model

Let's first build a simple model with a single continuous predictor (*temp*) to give better insights on how to interpret a linear regression model. In the simple linear regression settings we are going to fit a line $y = \beta_0 + \beta_1 x$ to our data in order to find the *least squares line* (the best linear approximation to the true relation $area = \beta_0 + \beta_1 temp$ called *population regression line*). The minimization problem becomes

$$min_{\beta_0,\beta_1} \sum [y_i - (\beta_0 + \beta_1 x_i)]^2$$

and the solution is

$$\hat{\beta}_1 = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2}$$

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x}$$

where $\bar{x}$ and $\bar{y}$ are the sample means.

Below, the *summary()* results of the simple linear model are displayed:

```
#first simple linear regression model
lm.simple <- lm(log(area+1) ~ temp,data = fires_reg)
summary(lm.simple)
```

```
##
## Call:
## lm(formula = log(area + 1) ~ temp, data = fires_reg)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.2851 -1.1034 -0.7298  0.9278  5.8046
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.86771    0.20940   4.144 3.99e-05 ***
## temp         0.01288    0.01060   1.216    0.225
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.398 on 515 degrees of freedom
## Multiple R-squared:  0.002861,   Adjusted R-squared:  0.0009246
## F-statistic: 1.478 on 1 and 515 DF,  p-value: 0.2247
```

The *Residuals* section displays some points of the residuals distribution giving some insights on how well the model is performing. Residuals are calculated as $Y - \hat{Y}$ so high values of residuals means predictions far from the ground-truth.

In the *Coefficients* part we have the estimated values of the regression coefficients $(\hat{\beta}_0, \hat{\beta}_1)$ that represents the mean change in the response variable for each one unit change in the independent variable when you hold all the other independent variables constant; the Std. Error, useful to build confidence intervals on the true parameter β_i with the form $\hat{\beta}_i \acute{s} 2SE(\hat{\beta}_i)$; the t-values and the related p-values useful to reject or fail to reject the $H_0 : \beta_i = 0$ for a explanatory variable i, when all the others variables are in the model. In this simple regression model we see that the *intercept* ha a low p-value (reject $H_0 : \beta_0 = 0$) while *temp* has a high p-value (with a given confidence level the true value of β_1 is zero so it has no effect on the response variable). The same result can be deduced by observing the CI:

```
#computing confidence intervals for the simple model
confint(lm.simple)
```

```
##                     2.5 %     97.5 %
## (Intercept)  0.456323030 1.27909031
## temp        -0.007937741 0.03370056
```

having *temp* the zero value included in his confidence interval we make our theoretical believe to drop this variable stronger.

Finally in the last part we have the *Residual Standard Error* (an estimate of the standard deviation of the errors ϵ), the R^2 Statistics that provides a measure of fit in terms of the proportion of variance explained by the model and the *Global F Test*, to test the null hypothesis $H_0 : \beta_1 = \beta_2 = ... = \beta_{p-1} = 0$ basing on the p-value. In our case the high RSE and the low R^2 statistics indicates a bad performing model.

3.1.3 Complete Model

In this section we will create a model containing all the predictors in the dataset and see how it behaves.

```
lm.big <- lm(log(area+1) ~ .,data=fires_reg)
summary(lm.big)
```

```
##
## Call:
## lm(formula = log(area + 1) ~ ., data = fires_reg)
##
## Residuals:
```

```
##     Min      1Q  Median      3Q     Max
## -2.1195 -1.0121 -0.4162  0.7944  5.2104
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.877935   1.516448  -0.579  0.56290
## X2           0.038496   0.258470   0.149  0.88166
## X3          -0.727740   0.300525  -2.422  0.01583 *
## X4          -0.055463   0.272268  -0.204  0.83867
## X5          -0.555894   0.344813  -1.612  0.10759
## X6           0.018259   0.285047   0.064  0.94895
## X7          -0.217199   0.295662  -0.735  0.46293
## X8           0.112825   0.369459   0.305  0.76021
## X9           1.572099   0.592016   2.656  0.00818 **
## Y3           0.416611   0.322763   1.291  0.19741
## Y4           0.500716   0.274726   1.823  0.06899 .
## Y5           0.356298   0.289599   1.230  0.21918
## Y6           0.384162   0.380713   1.009  0.31346
## Y8           4.171124   1.429338   2.918  0.00369 **
## Y9          -1.982153   0.822335  -2.410  0.01631 *
## monthfeb     0.821635   1.132114   0.726  0.46835
## monthmar     0.201868   1.146243   0.176  0.86028
## monthapr     0.465576   1.200615   0.388  0.69835
## monthmay     1.225703   1.480916   0.828  0.40827
## monthjun     0.124635   1.193679   0.104  0.91689
##  [ reached getOption("max.print") -- omitted 19 rows ]
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.353 on 478 degrees of freedom
## Multiple R-squared:  0.1328, Adjusted R-squared:  0.06388
## F-statistic: 1.927 on 38 and 478 DF,  p-value: 0.001017
```

