École Polytechnique Fédérale de Lausanne

Modelling and predicting wildfires

Project report

Extreme Value Analysis 2021 Conference: Data Competition

Mongi Nouira, Thomas Blomet, Ji Won Min, Victoria Barenne
Supervised by Prof. Anthony Davison

Department of Mathematics
Spring 2021

Contents

1 Introduction

Wildfires are unplanned and unwanted fires that have an important impact on ecosystems and are one of the most common natural disasters. A wildfire can be characterized by its cause, the fuels responsible for its propagation and its severity. It is usually a great danger for the native vegetation, the animals and the human population so a lot of effort from governments, fire departments and experts is made in order to avoid and control wildfires which can be described as fire risk management. One of most useful tools are statistical models that predict the probability of fire occurrences and their intensity. First, it is useful to understand the phenomenon and prevent it. Second, it helps fire fighters plan their mission to stop already started fires.

Research shows that human activity has increased the risk of wildfires, either as a result of more human fire ignitions, accidental or intentional, and as a consequence of climate change. Indeed, what is known as fire weather, a combination of climate conditions ideal for wildfires to start and spread, has increased in duration in some regions. It is believed this change in fire weather is, in parts, due to global warming. Reciprocally, wildfires also have an effect on human health, either directly during the fire occurrence or indirectly through air pollution and loss of biodiversity.

Many variables have an influence on wildfires ignition and the size of the burnt area, in particular, the meteorological variables, the type of land covers (vegetation) and human activity. More precisely, three components are necessary for a fire to occur: oxygen, fuel and heat.

In this report, we describe the data challenge organized for the Extreme-Value Analysis program (EVA 2021). The challenge was to tackle the problem of predicting US wildfires count and the aggregated burnt area for the years 1993 until 2015. Several predictor variables such as meteorological variables and land covers were provided. We present several statistical models studied to predict the number of fires and the size of these fires. Some of them are more classical models such as generalized linear models, generalized additive models and extreme value models. Others are more machine learning oriented models such as decision trees, random forests and boosting methods. Moreover, we demonstrate the application of extreme value theory for large fires and the two final models submitted for the EVA data challenge.

2 Exploratory Data Analysis

We focus on two important components of wildfire activity: wildfire occurrence and size. A comprehensive wildfire dataset covering the period from 1993 to 2015 for the continental United States is used. The states of Alaska and islands such as Hawaii are excluded. The goal of this challenge is to estimate the predictive distributions of the number of wildfires (CNT) and aggregated burnt area (BA) of wildfires in acres for a validation set. The scores used for the competition are variants of weighted ranked probability scores but to summarize the lower the score the better.

2.1 Data Structure

The dataset contains the following variables as columns :

- CNT : number of wildfires (values to be predicted are given as NA)
- BA : aggregated burnt area of wildfires in acres (values to be predicted are given as NA)
- lon : longitude coordinate of grid cell center
- lat : latitude coordinate of grid cell center
- area : the proportion of a grid cell that overlaps the continental US (a value in (0,1], which can be smaller than 1 for grid cells on the boundary of the US territory)
- month : month of observation (integer value between 3 and 9)
- year : year of observation (integer value between 1 and 23, with 1 corresponding to year 1993 and 23 to year 2015)
- lc1 to lc18 : area proportion of 18 land cover classes in the grid cell
- altiMean, altiSD : altitude-related variables given as mean and standard deviation in the grid cell
- clim1 to clim10 : monthly averages of 10 meteorological variables in the grid cell

We are provided with 563,983 rows and 37 columns from which 111,053 rows contain a missing value either CNT or BA or both. There are exactly 80,000 missing values for the CNT and BA respectively. So one has multiple possibilities for training models :

First, to predict BA for example, we could use two models one using the available CNT for approximately 39% of the validation set and 61% without CNT. Second, we could also have a joint model for both response variables that could capture more properties such as the interaction between them. Finally, we could use only one model that predicts one response variable without using the other.

In real applications, the last two possibilities are appropriate to use for predictions since we don't know both variables future values. If we just want to understand the phenomenon of wildfires for non predictive risk management, then we could use the first possibility. For this challenge, since the aim is to have a better score, the first idea of using two models was used.

2.2 Spatial-Temporal and Response Variables

Looking at the spacial distribution of the mean number of fires across the USA [Figure 1], we can see that some regions such as California, Arizona, New Mexico, Georgia, South and North Carolina and New Jersey, have more wildfires than others. For the aggregated burned areas, California again, has large burnt area. We also see some states in the north-west that don't have a high number of fires but have the biggest fire sizes. Also, some states in the south-east seems to have a good control of fires since the size of burned area is rather small compared to the number of fires in those regions.

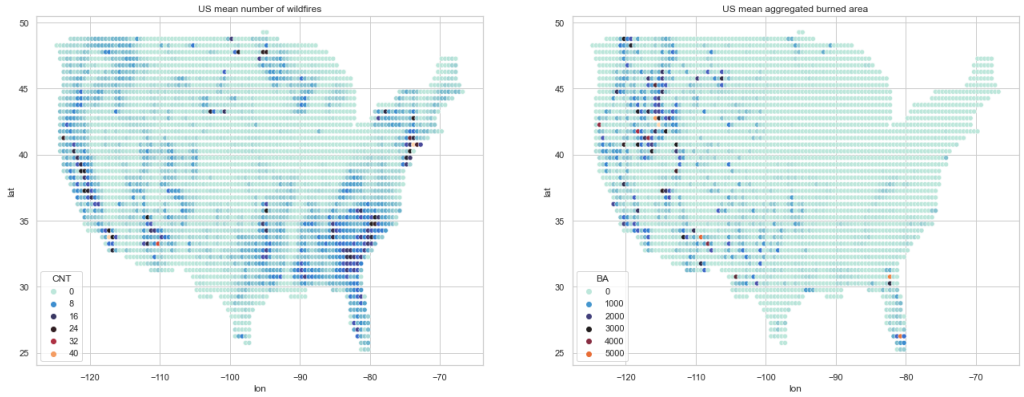

Figure 1: Spatial distribution of CNT and BA

We can also check if there is any correlation or a distribution pattern between the number of wildfires and the burnt areas [Figure 2]. We can see that a high number of fires don't necessarily mean a huge burned area and vice versa. Actually, most of the big fires may have started once and continued for several days, which might be why we see a peak for a fire count of one. Also, looking at the log transformation of both variables for fires only, we may see a small increasing trend and a concentration of BA for high number of fires. Again, a small (or even a single) number of fires varies in values of BA which is represented as vertical straight lines.

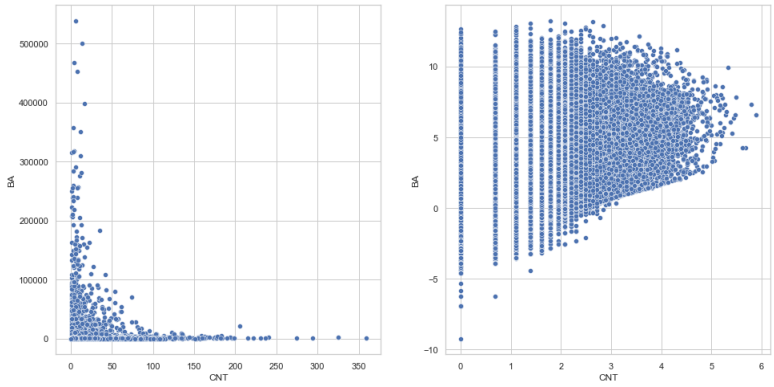

Figure 2: CNT vs BA

We can also see the distribution of the number of fires by year and month [Figure 3]. There is no particular trend for the evolution of the mean number of wildfires per year even if one can think

that global warming may have an effect, for example. It could be explained by the improvement of wildfires prediction. We also can see that the months of May, June and September have the lowest mean number of wildfires while April and July have the highest. This seems a little odd but it may be due to the dataset.

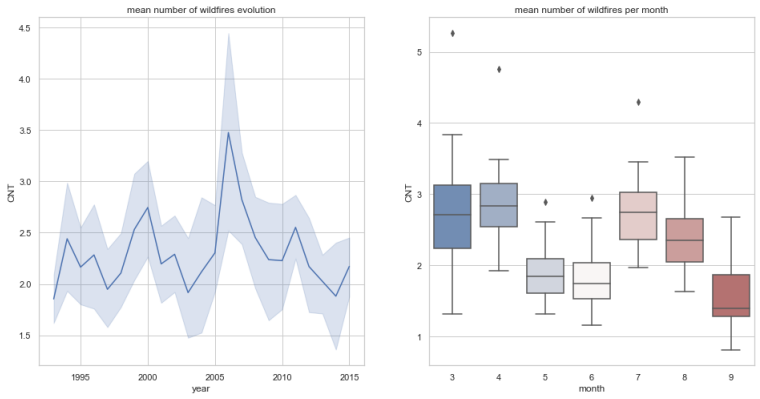

Figure 3: Temporal distribution of CNT

The following figures [Figure 4] represent the histograms of the two response variables we want to predict using only the strictly positive values on the logarithm scale.
The first plot suggests that fitting with a Poisson regression can be a good start for the search of the final model. We can use for example a generalized linear model (GLM) with the Poisson distribution and the logarithm as a link function.
The second is very interesting : It looks like a nearly perfect Gaussian distribution on the right side (BA $\geq$ 10) and a very noisy one on the left. This suggests that we could use a log normal distribution for positive values of BA and combine that with a zero/non-zero BA classifier.

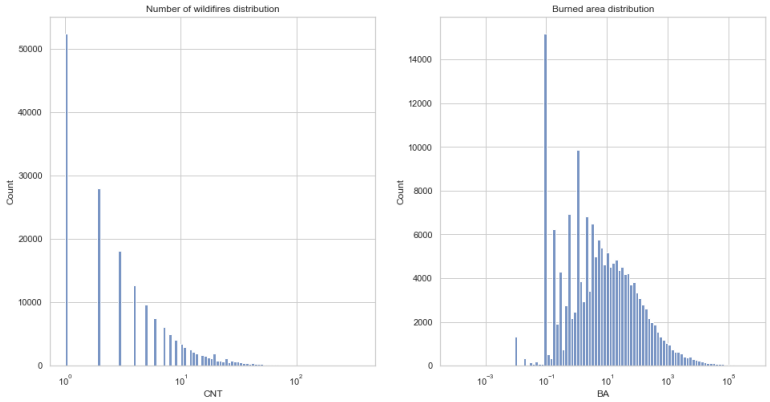

Figure 4: Non-Zero CNT and BA histograms 1

We can have a closer look at the burnt area distribution by separating the data into two parts using the threshold BA = 10 [Figure 5]. This suggests using three models : First a classifier for BA with classes BA=0, BA $\in$]0,10] and BA $>$ 10. Then a log-normal distribution for BA $\geq$ 10 and some kind of Poisson or Gamma regression for BA $\leq$ 10. All of these thoughts are useful for the

next section where we present the models.

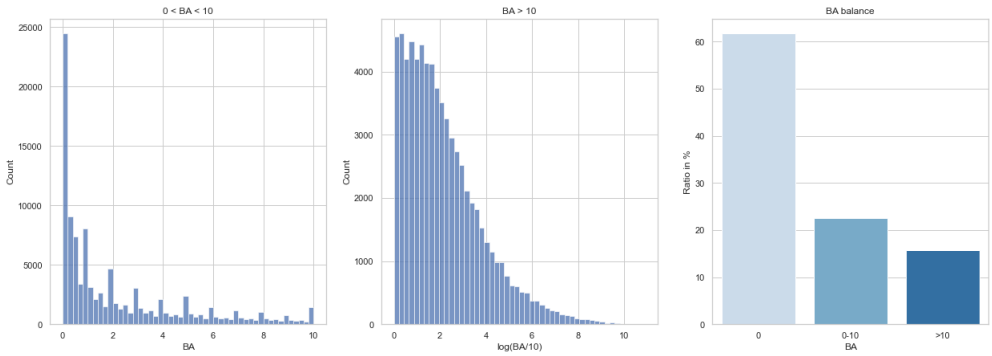

Figure 5: BA histograms

Going back to the data challenge, there are 28 severity thresholds that represent the levels of the distribution we want to predict. We can look at the histograms [Figure 6] using these thresholds as bins, using just non zero values. It looks like the data is highly unbalanced for CNT and BA but at the same time the scoring function of the challenge gives importance to extreme values so we should find a way to improve the modeling of rare events; we could use extreme distributions and also under sampling or over sampling for balancing the data if we plan to use some type of classification.

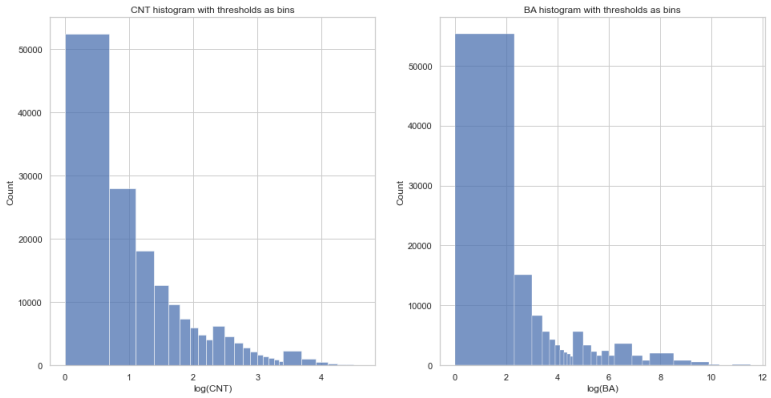

Figure 6: Non-Zero CNT and BA histograms 2

In the last parts, we removed the zeros for the analysis many times without talking about the balance of dataset. It is highly unbalanced since fires are quite "rare" events if we look at US as a whole (even urban areas with no forest). We actually have 62% of rows with no fires, 73% with one fire or less and 94% with 10 fires or less. As a remark, we previously suggested a Poisson regression for CNT but with such high number of zeros a Zero-Inflated Poisson regression may be better suited. To conclude, wildfires prediction isn't an easy task because of the unbalance of fire occurrences and the huge difference in fire sizes even if extreme values are quite rare. For the next parts, we will have a closer look at the available predictors which are land covers and meteorological variables for each grid cell at a given time.

2.3 Land Covers

In this part, we analyse the land cover variables which are labeled lc1 to lc18 and represent nearly the fraction of a particular type of land cover in the grid cell so their sum for a specific time and place is approximately 1. The description of each variable is as follows :

1. cropland rain fed
2. cropland rain fed herbaceous cover
3. mosaic cropland
4. mosaic natural vegetation
5. tree broad leaved evergreen closed to open
6. tree broad leaved deciduous closed to open
7. tree needle leaved evergreen closed to open
8. tree needle leaved deciduous closed to open
9. tree mixed
10. mosaic tree and shrub
11. shrub land
12. grassland
13. sparse vegetation
14. tree cover flooded fresh or brackish water
15. shrub or herbaceous cover flooded
16. urban
17. bare areas
18. water

We can start by looking at the Pearson's correlation between the land variables [Figure 7]. It seems that we do not have highly correlated land covariates, but we may have a non-linear correlations that were not captured by the Pearson's approach. We feel like we should not use that much variables when fitting models. One possible dimension reduction method is Principal Component Analysis but if we have a more "physics" approach to the problem, we could create a new feature that represents the percentage or fraction of "forest" which is the types of land cover that tends to burn during fires such as shrubs. In our case, it could be just the sum of the appropriate land covers and we should have a value in [0,1].

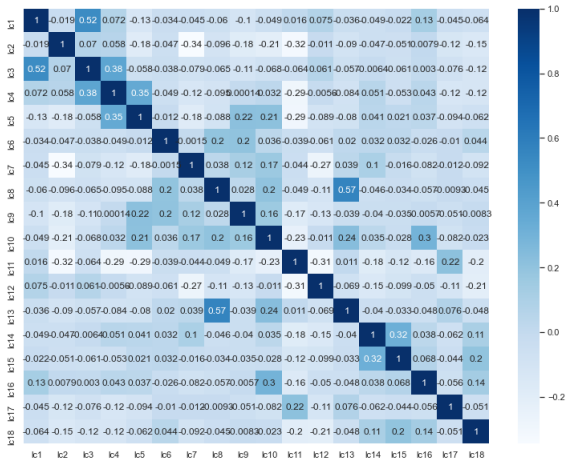

Figure 7: Pearson's correlation for land covers

We also plot the distribution of each land cover against CNT and BA [Figure 8-9]. First, we see that the scales of the variables are not the same even if they are all in $[0, 1]$. Some land covers have a rather small fraction of the land but we believe it doesn't mean they are not important. Also, we can see that some variables have no particular pattern at all so they may be not useful to use in the models or need to be transformed. Some land covers such as lc1, lc3, lc4, lc6, lc13 and lc17 have similar patterns; an increase of the land cover decreases the number of wildfires and the aggregated burned area.

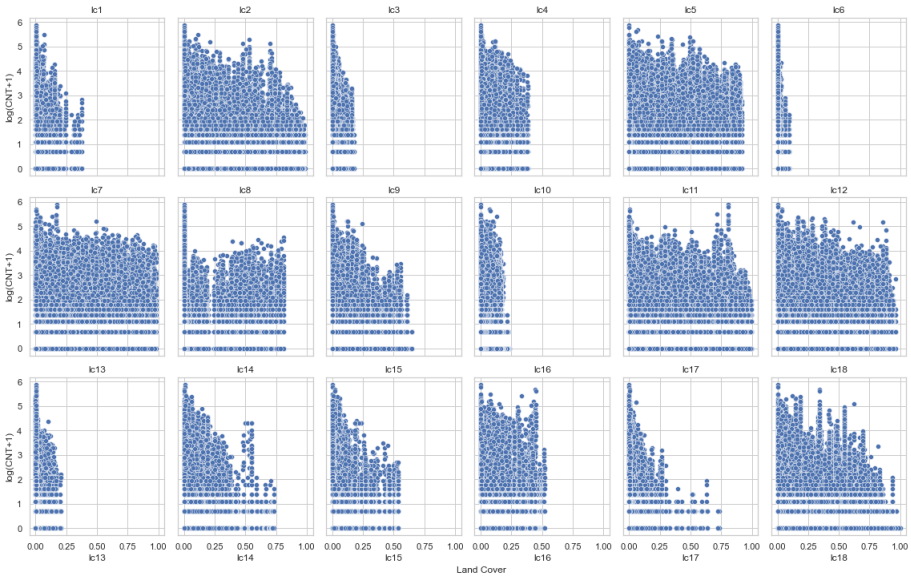

Figure 8: Land covers - CNT

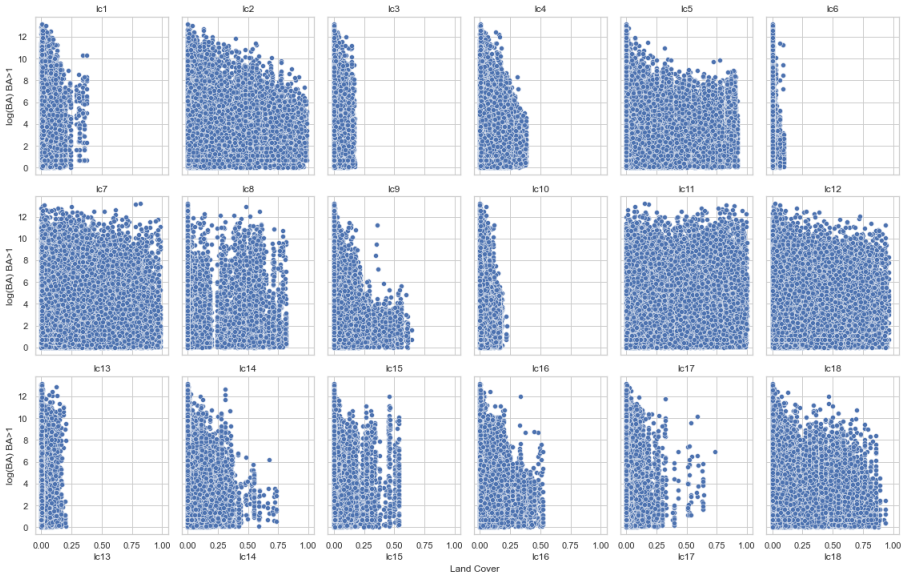

Figure 9: Land covers - BA

Another idea is to see the changes in land covers during fires. If a land type is a fuel for fires and its present before a fire occurrence, its ratio should decrease until the fire is ended. A grid cell with the highest average burned area (longitude -115.25 and latitude 42.25) was chosen, where the time series of BA for each land cover was plotted [Figure 10]:

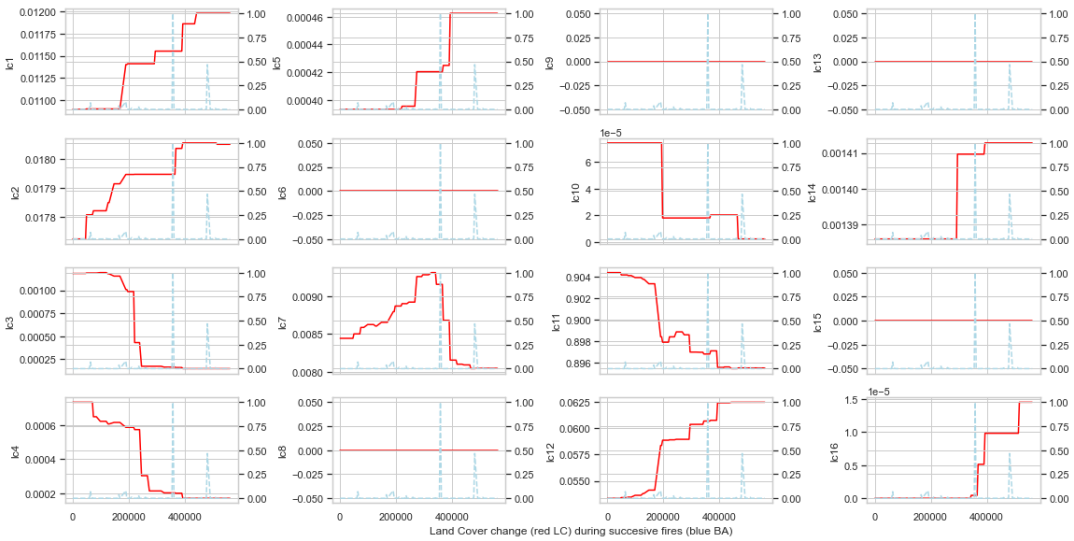

Figure 10: Land covers and burned area evolution

First, lc6, lc8, lc9, lc13 and lc15 are not present and don't change in time. The interesting land covers to see are lc7 and lc11. The land cover lc11 represents shrub land, which is a mature vegetation type in a particular region and remain stable over time, or a transitional community that occurs temporarily as the result of a disturbance, such as fire. A stable state may be maintained

by regular natural disturbance such as fire or browsing. Shrub land is known to be unsuitable for human habitation because of the danger of fire.
Speaking of population, we plotted the spatial distribution of urban density (lc16) across the US and it seems there is a correlation between urban density and wildfires. Indeed, more than 90% of wildfires are known to be induced directly or indirectly by human activity [2].

2.4 Meteorological Variables

In this part, we analyse the meteorological variables which are labeled clim1 to clim10. The description of each variable is as follows :

1. 10m U-component of wind (the wind speed in Eastern direction) (m/s)
2. 10m V-component of wind (the wind speed in Northern direction) (m/s)
3. Dewpoint temperature (temperature at 2m from ground to which air must be cooled to become saturated with water vapor, such that condensation ensues) (Kelvin)
4. Temperature (at 2m from ground) (Kelvin)
5. Potential evaporation (the amount of evaporation of water that would take place if a sufficient source of water were available) (m)
6. Surface net solar radiation (net flux of shortwave radiation; mostly radiation coming from the sun) (J/m2)
7. Surface net thermal radiation (net flux of long-wave radiation; mostly radiation emitted by the surface) (J/m^2)
8. Surface pressure (Pa)
9. Evaporation (of water) (m)
10. Precipitation (m)

We can start by looking at the Pearson's correlation between the climate variables [Figure 11]. There are a lot of highly correlated meteorological variables. Some of them are expected based on physics knowledge such as the two altitude related variables (0.7), the two temperature variables (0.76) and the mean altitude and surface pressure (-1). This needs some serious investigation and a dimension reduction before fitting models.

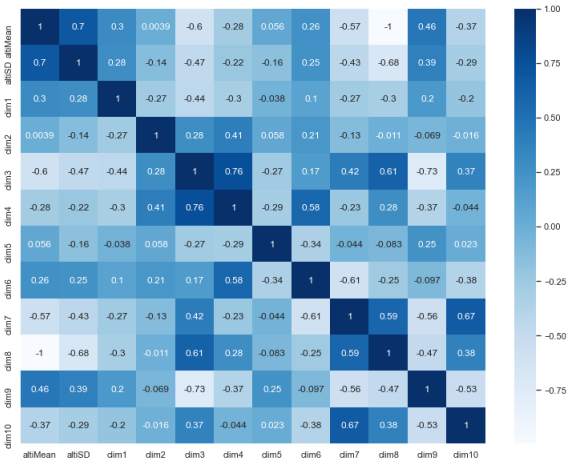

Figure 11: Pearson's correlation for climate variables

We also plot the distribution of each climate variable against CNT [Figure 12]. We use the min-max scaler for each variable to have the same scale order ($[0, 1]$).