The reason why I decided to report the *summary()* of the complete model, even if 19 rows are omitted, is to point out how categorical variables with many levels (like X, Y, day and $month$ in our case) are handled and how to interpret them. In general, when a factor has l levels, $l-1$ dummy variables are used to code it, taking as reference the first level. The reason behind the $l-1$ is that we would generate linearly dependent columns in the X matrix, making $X'X$ a non invertible matrix, assumption necessary for OLS estimates.

For *day* and *month* the first levels taken as reference are:

```
## reference level for day:  mon
```

```
## reference level for month:  jan
```

Their coefficients are:

```r
#day coefficients
round(lm.big$coefficients[grep("day",names(coefficients(lm.big)))],3)
```

```
## daytue daywed daythu dayfri daysat daysun
##  0.229 -0.024 -0.024 -0.060  0.174  0.145
```

```r
#month coefficients
round(lm.big$coefficients[grep("month",names(coefficients(lm.big)))],3)
```

```
## monthfeb monthmar monthapr monthmay monthjun monthjul monthaug monthsep
##    0.822    0.202    0.466    1.226    0.125    0.843    1.046    1.744
## monthoct monthnov monthdec
##    1.639   -0.690    2.563
```

The coefficients represent the average area burnt with respect to the reference level and by analyzing the *day* factor we can see that the highest coefficients values are in the weekend (**weekend effect**) where the turism is higher and humans may have caused fires. As far as the *month*, December has the highest coefficient, and also here turism may play a key role in causing fires.

Similarly, by analyzing the coefficients of the spacial variables (X& Y) I believe we can observe the coordinates where the most of the fires took place and deduce information about the territory (how much forest, presence of rivers etc. . .).

3.1.4 Smaller models

Following the Cortez and Morais (2007) work, four distinct feature selection setups were tested:

- *STFWI*: spatial, temporal and the four FWI components;
- *STM*: spatial, temporal and three weather variables (originally were four, but I removed *rain* in section 2.4);
- *FWI*: the four FWI components;
- *M*: the three weather conditions;

```r
lm.STFWI <- lm(log(area+1) ~ X + Y + day + month + FFMC + ISI +
                 DMC + DC, data = fires_reg)
lm.STM <- lm(log(area+1) ~ X + Y + day + month + temp + RH +
               wind, data = fires_reg)
lm.FWI <- lm(log(area+1) ~ FFMC + ISI + DMC + DC, data = fires_reg)
lm.M <- lm(log(area+1) ~ temp + RH + wind, data = fires_reg)
```