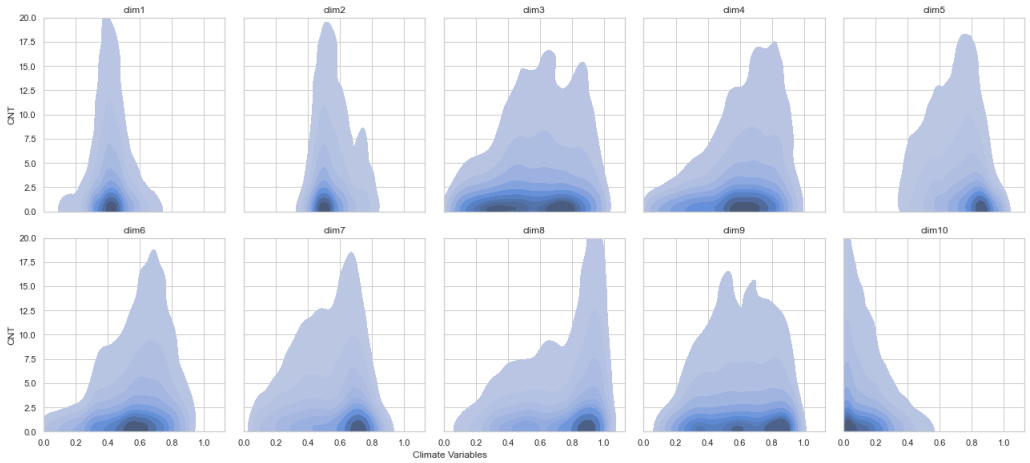

Figure 12: Meteorological Variables - CNT

We can see that precipitation (clim10) has a high density around zero CNT which is reasonable. The variables clim4, clim7 and clim8 share the same pattern where we have a higher number of fires for high values of these variables. The wind related variable clim1 and clim2 are nearly the same which means that the direction of wind doesn't matter but only the value does. One observation is the variables that seem correlated with CNT are used in the Fire Weather Index (FWI) [7], However, they are usually not used in a linear form. We could get inspired by the FWI to transform the variables.

2.5 New Variables and Scaling

Variables clim1 and clim2 was replace by a new feature, "wind", which is the square root of the mean square of the two variables. We also added a new variable "RH" representing relative

humidity, which was computed as an approximation using the available data [5]. We added a third variable "meanCNT", which represents the mean number of fires for each grid cell at a specific month. This was computed on the training set and copied to the validation set since we believe some regions have more common fires based on unavailable factors, which could be captured by the mean.
The land covers variables are in the range of [0,1] so we can keep them in that scale. For the climate variables, the scales are quite different, so they were standardized to have mean 0 and standard deviation 1.

2.6 Validation Set

No data was masked for uneven years (1993, 1995, ..., 2015) but only for even years (1994, 1996, ..., 2014). First, as mentioned before, there are 80,000 observations of CNT and BA respectively that must be predicted. For each variable, we have 39% of the other, available to use. Second, we don't have any particular pattern in terms of variables for the validation [Figure 13]. It seems that it is uniformly distributed even though the organizer of the challenge mentioned that "the spatial and temporal positions of validation data are not completely random, but they tend to be clustered in space and time" [8]. We may have missed an important aspect, so our work could be improved later.

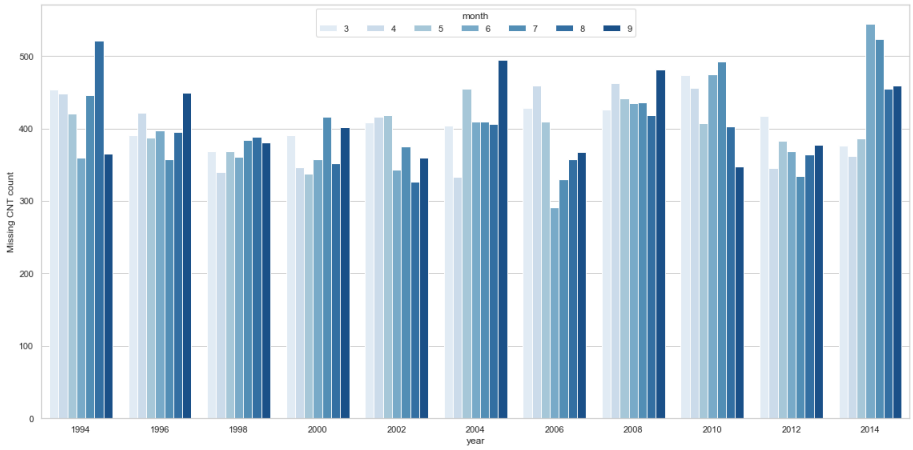

Figure 13: Temporal distribution of the validation set

We finally decided to use the years 1995, 2005 and 2015 for the training set and 1997 and 2007 for the testing set. This makes the models quicker to fit in order to compare several models and implementations. However, we usually use all the available data to fit the final model for the challenge. This is not a rule for every tested model. but at each step in the following parts of this report we should clearly explain which sample of the data we used to fit and test the model.

3 Evaluation of model performance and first models

3.1 Evaluation of Model Performance

The aim is to estimate predictive distributions for the BA and CNT data masked in the training dataset. More precisely, the value of the predictive distribution function of each NA observation of BA and CNT must be estimated for a list of severity thresholds.

We used the scoring functions provided by the competition to evaluate our models. Since the competition was in the context of an Extreme Value Analysis conference, naturally, more weight was given to predicting extreme values of CNT and BA accurately. Separate predictions scores are calculated for BA and CNT, and the final score is calculated as the sum of those two scores. A lower score corresponds to better predictions. To facilitate comparing models, we often use the scaled score, that is the score divided by the number of observations in the testing set.

We denote by $\hat{p}_{CNT,i}(u)$ the predicted probability for $\mathbb{P}(CNT_i \leq u)$ and by $\hat{p}_{BA,i}(u))$ the predicted probability for $\mathbb{P}(BA_i \leq u)$. There are $k_{CNT} = k_{BA} = 80,000$ values to be predicted for CNT and BA, respectively.

For the counts CNT to be predicted, we use the score:

$$S_{CNT} = \sum_{i=1}^{k_{CNT}} \sum_{u \in \mathcal{U}_{CNT}} w_{CNT}(u)(\mathbb{1}\{u \geq CNT_i\} - \hat{p}_{CNT,i}(u))^2,$$

where the weight function is given as:

$$w_{CNT}(u) = \frac{\tilde{w}_{CNT}(u)}{\tilde{w}_{CNT}(28)},\ \tilde{w}_{CNT}(u) = 1 - (1 + \frac{(u+1)^2}{1000})^{-\frac{1}{4}},$$

and the severity thresholds are:

$$\mathcal{U}_{\mathcal{CNT}} = \{u_1, \ldots, u_{28}\} = \{0, 1, 2, \ldots, 9, 10, 12, 14, \ldots, 30, 40, \ldots, 100\}.$$

For the burnt areas BA to be predicted, the score is:

$$S_{BA} = \sum_{i=1}^{k_{BA}} \sum_{u \in \mathcal{U}_{BA}} w_{BA}(u)(\mathbb{1}\{u \geq BA_i\} - \hat{p}_{BA,i}(u))^2,$$

where the weight function is given as:

$$w_{BA}(u) = \frac{\tilde{w}_{BA}(u)}{\tilde{w}_{BA}(28)},\ \tilde{w}_{CNT}(u) = 1 - (1 + \frac{(u+1)}{1000})^{-\frac{1}{4}},$$

and the severity thresholds are:

$$\mathcal{U}_{\mathcal{BA}} = \{u_1, \ldots, u_{28}\} = \{0, 10, 20, 30, \ldots, 100, 150, 200, 250, 300, 400, 500, 1000, 1500, 2000, 5000, 10000, 20000, 30000, 40000, 50000, 100000\}.$$

3.2 CNT Benchmark Model

The benchmark model given for CNT is a generalized linear model, with Poisson response distribution and log-link, using all available explanatory variables.

A generalized linear model (GLM) is formulated as follows:

$$g(\mathbb{E}[Y_i]) = X_i\beta = \beta_0 + \beta_1 x_i 1 + \cdots + \beta_p x_i p,$$

where g is a smooth monotonic "link function", β is a vector of unknown parameters, X_i is the i-th row of the model matrix and Y_i is the i-th response variable assumed to follow some exponential family distribution (assumed to be independent from other response variables).

Poisson distribution is the natural candidate for count data since it takes integer-valued and non-negative values. A log-link is used to force the mean to be non-negative. Indeed, the term $X\beta$ takes values from negative infinity to infinity, and by taking the exponential to recover the mean μ_i, we get a positive mean.

There were quite a few indicators that this model performed poorly in fitting the training data and predicting new data. Firstly, the coefficients for the explanatory variables seemed counter-intuitive to the effects they should have on CNT. For example, temperature had a negative coefficient, indicating that an increase in temperature leads to a decrease in the number of fire occurrences. Or inversely, precipitation had a positive coefficient, indicating that an increase in rainfall results in a increase of fire occurrences. Secondly, the residual deviance had value 536,292 for 73,562 degrees of freedom, indicating a significant over-dispersion and suggesting that the Poisson distribution might not be appropriate. Lastly, it performed very badly in predicting new data. It had a scaled score of 0.07390794 when trained on the years 1995, 2005 and 2015.

We observe that the mean CNT for the whole dataset is around 2.28. For that mean, assuming CNT follows a Poisson distribution, the probability of getting no fire occurrence should then be $e^{-2.285291} = 0.1017445$. But in our data, we observe around 61.69494% of zero counts, which is well above the 10.17445% that should be observed if the Poisson distribution was appropriate.

Potential solutions are considering other distributions such as a Quasi-Poisson which allows for a larger variance or a zero-inflated Poisson distribution. We explored the latter option.

3.3 Zero-inflated Poisson GLM

A zero-inflated Poisson density is a mixture of a point mass at 0 and a Poisson distribution. Thus, there are two possible sources of zeros. The zero-inflated Poisson distribution can be written as:

$$\mathbb{P}(Y = k) = p \cdot \mathbb{1}\{k = 0\} + (1 - p)\frac{k^\lambda e^{-\lambda}}{k!},$$

where $p \in]0, 1[$, $\lambda > 0$ and $k = 0, 1, 2, \cdots$.

A zero-inflated Poisson model seems to be very appropriate in our case. Indeed, for there to be a possibility of a fire occurrence, some conditions are necessary. For example, we can assert with

certainty that if a voxel is composed of 100 % of water, the probability of a fire occurrence in that voxel will be 0. We can assume there are lots of other conditions that need to be satisfied for a fire to possibly occur (temperature over a threshold, relative humidity under a threshold, etc ...).

We implemented a zero-inflated model in R using the `'pscl'` package, and more precisely the `zeroinfl` function. Using that function, we can then specify different set of regressors for the count component and the zero component. Here we used all explanatory variables except year and BA for both the count and zero components.

```
# fits the zero-inflated model using a Poisson Zero-inflated distribution
zeroinfl(formula = CNT ~ . - BA - year, data = trainingset, dist = "poisson")

# gives p the probability of getting a zero resulting from the point mass process
predict(model.zeroinfl1, type="zero", newdata=testingset)

# gives the mean of the Poisson process
predict(model.zeroinfl1, type="count", newdata=testingset)
```

Using the probability of a zero count resulting from the point mass process and the mean of the Poisson process, we recovered the Zero-inflated distribution for each observation to be predicted and got a scaled score of 0.07240156, a slight improvement on the benchmark model.

3.4 BA Benchmark Model

The suggested benchmark model for BA was a combination of the probabilities of no fire from the CNT benchmark model and a simple linear model on the $\log(BA)$ for the positive value of BA only using the GLM package. Using the years 1995, 2005 and 2015 as training and the years 1997 and 2007 as testing, we got a score of 3,854 when scaled to 80,000 observation. It corresponds to a scaled score of 0.0048.

4 Generalized Additive Models

In this section, we will first make a brief introduction to Generalized Additive Models (GAMs). Second, in order to estimate our probabilities, we will model GAMs based on our data.

4.1 GAM theory

4.1.1 Writing the model

A standard generalized additive model has the form

$$g(\mu_i) = \mathbf{A}_i\alpha + \sum_j f_j(x_{ji}),\ y_i \sim \mathrm{EF}(\mu_i, \phi)$$

where $\mathbf{A}_i$ is the i^{th} row of a parametric model matrix, with corresponding parameters α, f_j is a smooth function of (possibly vector) covariate x_j, and $\mathrm{EF}(\mu_i, \phi)$ denotes an exponential family distribution with mean μ_i and scale parameter ϕ. The y_i are modelled as independent given μ_i.

We choose smoothing basis and penalties for each f_j, which imply matrices $\mathcal{X}^{[j]}$ and $\mathcal{S}^{[j]}$, respectively model and penalty matrices. If $b_{jk}(x)$ is the k^{th} basis function for f_j, then $\mathcal{X}^{[j]}_{ik} = b_{jk}(x_{ji})$. To ensure that there will not be confoundedness with the intercept included in $\mathbf{A}$, we have to apply an identifiability constraint to any smooth which contains $\mathbf{1}$ in the span of its $\mathcal{X}^{[j]}$.

We need to create a model matrix $\mathbf{X}$. First, an identifiability constraint has to be applied to each smooth. Constraints that orthogonalise the smooth to the intercept term lead to sharpest inference about the constrained components, i.e. $\sum_j f_j(x_{ji}) = 0$. By reparametrization, these constraints are most usefully absorbed into the basis. Let $\mathcal{X}^{[j]}$ and $\mathcal{S}^{[j]}$ denote the model and penalty matrices for f_j after this reparametrization. Now we can combine $\mathbf{A}$ and the $\mathcal{X}^{[j]}$, column-wise, to create the needed model matrix

$$\mathbf{X} = (\mathbf{A} : \mathcal{X}^{[1]} : \mathcal{X}^{[2]} : \mathcal{X}^{[3]} : \cdots).$$

The corresponding model coefficient vector β contains α and the individual smooth term coefficient vectors stacked on end, $\mathcal{X}^{[j]}$. We are able to write a total smoothing penalty as

$$\sum_j \lambda_j \beta^T \mathbf{S}_j \beta,$$

where λ_j is a smoothing parameter and $\mathbf{S}_j$ is $\mathcal{S}^{[j]}$ embedded as a diagonal block in a matrix otherwise containing only zeros entries such that $\lambda_j\beta^{\mathrm{T}}\mathbf{S}_j\beta$ is the penalty for f_j.

Our model is then a generalized linear model

$$g(\mu_i) = \mathbf{X}_i\beta,\ y_i \sim \mathrm{EF}(\mu_i, \phi),$$

which can be estimated by maximization of

$$l_p(\beta) = l(\beta) - \frac{1}{2\phi}\sum_j \lambda_j \beta^T \mathbf{S}_j \beta.$$

The λ_j are smoothing parameters which control the trade-off between goodness of fit of the model and model smoothness.

To estimate β given λ, we are able to maximize the previous equation via the penalized iteratively re-weighted least squares (PIRLS) iteration :

1. Initialize $\hat{\mu}_i = y_i + \delta_i$ and $\hat{\eta}_i = g(\hat{\mu}_i)$, where δ_i is usually zero, but may be a small constant ensuring that $\hat{\eta}_i$ is finite. Iterate the following two steps to convergence.

2. Compute pseudodata $z_i = g'(\hat{\mu}_i)()y_i - \hat{\mu}_i)/\gamma(\hat{\mu}_i) + \hat{\eta}_i$, and iterative weights $w_i = \gamma(\hat{\mu}_i)/\{g'(\hat{\mu}_i)^2 V(\hat{\mu}_i)\}$.

3. Let $W = \text{diag}(w_i)$ and find $\hat{\beta}$ to minimize the weighted least squares objective

$$\|z - \mathbf{X}\beta\|_W^2 + \sum_j \lambda_j \beta^{\mathrm{T}} \mathbf{S}_j \beta$$

and then update $\hat{\eta} = \mathbf{X}\hat{\beta}$ and $\hat{\mu}_i = g^{-1}(\hat{\eta}_i)$.

We define $\|a\|_W^2 = a^{\mathrm{T}} W a$ and $V(\mu)$ the variance function determined by the exponential family distribution, while $\gamma(\mu_i) = [1 + (y_i - \mu_i)\{V'(\mu_i)/V(\mu_i) + g''(\mu_i)/g'(\mu_i)\}]$.

To estimate scale parameter ϕ we can show that an appropriate REML (restricted maximum likelihood) estimator is

$$\hat{\phi} = \frac{\|z - \mathbf{X}\hat{\beta}\|_W^2}{n - \tau},$$

where

$$\tau = \text{tr}\{(\mathbf{X}^{\mathrm{T}} W \mathbf{X} + \mathbf{S}_\lambda)^{-1} \mathbf{X}^{\mathrm{T}} W \mathbf{X}\},$$

and $\mathbf{S}_\lambda = \sum_j \lambda_j \mathbf{S}_j$. Then we can interpret τ as the effective degrees of freedom of our model.

4.1.2 Smoothing basis

We would like to represent the smooth functions in GAMs of section 4.1.1. To do so we need smoothing basis of certain dimension to reduce function approximation error. To better understand, we will consider only one dimensional smoothing spline.

For a set of points $\{(x_i, y_i) : i = 1, \ldots, n\}$ where $x_i < x_{i+1}$, natural cubic spline, $h(x)$, interpolating these points, is a combination of cubic polynomials, one for each $[x_i, x_{i+1}]$ interval which are joined together such that the whole spline is continuous to second derivative, while $h(x_i) = y_i$ and $h''(x_1) = h''(x_n) = 0$. Of all functions that are on $C^1([x_1, x_n])$, $h(x)$ is the one minimizing :

$$J(f) = \int_{x_1}^{x_n} f''(x)^2 dx,$$

i.e. in this sense $h(x)$ is the smoothest.

In our case, we do not want to interpolate $\{(x_i, y_i)\}_{i=1}^n$ but to smooth them. Then, in order to do that, we will treat the $h(x_i)$ as n free parameters of the cubic spline. Of all functions that are on $C^1([x_1, x_n])$, smoothing spline $h(x)$ is the one minimizing:

$$\sum_{i=1}^{n} \{y_i - f(x_i)\}^2 + \tau \int f''(x)^2 dx,$$

where τ is a tunable parameter, used to control the relative weight to be given to the conflicting goals of matching the data and producing h.

4.1.3 Smoothness selection

We can base smoothing parameter estimation on mean square prediction error, which is

$$P = \sigma^2 + \mathbb{E}\left(\frac{\|\mu - \mathbf{X}\hat{\beta}\|}{n}\right)$$

Supposing smoothing penalties are a Gaussian prior on the model coefficients, choosing smoothing parameters, λ, that maximize the Bayesian log marginal likelihood :

$$\mathcal{V}_r(\lambda) = \log \int f(\mathbf{y} \mid \beta) f(\beta) d\beta,$$

the log of the joint density of the data and coefficients β, with the coefficients integrated out. This approach is known as empirical Bayes.

4.2 Implementation in R

4.2.1 Introduction of the 'mgcv' package

Package 'mgcv' will be used to fit our GAMs models, this package is the one used and cited most times for GAMs.

The main modelling functions of this package are `gam`, basic function for GAMs; `bam`, same as `gam` but designed for large datasets; and `gamm`, for generalized additive mixed models. Due to the size of our data set we will only use the `bam` function which is better to fit GAMs for large data sets.

There are several smooths to use for `bam` function. First, there is three types of useful smooths that we can use : `s()` for univariate smooths, isotropic smooths of several variables and random effects; `te()` for tensor products smooths constructed from any single penalized marginal smooths; and `ti()` for tensor product interactions with the marginal smooths excluded. Principal arguments to all these functions are the covariates but we can specify others such as `bs` for the type of basis that we would like to use, `k` for the basis dimension, or `m` for the order of the basis and penalty.

In the whole chapter, our training set will be made of years 1995, 2005 and 2015 and our testing set of years 1997 and 2007. For each model that will be presented, `BA` variable will not be part of the model.

4.2.2 GAMs for CNT

First, we will take a look at the following model :

```
gam1CNT<- bam(CNT ~ s(lon,lat) + s(area,bs="cr") + month + s(lc1,bs="cr")
              + s(lc2,bs="cr") + s(lc3,bs="cr") + s(lc4,bs="cr") + s(lc5,bs="cr")
              + s(lc6,bs="cr") + s(lc7,bs="cr") + s(lc8,bs="cr") + s(lc9,bs="cr")
              + s(lc10,bs="cr")+ s(lc11,bs="cr")+ s(lc12,bs="cr")+ s(lc13,bs="cr")
              + s(lc14,bs="cr")+ s(lc15,bs="cr")+ s(lc16,bs="cr")+ s(lc17,bs="cr")
              + s(lc18,bs="cr")+ s(altiMean,bs="cr") + s(altiSD,bs="cr")
              + s(relative_humidity,bs="cr") + s(wind,bs="cr") + s(clim3,bs="cr")
              + s(clim4,bs="cr") + s(clim5,bs="cr") + s(clim6,bs="cr")
              + s(clim7,bs="cr") + s(clim8,bs="cr") + s(clim9,bs="cr")
```

```
            + s(clim10,bs="cr") + s(mean_monthly_fires,bs="cr"),
            data = train_data, family = poisson(link="log"))
```

Here, we fit a Poisson model with a log link function and all covariates that we could used, except year, and using cubic regression splines as basis for each smooth, except for `lat` and `lon`. The parameters of smooths are `k = 9` and `bs = "cp"` for cubic regression splines and `bs = "tp"` for thin plate regression splines as smooth basis (which is the default) with maximum degree of freedom of 9. A little summary of our model is obtained with `gam.check()` which compute a table with basis dimension results for each smooths and plots [Figure 14] concerning our model, e.g. Q-Q plot.