3.1.5 Model Selection

3.1.5.1 ANOVA and AIC A good practice to select among nested models is to use the *ANOVA* test. We test the null hypothesis that the added predictors (i.e. the predictors not in common between the models) have zero coefficients using an F-statistic. We will now compute the *ANOVA* test among the two couple of nested models defined above (*lm.STFWI* & *lm.FWI*, *lm.STM* & *lm.M*), including also the complete one:

```r
#ANOVA for the two pairs of nested models
print(anova(lm.FWI,lm.STFWI,lm.big))
```

```
## Analysis of Variance Table
##
## Model 1: log(area + 1) ~ FFMC + ISI + DMC + DC
## Model 2: log(area + 1) ~ X + Y + day + month + FFMC + ISI + DMC + DC
## Model 3: log(area + 1) ~ X + Y + month + day + FFMC + DMC + DC + ISI +
##     temp + RH + wind
##   Res.Df     RSS Df Sum of Sq      F    Pr(>F)
## 1    512 1000.98
## 2    481  882.83 31   118.148 2.0818 0.0007082 ***
## 3    478  875.07  3     7.762 1.4134 0.2380880
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
print(anova(lm.M,lm.STM,lm.big))
```

```
## Analysis of Variance Table
##
## Model 1: log(area + 1) ~ temp + RH + wind
## Model 2: log(area + 1) ~ X + Y + day + month + temp + RH + wind
## Model 3: log(area + 1) ~ X + Y + month + day + FFMC + DMC + DC + ISI +
##     temp + RH + wind
##   Res.Df    RSS Df Sum of Sq      F   Pr(>F)
## 1    513 998.92
## 2    482 888.05 31   110.874 1.9537 0.001901 **
## 3    478 875.07  4    12.974 1.7717 0.133301
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Looking at p-values for the F-statistic, we can conclude that the data provide enough evidence to **reject the null** in the case of *lm.STFWI* and *lm.STM*, while we **fail to reject** the null in the case of the *lm.big*. This means we prefer the two "medium sized" models by now.

Another way of comparing models is the the *Akaike's Information Criterion*. *AIC* estimates the relative amount of information lost by a given model, focusing on the balance between the goodness of fit and the simplicity of the model. Lower values of information loss means higher quality of the model.

$$AIC = 2(p+1) - 2ln(\hat{L})$$

where $\hat{L}$ is the maximized likelihood function and p is the number of independent predictors in the model.

```
#AIC computation
as.matrix(AIC(lm.big,lm.STFWI,lm.STM,lm.FWI,lm.M))
```

```
##          df      AIC
## lm.big   40 1819.261
## lm.STFWI 37 1817.827
## lm.STM   36 1818.869
## lm.FWI    6 1820.762
## lm.M      5 1817.695
```

As a tradeoff between the results obtained in the *ANOVA* tests and the *AIC* values, the model that seems to better fit the data with the lowest information loss is *lm.STFWI*, probably because it contains the temporal features (that as we saw in the Complete model section can give some insights on human based fires), the spacial features (presumably information about the territory) and the *FWI* indexes that are built with the weather conditions.

3.1.5.2 K-fold Cross Validation To take a further step in the analysis of the performances of the presented models, we are gonna use **cross-validation**, a statistical tool to estimate the test error associated with our models. Setting K=10 as in the *Cortez* work, the data is splitted into K subset. Each of this partitions is gonna serve as test set for the model trained on the remaining K-1 subsets and the prediction error is recorded. Finally, when all the K subsets has served as test set, the average of the recorded errors is computed.

For the computation of the cross-validation the library *caret* will be used and for the the evaluation of test error we will observe:

- $RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y_i})^2}{n}}$ That is, the average difference between the observed known outcome values and the values predicted by the model. The lower the RMSE, the better the model.
- $R^2 = \frac{\sum_{i=1}^{n}(\hat{Y_i} - \bar{Y})}{\sum_{i=1}^{n}(Y_i - \bar{Y})}$ That is a measure of the variance explained by the model. The higher the R^2, the better the model.
- $MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y_i}|$ That is an alternative to $RMSE$ less sensitive to outliers. The lower the $RMSE$, the better the model.

We underline the fact that the errors refers to $log(area + 1)$ as the unit and that, since in all models the dependent variable follows the same transformation, we can safely compare the results.