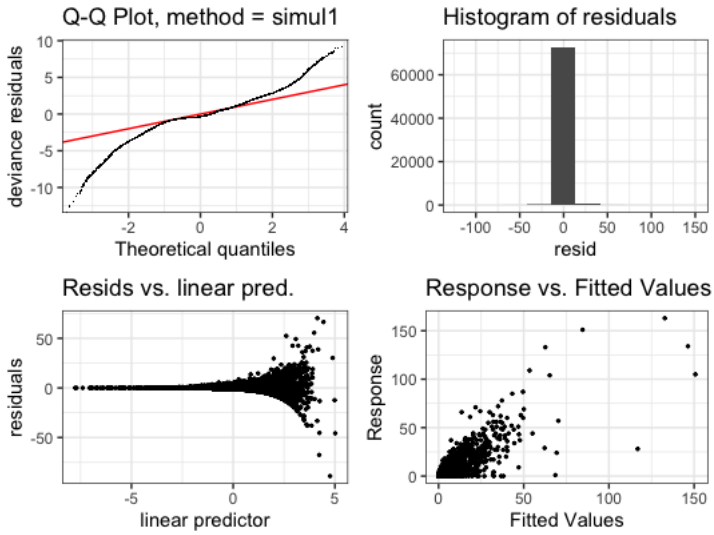

Figure 14: Plots of `gam.check(gam1CNT)`

We can see that the Q-Q plot is heavy tailed, this is expected since we remarked overdispersion. Histogram of residuals suggests that the variance is normally distributed. The residuals vs linear predictor is as expected since we use Poisson family in our model. The response vs fitted values plot is good for low values but clearly shows overdispersion for higher values.

```
##
## Method: fREML   Optimizer: perf newton
## full convergence after 22 iterations.
## Gradient range [-0.0005317471,0.0001601857]
## (score 132001.4 & scale 1).
## Hessian positive definite, eigenvalue range [0.0004177109,12.38539].
## Model rank =  319 / 319
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##                          k'   edf k-index p-value
```

```
## s(lon,lat)            29.00 28.28    0.95   0.715
## s(area)                9.00  7.62    0.94   0.570
## s(lc1)                 9.00  5.35    0.87  <2e-16 ***
## s(lc2)                 9.00  5.08    0.90  <2e-16 ***
## s(lc3)                 9.00  7.04    0.92   0.025 *
## s(lc4)                 9.00  1.00    0.90   0.005 **
## s(lc5)                 9.00  4.64    0.92   0.075 .
## s(lc6)                 9.00  1.01    0.95   0.800
## s(lc7)                 9.00  4.26    0.92   0.040 *
## s(lc8)                 9.00  4.06    0.95   0.720
## s(lc9)                 9.00  5.93    0.91   0.020 *
## s(lc10)                9.00  5.61    0.90  <2e-16 ***
## s(lc11)                9.00  5.04    0.90  <2e-16 ***
## s(lc12)                9.00  4.44    0.90  <2e-16 ***
## s(lc13)                9.00  5.96    0.90  <2e-16 ***
## s(lc14)                9.00  1.86    0.91   0.015 *
## s(lc15)                9.00  6.58    0.92   0.035 *
## s(lc16)                9.00  5.06    0.88  <2e-16 ***
## s(lc17)                9.00  3.40    0.92   0.075 .
## s(lc18)                9.00  4.27    0.94   0.355
## s(altiMean)            9.00  8.59    0.95   0.810
## s(altiSD)              9.00  7.84    0.96   0.840
## s(relative_humidity)   9.00  8.26    0.93   0.175
## s(wind)                9.00  8.77    0.94   0.275
## s(clim3)               9.00  8.91    0.93   0.160
## s(clim4)               9.00  8.08    0.94   0.350
## s(clim5)               9.00  8.51    0.94   0.275
## s(clim6)               9.00  8.92    0.95   0.760
## s(clim7)               9.00  8.73    0.95   0.560
## s(clim8)               9.00  8.66    0.93   0.090 .
## s(clim9)               9.00  7.53    0.93   0.110
## s(clim10)              9.00  8.61    0.95   0.765
## s(mean_monthly_fires)  9.00  9.00    0.90  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

For the basis dimension checking results (table), land cover smooths have low (between 1 and 8) estimated degree of freedom (`edf`), thus even if the p-values are low there is no need to put a higher value for `k`. Climate smooths have higher `edf`, but p-values suggests that there is no need to change `k` values. For `mean_monthly_fires` smooth, the `edf` equals `k'` and p-value is really low which signifies that we should augment `k` value.
It is clear that almost all of land cover covariates have a small degree of freedom and also some seems to be straight lines, e.g, `lc4`, `lc6` and `lc14`, due to the `edf` which is almost 1. We have the `mean_monthly_fires` covariate that attains the maximum degree of freedom (= 9), this suggests that we should set `k` to a higher value. For the other covariates the `edf` is almost the maximum, so

we should also try to augment `k`.
Lets take a look at each smooth to see if some covariates could be taken out of our model and what impact it has on our counts.
Looking at the plot for 2D smooth for `lon` and `lat` covariates [Figure 15], we can see that in the middle west part of the USA these covariates have a negative effect on counts inversely to the north-east. We could try to add more degrees of freedom for this smooth as suggested by `gam.check()` table.

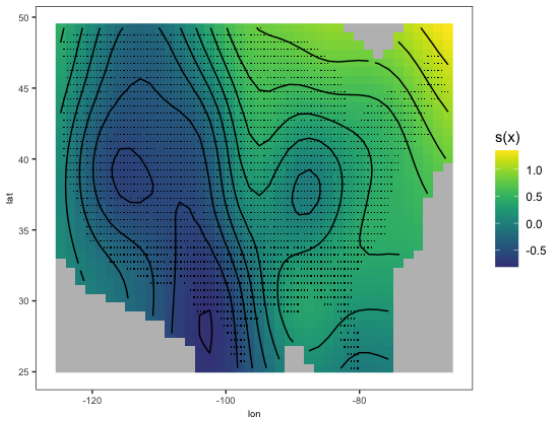

Figure 15: Smooth for `lon` and `lat`

Next, as our `gam.check()` table showed, we can see that the smooths for land cover covariates [Figures 16-17] are not wiggly, except around zero, e.g. `lc3` and `lc10`. Some of these smooths seems to be straight line, e.g. `lc4`, `lc6` and `lc14`, this suggests that there is no need to use smooths for these but just to use it in the model as linear coefficients. This corroborate the results from the `gam.check()` table. For some, there is a clear decreasing effect on counts, e.g. `lc10` and `lc17`. There are some that are almost zero, which suggests that there is no impact between counts and these covariates, e.g. `lc2`, `lc11` and `lc14`.

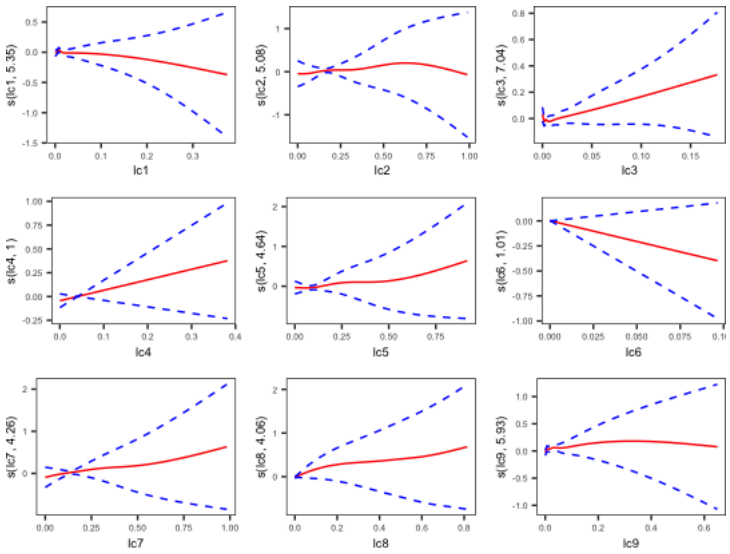

Figure 16: Smooths for `lc1` to `lc9`

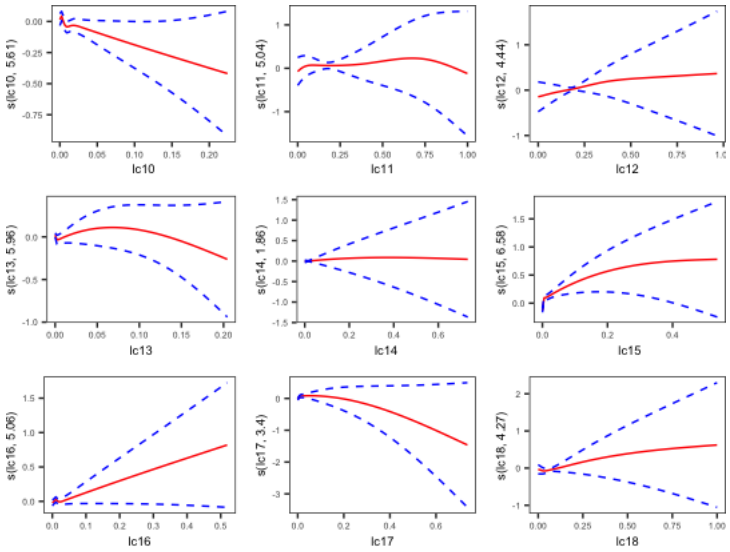

Figure 17: Smooths for `lc10` to `lc18`

Plotting smooths for climate covariates [Figure 18], except `relative_humidity` [Figure 19], there is a lot more wiggliness for those than for land covers ones. `wind` seems to have almost no effect on counts due to being almost zero and does not need a smooth. Observing the scale of the vertical axis, we obtain that some of these covariates have an heavier effect than land covers since the values go from -2 to 2, e.g. `clim4`, `clim7` and `clim8`. It follows our first intuitions that these climate covariates have more importance than land covers for counts of fires. We will try for some to increase the degrees of freedom.

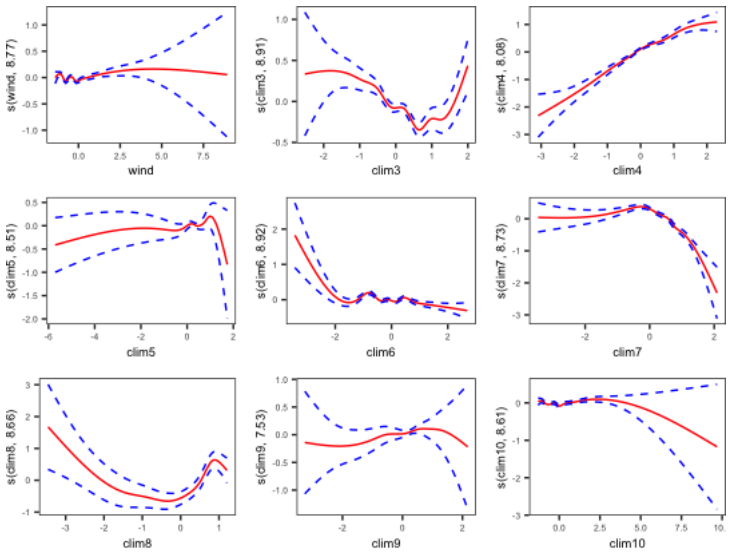

Figure 18: Smooths for climate covariates (`wind`, `clim3` to `clim10`)

Looking at plots of smooths for the remaining covariates (`area`, altitude, `relative_humidity` and `mean_monthly_fires`) [Figure 19], `altiSD` seems to have no impact on counts since it is almost zero, then we can use no smooth or just not use it in our model. Surprisingly, looking at vertical axis of `relative_humidity` plot, we observe that the values go from almost -10000 to 0 which is far bigger than what we saw for other covariates. This may be an explanation to why it is a good covariate to add to our data set, or this is a problem of modelling with this covariate. For the last covariate but not the least important, there is an increasing impact of `mean_monthly_fires` on counts which makes sense.

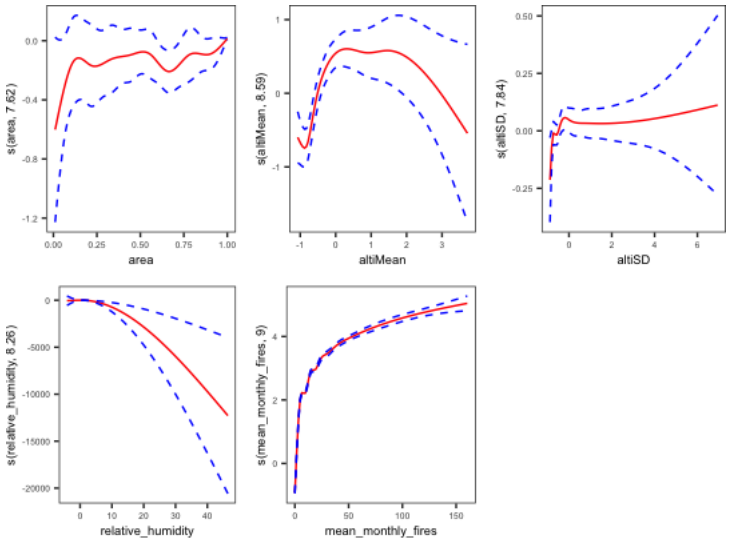

Figure 19: Smooths for `area`, altitude (`altiMean` and `altiSD`), `relative_humidity` and `mean_monthly_fires` covariates

To predict the mean of the Poisson process :

```
predict(gam1CNT, test, type="response")
```

Using these predictions, we obtained a scaled score of 0.05411833. This is a good improvement on the Zero-inflated Poisson GLM.

Lastly, after making modifications to the previous model, we obtain :

```
gam2CNT<- bam(CNT ~ s(lon,lat,k=60) + s(area,bs="cr",k=3) + month
              + s(lc1,bs="cr",k=4) + s(lc2,bs="cr",k=4) + s(lc3,bs="cr",k=4)
              + s(lc4,bs="cr",k=2) + s(lc5,bs="cr",k=4) + s(lc6,bs="cr",k=3)
              + s(lc7,bs="cr",k=3) + s(lc8,bs="cr",k=4) + s(lc9,bs="cr",k=4)
              + s(lc10,bs="cr",k=4) + s(lc11,bs="cr",k=3) + s(lc12,bs="cr",k=3)
              + s(lc13,bs="cr",k=4) + s(lc14,bs="cr",k=3) + s(lc15,bs="cr",k=4)
              + s(lc16,bs="cr",k=3) + s(lc17,bs="cr",k=4) + s(lc18,bs="cr",k=3)
              + s(altiMean,bs="cr",k=20) + s(altiSD,bs="cr",k=4)
              + s(relative_humidity,bs="cr",k=4) + s(wind,bs="cr",k=3)
              + s(clim3,bs="cr",k=20) + s(clim4,bs="cr",k=6)
              + s(clim5,bs="cr",k=6) + s(clim6,bs="cr",k=6)
              + s(clim7,bs="cr",k=6) + s(clim8,bs="cr",k=6) + s(clim9,bs="cr",k=4)
              + s(clim10,bs="cr",k=6) + s(mean_monthly_fires,bs="cr",k=20),
              data = train_data, family = poisson(link="log"))
```

As before, we fit a Poisson model with a log link function. We changed the `k` parameter as said previously in a way to improve our model, prediction, score and computing time. Recalling our `gam.check()` function, we obtain the following plots and basis dimension checking results.

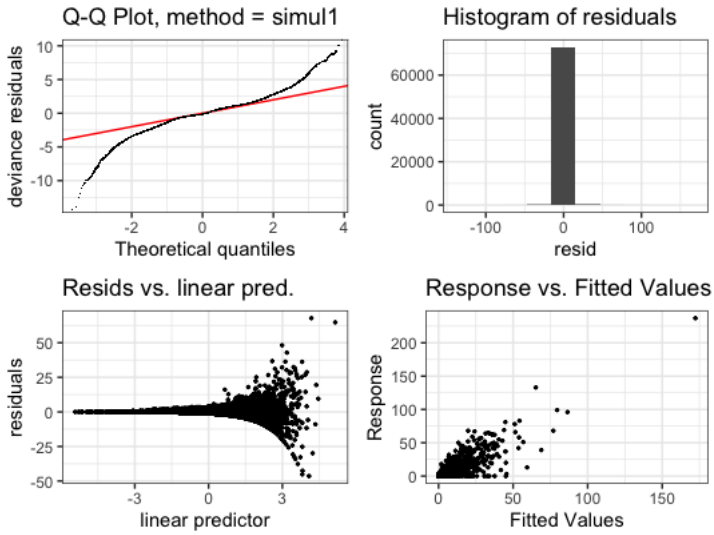

Figure 20: Plots of `gam.check(gam2CNT)`

The Q-Q plot []Figure 20 is similar as the one of our precedent model, it is expected since we did not change the distribution. Residuals vs linear predictor plot is as before, it is what we expect for a Poisson distribution but we can still assess of the overdispersion. Histogram of residuals and response vs fitted values are still quite the same as previously, but we remark that points have regrouped at the bottom left of the latter plot.

```
##
## Method: fREML   Optimizer: perf newton
## full convergence after 14 iterations.
## Gradient range [-0.0003487985,2.171805e-05]
## (score 126431.4 & scale 1).
## eigenvalue range [-2.131696e-05,21.63345].
## Model rank =  207 / 207
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##                          k'   edf k-index p-value
## s(lon,lat)            59.00 54.25    0.96   0.225
## s(area)                2.00  1.00    0.97   0.475
## s(lc1)                 3.00  1.91    0.91  <2e-16 ***
## s(lc2)                 3.00  2.51    0.91  <2e-16 ***
## s(lc3)                 3.00  1.00    0.93   0.005 **
## s(lc4)                 2.00  1.00    0.95   0.130
## s(lc5)                 3.00  2.11    0.92  <2e-16 ***
## s(lc6)                 2.00  1.00    0.96   0.435
## s(lc7)                 2.00  1.59    0.93  <2e-16 ***
## s(lc8)                 3.00  2.73    0.97   0.600
## s(lc9)                 3.00  2.87    0.93   0.010 **
## s(lc10)                3.00  1.96    0.91  <2e-16 ***
## s(lc11)                2.00  1.97    0.94   0.040 *
## s(lc12)                2.00  1.98    0.92   0.005 **
## s(lc13)                3.00  1.96    0.96   0.200
## s(lc14)                2.00  1.64    0.94   0.040 *
## s(lc15)                3.00  2.99    0.94   0.085 .
## s(lc16)                2.00  1.18    0.93  <2e-16 ***
## s(lc17)                3.00  1.96    0.93  <2e-16 ***
## s(lc18)                2.00  1.36    0.94   0.020 *
## s(altiMean)           19.00 14.55    0.97   0.455
## s(altiSD)              3.00  2.92    0.97   0.705
## s(relative_humidity)   3.00  2.77    0.96   0.270
## s(wind)                2.00  1.99    0.99   0.975
## s(clim3)              19.00 18.14    0.95   0.190
## s(clim4)               5.00  4.89    0.96   0.385
## s(clim5)               5.00  4.26    0.99   0.980
```

```
## s(clim6)                5.00  4.89    0.96   0.480
## s(clim7)                5.00  4.93    0.96   0.290
## s(clim8)                5.00  4.97    0.95   0.205
## s(clim9)                3.00  2.96    0.96   0.240
## s(clim10)               5.00  4.68    0.97   0.715
## s(mean_monthly_fires) 19.00 18.94    0.92  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We now have a model with lower value of freedom which should be better for predictions since it would over fit less our train data. We set `k` for land covers covariates in a way that almost all smooths are straight line (as in a linear model), it is confirmed by our `gam.check()` table. The only modification that would be useful is still to allowed more degree of freedom for `mean_monthly_fires` smooth.
The smooth for `lon` and `lat` [Figure 21] is quite similar as before [Figure 15], except that due to the larger degree of freedom it varies a little bit more but not enough to be significant.

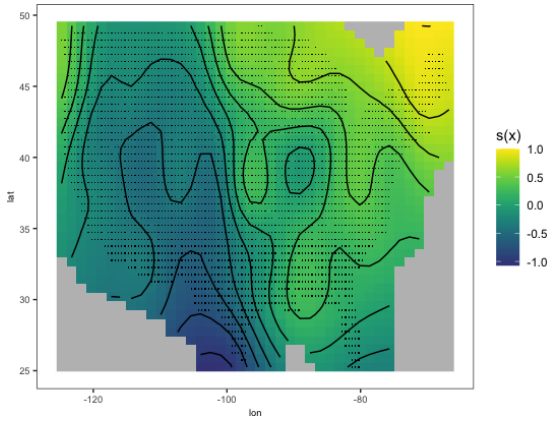

Figure 21: Smooth for `lon` and `lat`

Land cover smooths are now significantly different [Figure 22-23] than our previous model [Figures 16-17] due to the lower degree of freedom for each smooth. We can now better determine which land cover covariates have almost no influence on counts, there are straight lines on zero, e.g. `lc6`, `lc11` and `lc14`.

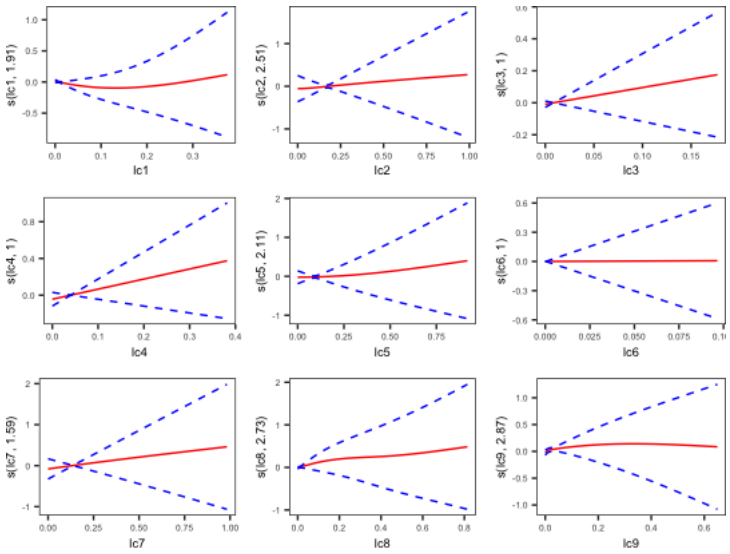

Figure 22: Smooths for `lc1` to `lc9`

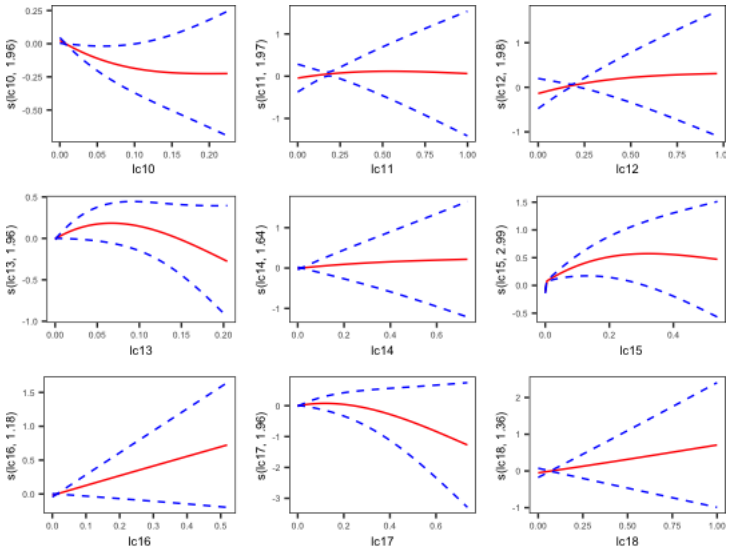

Figure 23: Smooths for `lc10` to `lc18`

Smooths for `clim4`, `clim7`, `clim8` and `clim10` [Figure 24] did not change from our previous model [Figure 18], but `wind` seems to be a lot more important than before. `clim3` smooth is almost zero except for lower values which is due to the higher degree of freedom (`k = 20`).

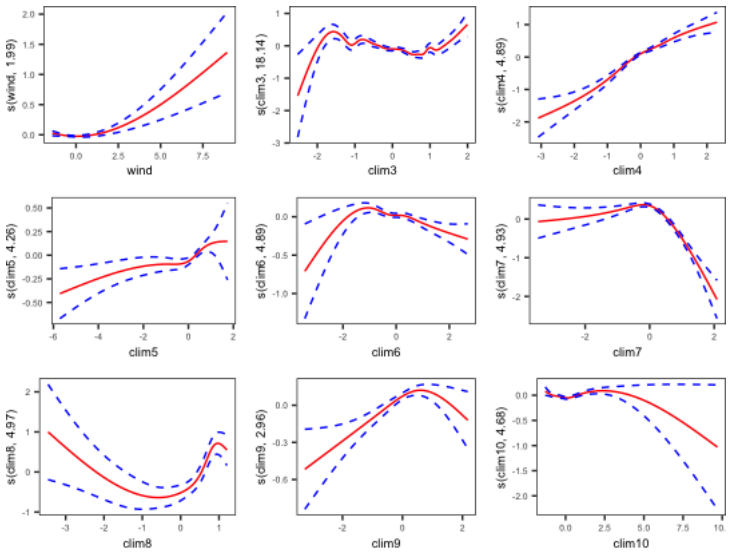

Figure 24: Smooths for climate covariates (`wind`, `clim3` to `clim10`)

Looking at our remaining smooths [Figure 25], there is almost no change, each one is like the one of our previous model, except for `relative_humidity`. For the latter one, we see that our scale on vertical axis is far smaller than before which makes more sense. One of the most important covariates is still `mean_monthly_fires`.

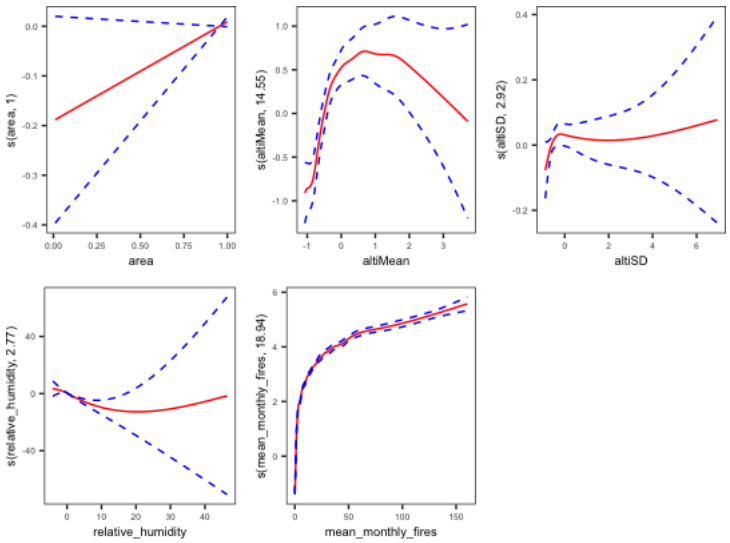

Figure 25: Smooths for `area`, altitude (`altiMean` and `altiSD`), `relative_humidity` and `mean_monthly_fires` covariates

To predict the mean of the Poisson process :

```
predict(gam2CNT, test, type="response")
```

Needing to make observations of our predictions, we plot the absolute error against counts and a boxplot of this error [Figure 26] with intervals on the horizontal axis which are determined by thresholds of the challenge. High value on the right of each plots means that there are bad predictions for extreme values. First plot suggests that there is a lot of error for zeros but looking at the second one we see that the box is flat, which means the opposite. Going form left to right in the second plot, we see that almost each box goes upper than the previous one, i.e. bad predictions for extremes. This implies that our score is affected since higher counts have higher weights.

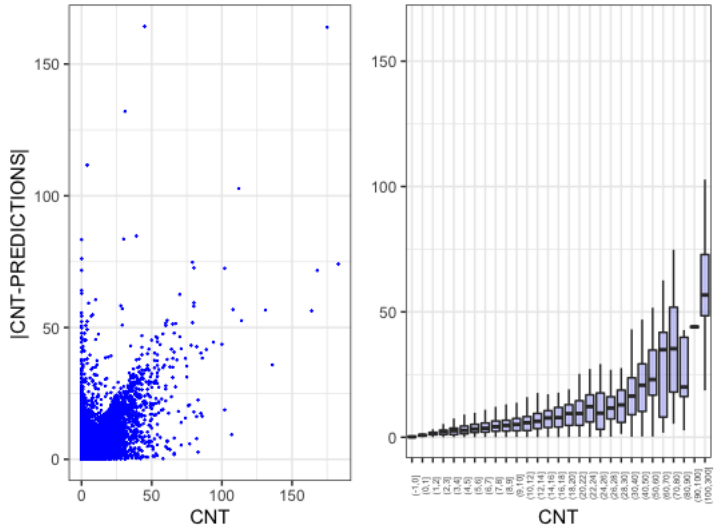

Figure 26: Absolute error of predictions

Using these predictions, we obtained a scaled score of 0.05300531. This is a small improvement on our previous model for counts.
Other models were implemented to improve our score, but those models were too low to compute or the improvement was not consequent enough to this last one.

4.2.3 GAMs for BA

For burnt areas, we will present only the best GAM fitted since the method is similar as in section 4.2.2. We fit the following model :

```
gam1BA <- bam(log(BA+1) ~ s(lon,lat,k=60) + s(area,bs="cr",k=3) + month
              + s(lc1,bs="cr",k=4) + s(lc2,bs="cr",k=4) + s(lc3,bs="cr",k=4)
              + s(lc4,bs="cr",k=2) + s(lc5,bs="cr",k=4) + s(lc6,bs="cr",k=3)
              + s(lc7,bs="cr",k=3) + s(lc8,bs="cr",k=4) + s(lc9,bs="cr",k=4)
              + s(lc10,bs="cr",k=4) + s(lc11,bs="cr",k=3) + s(lc12,bs="cr",k=3)
              + s(lc13,bs="cr",k=4) + s(lc14,bs="cr",k=3) + s(lc15,bs="cr",k=4)
              + s(lc16,bs="cr",k=3) + s(lc17,bs="cr",k=4) + s(lc18,bs="cr",k=3)
              + s(altiMean,bs="cr",k=6) + s(altiSD,bs="cr",k=4)
              + s(relative_humidity,bs="cr",k=4) + s(wind,bs="cr",k=3)
              + s(clim3,bs="cr",k=6) + s(clim4,bs="cr",k=6) + s(clim5,bs="cr",k=6)
```

```
              + s(clim6,bs="cr",k=6) + s(clim7,bs="cr",k=6) + s(clim8,bs="cr",k=6)
              + s(clim9,bs="cr",k=4) + s(clim10,bs="cr",k=6)
              + s(mean_monthly_fires,bs="cr",k=5),
              data = train_data, family = gaussian)
```

We made a Gaussian model with log link function and all covariates that we could used, except year, and using cubic regression splines as basis for each smooth, except for `lat` and `lon`. Calling `gam.check()` function, we obtain the following plots and basis dimension checking results.