```r
#defining training control
set.seed(1)
train.control <- trainControl(method = "cv", number = 10)

#Train the models
model.big <- train(log(area+1) ~ .,data = fires_reg,method="lm",
                   trControl = train.control)
model.STFWI <- train(log(area+1) ~ X + Y + day + month + FFMC + ISI + DMC + DC,
                     data = fires_reg,method = "lm",trControl = train.control)
model.STM <- train(log(area+1) ~ X + Y + day + month + temp + RH + wind,
                   data = fires_reg, method = "lm",trControl = train.control)
model.FWI <- train(log(area+1) ~ FFMC + ISI + DMC + DC,
                   data = fires_reg,method = "lm",trControl = train.control)
model.M <- train(log(area+1) ~ temp + RH + wind,
                 data = fires_reg,method = "lm",trControl = train.control)

#summarize results
models <- c("big","STFWI","FWI","STM","M")
RMSE <- c(model.big$results$RMSE,model.STFWI$results$RMSE,
          model.FWI$results$RMSE,model.STM$results$RMSE,model.M$results$RMSE)
R2 <- c(model.big$results$Rsquared,model.STFWI$results$Rsquared,
        model.FWI$results$Rsquared,model.STM$results$Rsquared,model.M$results$Rsquared)
MAE <- c(model.big$results$MAE, model.STFWI$results$MAE,
         model.FWI$results$MAE,model.STM$results$MAE,model.M$results$MAE)

res.df <- data.frame(RMSE,R2,MAE)
row.names(res.df) <- models
as.matrix(res.df)
```

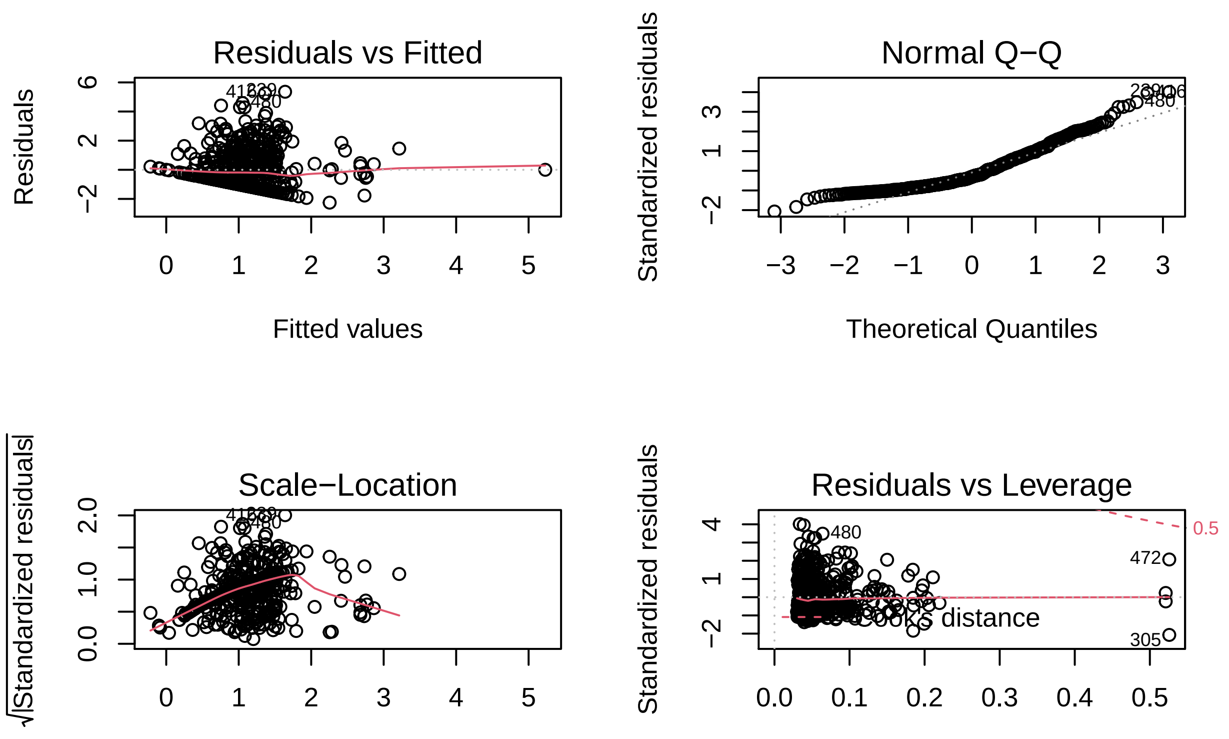
```
##           RMSE          R2      MAE
## big   1.430856 0.031079034 1.156746
## STFWI 1.412437 0.038618204 1.142064
## FWI   1.399780 0.030013518 1.162838
## STM   1.407295 0.034518546 1.130856
## M     1.395100 0.004542227 1.161189
```

Analyzing these results we can see that, considering the distribution of the log transformed *area* shown in the histogram 3, the errors are all pretty high. However we can observe that in terms of variability explained by the model, the *STFWI* is the one with better results but *STM* is the best performing one when we take also into consideration the prediction errors.

3.1.6 Validation of the model

As anticipated in the OLS theory section, we are now assessing the validity of the model by evaluating the statistical assumptions of regression analysis, using the residuals (remember that residuals are the **observable** distance $e = Y - \hat{Y}$) for diagnostic purposes.