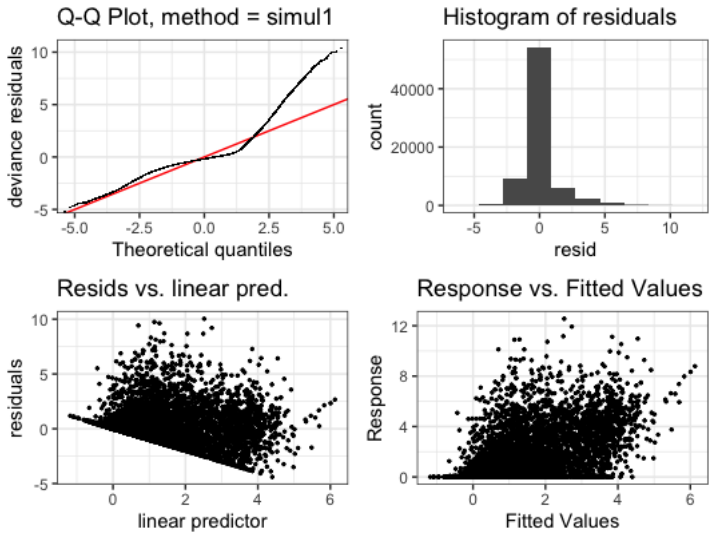

Figure 27: Plots of `gam.check(gam1BA)`

The Q-Q plot [Figure 27] is heavy tailed for high quantile values but not for low values. Residuals seems normally distributed as we can see on the histogram. Looking at the response vs fitted values plot, there is a different scale for each axis and the cloud seems to be flattened to the right which could be due to the different scaling.

```
##
## Method: fREML   Optimizer: perf newton
## full convergence after 15 iterations.
## Gradient range [-0.000155196,4.191587e-07]
## (score 123795.9 & scale 1.684813).
## Hessian positive definite, eigenvalue range [1.669199e-05,36763.52].
## Model rank =  164 / 164
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##                          k'   edf k-index p-value
## s(lon,lat)            59.00 55.31    0.94  <2e-16 ***
## s(area)                2.00  1.00    1.02   0.880
```

```
## s(lc1)                 3.00  1.21    0.98   0.045 *
## s(lc2)                 3.00  1.00    0.97   0.045 *
## s(lc3)                 3.00  1.65    0.98   0.115
## s(lc4)                 2.00  1.76    0.96  <2e-16 ***
## s(lc5)                 3.00  2.22    0.98   0.150
## s(lc6)                 2.00  1.53    1.02   0.855
## s(lc7)                 2.00  1.97    0.96   0.010 **
## s(lc8)                 3.00  2.49    1.01   0.725
## s(lc9)                 3.00  1.00    0.98   0.165
## s(lc10)                3.00  1.72    0.99   0.290
## s(lc11)                2.00  1.96    0.95  <2e-16 ***
## s(lc12)                2.00  1.98    0.96   0.005 **
## s(lc13)                3.00  1.83    0.99   0.205
## s(lc14)                2.00  1.74    0.98   0.090 .
## s(lc15)                3.00  1.00    0.99   0.190
## s(lc16)                2.00  1.00    0.97   0.055 .
## s(lc17)                3.00  1.53    1.00   0.565
## s(lc18)                2.00  1.95    0.99   0.260
## s(altiMean)            5.00  4.59    0.96  <2e-16 ***
## s(altiSD)              3.00  1.00    0.98   0.065 .
## s(relative_humidity)   3.00  2.11    1.00   0.630
## s(wind)                2.00  1.92    0.99   0.120
## s(clim3)               5.00  4.51    1.00   0.590
## s(clim4)               5.00  4.73    1.00   0.610
## s(clim5)               5.00  4.00    1.02   0.910
## s(clim6)               5.00  4.50    1.01   0.680
## s(clim7)               5.00  4.42    1.00   0.355
## s(clim8)               5.00  4.01    0.97   0.035 *
## s(clim9)               3.00  2.97    1.00   0.425
## s(clim10)              5.00  4.92    1.00   0.660
## s(mean_monthly_fires)  4.00  4.00    0.99   0.195
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

A lot of our land cover covariates have degrees of freedom between 1 and 2, which means that there are almost straight lines. This means that they could be passed as in a linear model, i.e. just some linear parameters. `k`-index are almost at one which means that there is no need to augment `k` values. The interesting covariate is `mean_monthly_fires` since `k'` is equal to `edf`, but the p-value is not so small. Thus how to know if it would be better to take a higher value of `k` or not ? We will take a look at this smooth in order to answer to this question.
Looking at the plot for 2D smooth for `lon` and `lat` covariates [Figure 28], for the west part of the united states the smooth is like the one for `CNT` [Figure 21] but not for the east part where high values are in the south-east instead of north-east.

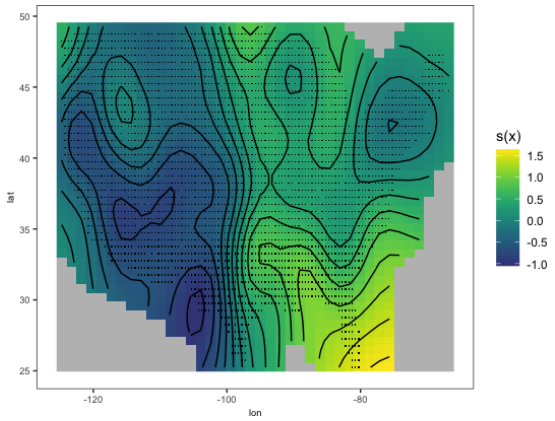

Figure 28: Smooth for `lon` and `lat`

For plots of land cover smooths [Figures 29-30], almost all land covers have decreasing effect on burnt areas since they all are straight lines, except for `lc13`. All seems to be non-zero which means they had to be part of our model.

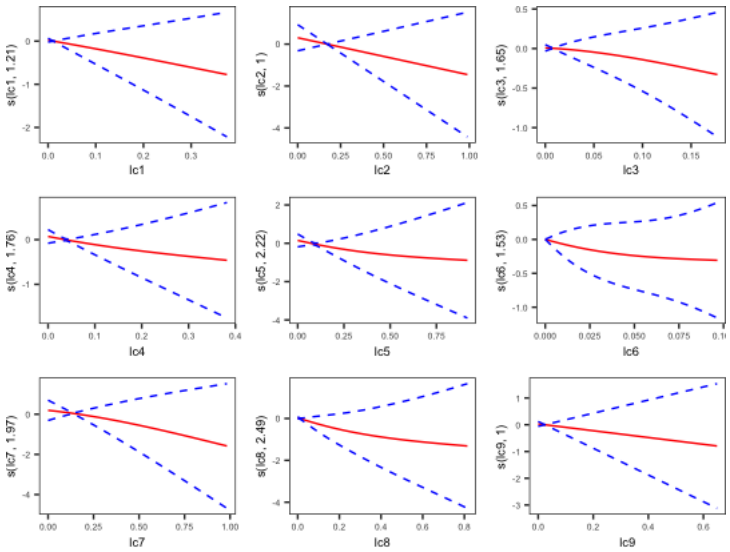

Figure 29: Smooths for `lc1` to `lc9`

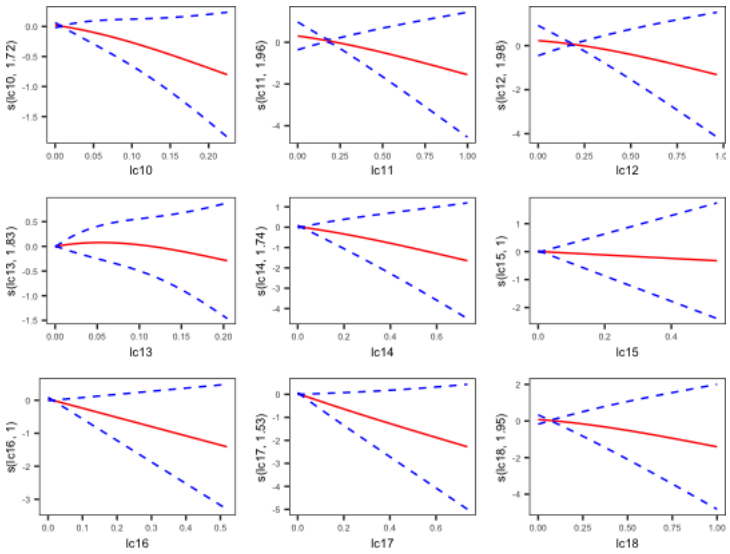

Figure 30: Smooths for `lc10` to `lc18`

Our climate smooths [Figure 31] are more wiggly than land cover ones. `clim10` seems of no importance since it is almost zero for every value. Their shapes correspond to the `k` value that we fixed in our model. Most important ones seems to be `clim3`, `clim4` and `clim8` due to the scale of their response values.

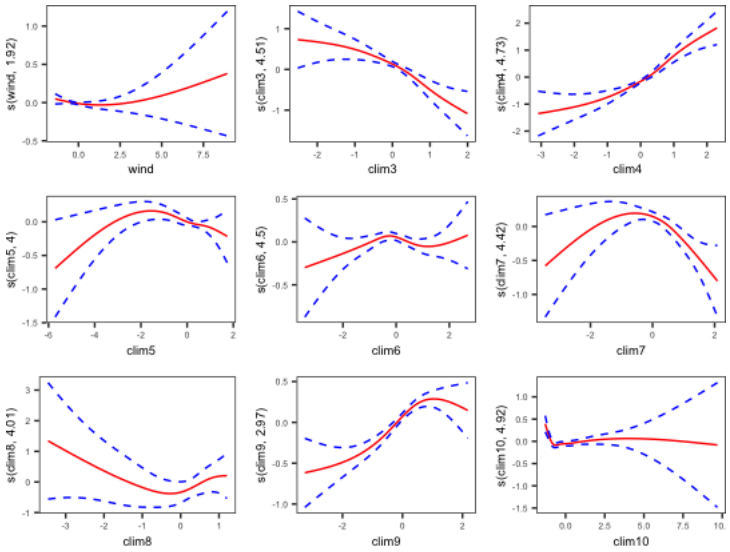

Figure 31: Smooths for climate covariates (`wind`, `clim3` to `clim10`)

Looking at our last smooths [Figure 32], we can answer to our previous question of the low value of `mean_monthly_fires` covariate which have shape of a degree four polynomial, but we observe that the oscillations are not significant. This means that their is no need to put a high value for the `k` parameter for this smooth, `k` = 3 would be sufficient in order to have the shape of a degree two polynomial. `relative_humidity` seems to be of importance due to the high value of its response

which is similar to `mean_monthly_fires` covariate.

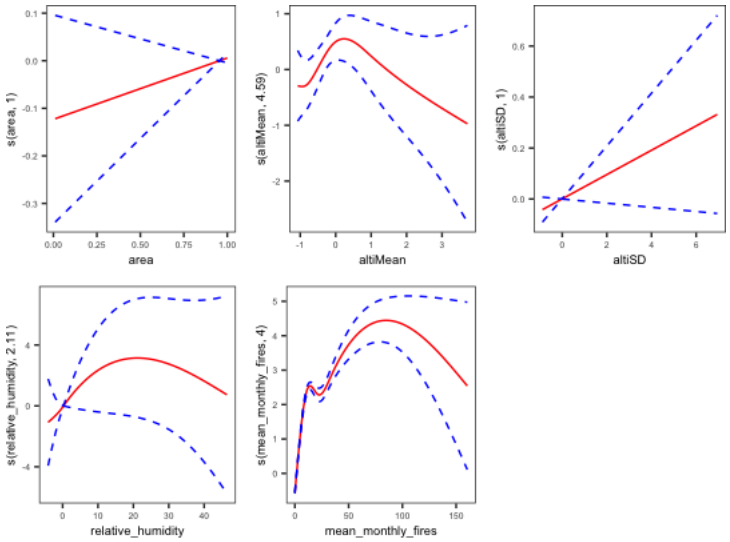

Figure 32: Smooths for `area`, altitude (`altiMean` and `altiSD`), `relative_humidity` and `mean_monthly_fires` covariates

To predict the response of our model for our test set :