```
par(mfrow=c(2,2))
plot(lm.STM)
```



The *Residual vs Fitted* graph is useful to check for the linearity assumption: a horizontal line without distinct pattern is what we were looking for. As far as the normality of the response variable, we would expected values in the *Normal Q-Q* plot that followed the straight dotted line. Finally the *Scale-Location* graph have an increasing behavior and the variability of the residuals increases with the value of the fitted outcome variable.

We can conclude that the Normality and the Homescedascity assumptions are not respected, as we were expecting since the bad performance obtained with the different evaluation methods. I believe the zero values of the dataset and the presence of outliers are the major cause to these results.

3.2 Generalized Linear Model

All the presented models based on linear regression are performing poorly and are not able to explain well the response variable.

In order to tackle the issue of the zero values in the dependent variable and to try to improve the predictions, similarly to the work by R. Rishickesh (2019), I moved to a **Logistic Regression** approach by encoding the *area* feature as a binary categorical feature.

3.2.1 Basic theory

Logistic Regression belongs to Generalized Linear Models that, as for the general linear model, are characterized by a response vector $Y_1, ..., Y_n$ of independent random variables with means $\mu_i = E(Y_i)$ and a linear combination of predictors $\eta = X\beta$. The new element with respect to the "old model" is a **link function** that depends on the family of the Y distribution and connect the means μ_i and the linear combination η:

$$g(\mu_i) = \eta_i$$

The typical choice for bernoulli distributed response variables (as in our case with the binary transformed *area*) is the *logit link*:

$$logit(p_i) = log(\frac{p_i}{1 - p_i})$$

that follows a sigmoid function which limits its range of probabilities between 0 and 1, so that it make sense to apply the linear combination

$$logit(p_i) = \sum_{j=0}^{p-1} \beta_j X_{ij}$$

If we apply the inverse transformation of the logit we obtain the *logistic distribution function (figure 4)*:

$$p_i = \frac{e^{logit}}{1 + e^{logit}}$$

where the probability $p_i = P(Y_i = 1|X_i)$ is the focus of the prediction.

```r
#define the logistic distribution function
logistic <- function(l)(exp(l)/(1+exp(l)))
#draw the general graph
curve(logistic,from = -10,to = 10,xlab = "l",ylab = "logistic(l)")
```

3.2.2 Logistic Regression Models

This section will be devoted to the creation of the equivalent models created for the General Linear Regression analysis and their evaluation. For this purpose we create a new categorical column in the dataset called *dangerous_fire* with two levels:

- **0** if the corresponding *area* value is equal to 0 (for not dangerous fires)
- **1** if the corresponding *area* value is greater than 0 (for dangerous fires)

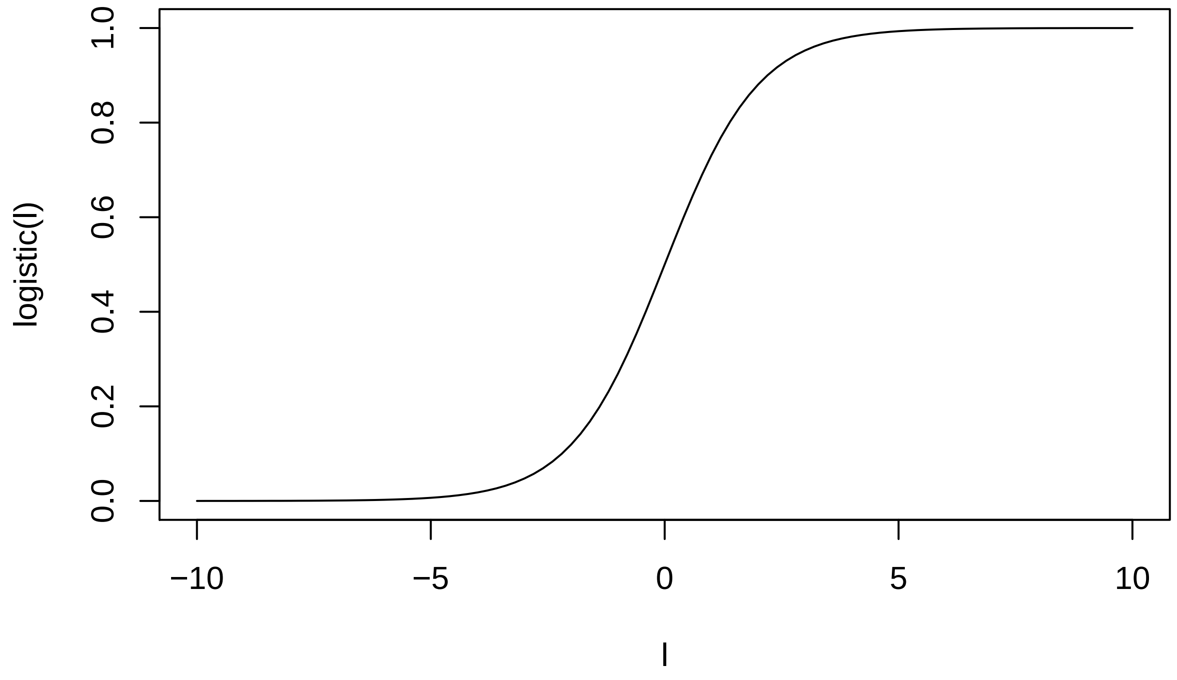


Figure 4: logistic distribution function

```r
#create the dataset with the new categorical variable
drop_area <- names(fires_reg) %in% c("area")
fires_cat <- fires_reg[!drop_area]

#set the 2 levels
fires_cat$dangerous_fire[fires$area == 0] <- 0
fires_cat$dangerous_fire[fires$area>0] <- 1

#set the new column as a factor
fires_cat$dangerous_fire <- factor(fires_cat$dangerous_fire,levels = c(0,1))
```

3.2.2.1 Simple model

Before diving into the main part of this GLM section, let's first build a simple model made by a continuous and a categorical independent variables, i.e. *temp* and *day*, to point out one big difference between linear and logistic models: the coefficient interpretation.

```r
#build the model
simple.glm <- glm(dangerous_fire ~ fires_cat$temp + fires_cat$day, data = fires_cat,
                  family = binomial(link = "logit"))

#extract the coefficients
coeff<-as.matrix(simple.glm$coefficients)
colnames(coeff)<-c("coefficients")
coeff
```

```
##                    coefficients
## (Intercept)         -0.33104882
## fires_cat$temp       0.02595498
## fires_cat$daytue     0.07896041
## fires_cat$daywed     0.17758416
## fires_cat$daythu    -0.14760557
## fires_cat$dayfri    -0.11121905
## fires_cat$daysat    -0.17123895
## fires_cat$daysun    -0.18506653
```

The coefficient of *temp* (0.03) in Logistic Regression correspond to the expected change in the **log odds** of being a big fire(*dangerous_fire = 1*) for a unit increase of *temp*, holding the other predictors at a fixed value (since as we saw it holds $logit(p_i) = \sum_{j=0}^{p-1} \beta_j X_{ij}$).

As far as the l-1 dummy variables created for *day* (where l=7) the coefficients represents the **odds ratio** of the odds that *dangerous_fire = 1* within that level of the categorical variable, compared to the odds of *dangerous_fire = 1* within the reference level (*daymon*). Let's take, for example, $\hat{\beta}_{daywed}$

```
## daywed coefficient: 0.18
```

If we encode with c all the other contributions to the linear combination except for *daywed*, we can write the logit as:

$$logit(p_i) = c + 0.18 daywed$$

Being *daywed* a dummy variable, we can write

$$logit(p_i) = logit(P(dangerous\_fire_i = 1)) = \begin{cases} c + 0.18 & \text{if } daywed = 1 \\ c & \text{if } daywed = 0 \end{cases}$$

Now we simply define

- $p_{i1} = P(dangerous\_fire_i = 1|daywed = 1)$
- $p_{i0} = P(dangerous\_fire_i = 1|daywed = 0)$ or equivalently $p_{i0} = P(dangerous\_fire_i = 1|daymon = 1)$

and we can write the *daywed* coefficient as

$$0.026 = log(\frac{\frac{p_{i1}}{1-p_{i1}}}{\frac{p_{i0}}{1-p_{i0}}})$$

that is the **logodds ratio**, a measure of comparing probabilities of success under two situations, in this case comparing the the probability of having a big fire being *daywed* with respect to its reference level *daymon*.

3.2.2.2 Smaller Models

Equivalently to the section 3.1.4, we will now create 4 different logistic regression models with different set of parameters and the complete one and evaluate their performances.

```
#create the four model inspired by Cortez work
glm.big <- glm(dangerous_fire ~ .,data = fires_cat,family = binomial)
glm.STFWI <- glm(dangerous_fire ~ X + Y + day + month + FFMC + ISI +
                 DMC + DC, data = fires_cat,family = binomial)
glm.STM <- glm(dangerous_fire ~ X + Y + day + month + temp + RH +
               wind, data = fires_cat, family = binomial)
glm.FWI <- glm(dangerous_fire ~ FFMC + ISI + DMC + DC,
               data = fires_cat,family = binomial)
glm.M <- glm(dangerous_fire ~ temp + RH + wind, data = fires_cat,
             family = binomial)
```

3.2.2.3 ANOVA Test and AIC

For the first comparison of the models we will use again *ANOVA Test* and *AIC*:

```
#ANOVA test
print(anova(glm.FWI,glm.STFWI,glm.big,test = "Chisq"))
```

```
## Analysis of Deviance Table
##
## Model 1: dangerous_fire ~ FFMC + ISI + DMC + DC
## Model 2: dangerous_fire ~ X + Y + day + month + FFMC + ISI + DMC + DC
## Model 3: dangerous_fire ~ X + Y + month + day + FFMC + DMC + DC + ISI +
##     temp + RH + wind
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1       512     709.61
## 2       481     646.38 31   63.234 0.0005517 ***
## 3       478     644.29  3    2.092 0.5534850
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
print(anova(glm.M,glm.STM,glm.big,test = "Chisq"))
```

```
## Analysis of Deviance Table
##
## Model 1: dangerous_fire ~ temp + RH + wind
## Model 2: dangerous_fire ~ X + Y + day + month + temp + RH + wind
## Model 3: dangerous_fire ~ X + Y + month + day + FFMC + DMC + DC + ISI +
##     temp + RH + wind
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1       513     709.71
## 2       482     646.61 31   63.101 0.0005728 ***
## 3       478     644.29  4    2.326 0.6760961
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

As for the linear regression, in both cases we **don't** have enough evidence to **reject the null** hypothesis that the added coefficients have $\hat{\beta}_i = 0$ in the medium models while we **fail to reject the null** in case of the big models.

```
#AIC
as.matrix(AIC(glm.big,glm.STFWI,glm.STM,glm.FWI,glm.M))
```

```
##           df      AIC
## glm.big   39 722.2883
## glm.STFWI 36 718.3805
## glm.STM   35 716.6140
## glm.FWI    5 719.6141
## glm.M      4 717.7148
```

Basing on *AIC* and the *ANOVA Test*, also in the logistic scenario the *glm.STM* model seems to be the best choice.

3.2.2.4 K-Fold Cross Validation

Finally as last step lets compute K fold cross validation (K=10) to see how the different models perform basing on a the accuracy metric:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

that is suitable in this situation since the two classes in which the target variable have been divided are quite balanced

```
##
##   0   1
## 247 270
```

```
#Train the models
glm_model.big <- train(dangerous_fire ~ .,data = fires_cat,method="glm", family= binomial,
                   trControl = train.control)
glm_model.STFWI <- train(dangerous_fire ~ X + Y + day + month + FFMC + ISI + DMC + DC,
                     data = fires_cat,method = "glm", family = binomial,trControl = train.control)
glm_model.STM <- train(dangerous_fire ~ X + Y + day + month + temp + RH + wind,
                   data = fires_cat, method = "glm",family = binomial,trControl = train.control)
glm_model.FWI <- train(dangerous_fire ~ FFMC + ISI + DMC + DC,
                   data = fires_cat,method = "glm", family = binomial,trControl = train.control)
glm_model.M <- train(dangerous_fire ~ temp + RH + wind,
                 data = fires_cat,method = "glm", family = binomial,trControl = train.control)
```

```r
#summarize results
glm_models <- c("big","STFWI","FWI","STM","M")
Accuracy <- c(glm_model.big$results$Accuracy,glm_model.STFWI$results$Accuracy,
          glm_model.FWI$results$Accuracy,glm_model.STM$results$Accuracy,glm_model.M$results$Accuracy)

glm_res.df <- data.frame(Accuracy)
row.names(glm_res.df) <- glm_models
as.matrix(glm_res.df)
```

```
##        Accuracy
## big   0.5880845
## STFWI 0.5783183
## FWI   0.5337481
## STM   0.5686652
## M     0.5240573
```

The highest accuracy score is from the complete model, but as we saw in *AIC* and *ANOVA Test* the better accuracy performance cannot justify the high number of parameters. Considering the previous results and the accuracy score, let's consider **glm.STM** the best model and observe the confusion matrix to have some better insights on the prediction errors:

```r
#divide the dataset into train and test set 80-20
smp_size <- floor(0.70 * nrow(fires_cat))

## set the seed to make the partition reproducible
set.seed(3)
train_ind <- sample(seq_len(nrow(fires_cat)), size = smp_size)

train <- fires_cat[train_ind, ]
test <- fires_cat[-train_ind, ]

#train the stm model on train subset
glm_train.STM <- glm(dangerous_fire ~ X + Y + day + month + temp + RH +
               wind, data = train, family = binomial)

#prediction on the test
predictTest = predict(glm_train.STM,newdata = test,type = "response")

# Confusion matrix on test set
confusion_matrix(glm_train.STM,test)
```

```
##          Predicted 0 Predicted 1 Total
## Actual 0          35          41    76
## Actual 1          27          53    80
## Total             62          94   156
```

From the confusion matrix we can observe that the model predict better the occurrence of large fires

4 Conclusions

On our work we tried to face the challenging task of predicting the area of burned forest due to fires and the magnitude of the fire. Even if we did not obtain high prediction accuracy, we could derive from the models important informations that can help preventing big forest fires by studying the coefficients of the predictors (i.e thanks the analysis of the *day* and *month* coefficients we saw that turism may have a big role in Montesinho Park fires).

I believe that limitations are directly related to the dataset, since we are missing some useful information like **future weather** or **fire fighter response** that may cause a fire break even if it was potentially a big fire. This may be the reason behind the *logistic regression* setting where many small fires are predicted as big fires.

References

Cortez, Paulo, and Anibal Morais. 2007. “A Data Mining Approach to Predict Forest Fires Using Meteorological Data.”

R. Rishickesh, A. Nayeemulla Khan, A. Shahina. 2019. “Predicting Forest Fires Using Supervised and Ensemble Machine Learning Algorithms.” *International Journal of Recent Technology and Engineering* 8.