```
predict(gam1BA, test, type="response")
```

Now we plot the absolute error of our predictions against `BA` [Figure 33] which is log scaled. On the left plot, higher the burnt area is, higher the error is, which means that our model is bad for extreme values which is confirmed by the right boxes of the right plot. For lower values of `BA`, it is better to look at boxplots since there is too much density of points in the first one to interpret it. There is still errors for lower values, which can be corrected in predictions when counts are known.

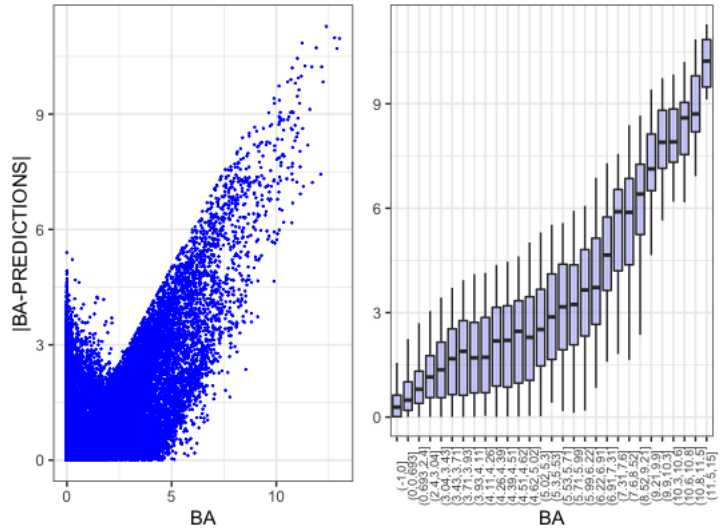

Figure 33: Absolute error of predictions

Using these predictions, we obtained a scaled score of 0.04247587. This was the best score attained with GAMs models.

4.3 Conclusion on GAMs

For GAMs, we used cubic splines for every model due to computation times which are far less than thin plate smoothing splines (default smoothing basis) of `bam` function, except for 2D smooths because cubic splines are for 1D smooths. The use of all covariates was a choice made since if a covariate is of no importance for our model, then it smooths would be equal to zero for every value of the covariate. The objective was not to choose which covariates to include into our model but to optimize our predictions.

GAMs are good for burnt areas prediction but not as much for counts, we will see in the next section another method that will really improve our count predictions.

5 Decision Trees, Random Forests and Boosting

5.1 Decision Trees

The idea behind decision trees is to partition the feature space into a set of rectangles and to fit a constant on each rectangle. The response is then predicted as:

$$f(x_1, \cdots, x_p) = \sum_{m=1}^{M} c_m \mathbb{1}\{(x_1, \cdots, x_p) \in R_m\},$$

where $R_1, \cdots, R_M$ is a partition of the feature space and p is the dimension of the feature space.

The Classification and Regression Trees (CART) algorithm needs to decide on which variables to split and at which threshold. Since our goal is to best fit the training data, the problem is reduced to minimizing a loss function. We will introduce this problem in the regression and classification contexts.

5.1.1 Regression Trees

In the regression case, we want to minimize the residual sum of squares. That is, having decided on a number of final leaves M, we want to solve:

$$\min_{R_1,\cdots,R_M} \min_{c_1,\cdots,c_M} \sum_{m=1}^{M} \sum_{\vec{x_i} \in R_m} (y_i - c_m)^2$$

.

It is clear that the solution to the inner minimization problems is given for each $m = 1, \ldots, M$ by $c_m = \frac{1}{|R_m|} \sum_{x_i \in R_m} y_i$. As for choosing the splitting variables and splitting value, the CART algorithm will compute for each of the p explanatory variables the splitting value at which the residual sum of squares is minimized, and choose that variable as a splitting variable.

Choosing the right tree size M is in itself a complex problem as we want to build a tree large enough to explain the data without over-fitting it. This is done by specifying a maximum number of final leaves M and building a full tree T. Then we would like to find a sub-tree $T_0 \subset T$, obtained by collapsing branches of T, that minimizes the cost complexity criterion:

$$C_\alpha(T) = \sum_{m=1}^{|T|} \sum_{\vec{x_i} \in R_m} (y_i - c_m)^2 + \alpha \mid T \mid$$

Thus α is a tuning parameter that determines the trade-off between the complexity of the tree and how well it fits the training data. If it is set to 0, there is no penalty for complexity and CART algorithm returns a full tree. And the higher the α value, the simpler the resulting tree will be. Finally, another way to avoid over-fitting the data is to set `minsplit`, the minimum number of observations that must exist in a node to allow for further splits, and `minbucket`, the minimum number of observations in any terminal leave.

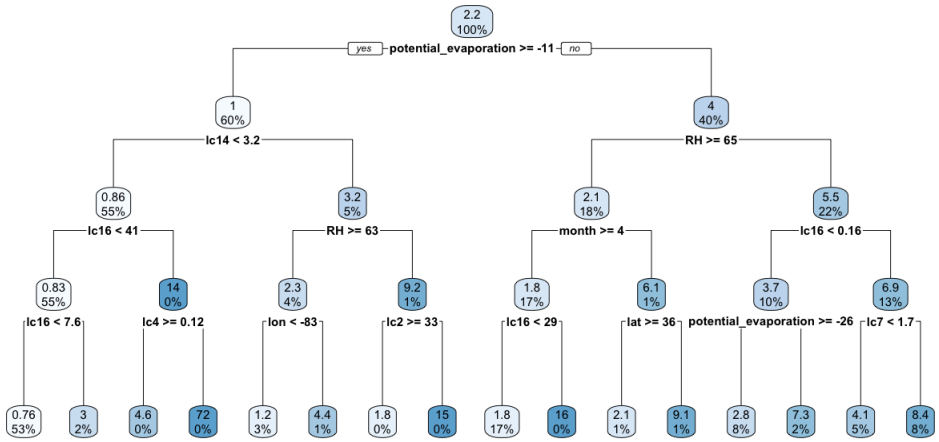

Figure 34: Regression tree for CNT trained on all odd years with maximum depth 4 and $\alpha = 0$

The regression tree above seems more pertinent than the results we got using generalized linear models and it is also more easy to interpret. Looking at the splitting variables and values, we recognize some expected effects, such as the fact that more `lc16` (urban) land cover leads to higher CNT or that higher `RH` relative humidity leads to lower CNT for example. For that particular regression tree, tested on a random sub-sample of size 80 000 taken across non-masked data on even years, we got a scaled score of 0.06809. Comparatively, the benchmark model described in Section 2 got scaled score of 0.06760 on the same training and testing sets.

It is important to notice that the above regression tree only predicted 16 different CNT values spread across the interval $[0.76, 72]$. Whereas the training data contained 157 different CNT values ranging from 0 to 275. This problem could be solved by taking deeper trees. A depth of at least 7 seems more fitting as it would result in at least 128 nodes.

The following table summarizes the different scaled scores we got for different tree depths, using the same training and testing data as described above.

Maximum Depth	4	10	15	20	25
Scaled Score	0.06809	0.06201	0.05979	0.05983	0.05988

Table 1: Scaled scores for decision trees of different depth trained on odd years, with $\alpha = 0$

5.1.2 Classification Trees

In the case of classification trees, the goal is to predict a class amongst $1, \ldots, K$. Classification trees are also built using the CART algorithm described above in the case of regression trees, except that the sum of square is replaced by a more appropriate loss function. An example of loss function would be the miss-classification error $\frac{1}{|R_m|} \sum_{\vec{x_i} \in R_m} \mathbb{1}\{y_i \neq k\}$.

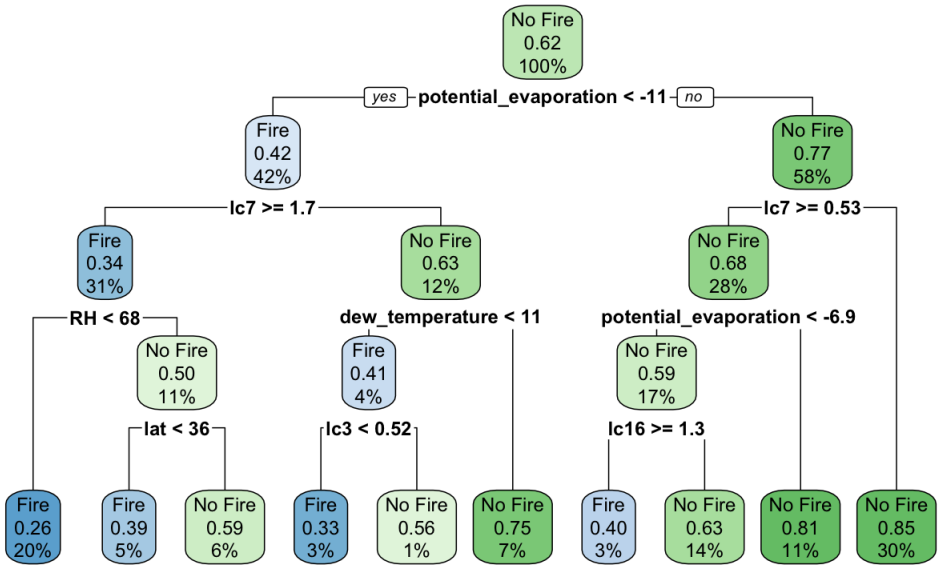

Figure 35: Classification tree for fire occurrence trained on all odd years with maximum depth 4 and $\alpha = 0$

The Figure above is a classification tree predicting the probability that there is no fire occurrence. The outcome is predicted as `"No Fire` for a probability above 0.5 and as `"Fire"` otherwise.

Decision trees seem to be a good first approach, they are particularly appropriate for fire weather data in modelling the conditions necessary for their occurrence. However, they are very unstable and sensitive to the training data. Small changes can result in entirely different tree structures. They also lack the smoothness of GLM and GAM models in predicting the data, which can be partially remedied by using deeper trees. It was necessary to understand decision trees as they are the building blocks for much more robust methods such as random forests and boosting, which are introduced next.

5.2 Random Forests

The motivation behind random forests is to combine many decision trees to produce a more powerful model. A random forest builds several trees and averages the predictions made by each of its decision trees to produce a final prediction. In the case of classification, it predicts the class

that got the most "votes". The decision trees are built independently of each other in the following way:

1. Each tree is build on a different subsample of the same size, taken randomly from the training data.
2. Each time a split is attempted in a tree, only $m < p$ out of the p explanatory variables are considered as candidates for the split.

Considering only a subset of all explanatory variables reduces correlation between decision trees and thus makes random forests more robust. Indeed, if all variables were considered at every split, most of the trees would choose amongst the same subset of more "powerful" features. But by considering only m explanatory variables as splitting candidates at every split and by using a large number of trees, most features will eventually end up being used by the random forest.

We implemented random forests using the R package `'randomForest'`. We set `mtry`, the number of variables considered at each split to be 6. And got the following scaled scores for different values of `ntrees` the number of decision trees in the random forest.

`ntrees`	100	200	300	500
Scaled Score	0.06809	0.06201	0.05979	0.05567

Table 2: Scaled scores for random forests of different size trained on all odd years

The scaled scores were of same order as decision trees for low numbers of decision trees. The difference became notable for 500 trees where we got a scaled score of 0.05567 compared to 0.05979, the best score obtained using decision trees. We did not attempt to use even more trees as it was computationally too expensive, but it is expected that the score would have improved, up to a certain point where it would start overfitting.

5.3 Boosting

Similarly to random forests, boosting also combines many decision trees. However, it is done differently: instead of building `ntrees` trees independently and in parallel, they are built one after the other in the goal of minimizing a loss function $L(y, \hat{y})$. The model is formulated as:

$$f(\vec{x}) = \sum_{k=1}^{K} T(\vec{x_i}, \Theta_k),$$

where K is the number of trees in the boosted model, M is the depth of all trees and for all $k = 1, \ldots, K$, $\Theta_k = \{R_{km}, c_{km}\}_{m=1}^{M}$ are the parameters of tree k.

The boosted model $f(\vec{x})$ is built iteratively using gradient boosting. It is initialized as $f(x) = \min_{\hat{\Theta}} T(\vec{x}, \hat{\Theta})$. Then given the model made up of $K-1$ trees $f_{K-1}(\vec{x})$, a K-th tree $T(\vec{x}, \Theta_K)$ is added to the model such that the loss function $\frac{1}{N}\sum_{i=1}^{N} L(y_i, f_{K-1}(\vec{x_i}) + T(\vec{x_i}, \Theta_K))$ is minimized over Θ_K.

We implement boosting to model CNT using the `'gbm'` package in R. We optimized the model by tuning the hyper-parameters `ntrees`, number of trees in the model or number of iterations and `interaction.depth`, the size of each individual tree.We specified the distribution to be `'poisson'`, the loss function associated by default was the poisson loss:

$$L(y, \hat{y}) = \hat{y} - ylog(\hat{y})$$

.

We decided to set `ntrees` to 10,000 and the tree depth `interaction.depth` to 6. The scaled score did get better by using even more trees or deeper trees, however, the difference was only of order 10^{-4} which was not significant enough to demand the extra computing time. The Table below shows the order of improvement for using different `interaction.depth` values. As for the learning rate, we kept the one chosen by default by R, that is 0.1.

`interaction.depth`	4	6	8	10
Scaled Score	0.04558	0.04481	0.04475	0.04469

Table 3: Scaled scores for GBM models of different `interaction.depth` trained on all odd years

Further fine tuning of the hyper-parameters was judged unnecessary as any other resulting improvements were of order 10^{-4}. We thus decided to turn to feature engineering, more particularly the creation of new features. Our reasoning was based on the Fire Weather Index System [7], an index used to evaluate fire danger worldwide. It is made of three indices:

1. Initial Spread Index: Indicator of fire ignition danger
2. Build-Up Index: Indicator of the quantity and state of fuel available for burning
3. Fire Weather Index: Indicator of extreme fire weather

We already added the variables `RH`, the relative humidity, as well as `mean_monthly_fires`, the mean number of fires over the years in each voxel in each month. We considered adding two other covariates:

1. `max_landcover`: A categorical variable giving the type of land cover that is the most prominent in a voxel
2. `previous_fires`: The number of fires that occurred in a voxel the previous month. Indeed, prescribed fires is well known fire management technique. It reduces the fuel available for burning and thus the risk of future wildfires. However, a difficulty in using that variable is that the dataset contained fire occurrences only from March to September for each year. As a solution, we used this model on the months from April to September and used a simpler GBM model on March that did not contain this added variable `previous_fires`.

	No added features	`mean_monthly_fires`	`mean_monthly_fires` and `max_landcover`
Without BA	0.04402	0.04387	0.04401
With BA	0.03479	0.03463	0.03495

Table 4: Scaled score for GMB models using added features, trained on all odd years

In the end, the variable `previous_fires` worsened our scaled scores by a lot so we didn't include it. The Table above shows the score changes we got by using the features `mean_monthly_fires` and `max_landcover`. In the end, only `mean_monthly_fires` improved our model, so it was the only added feature used in our final model.

Finally, boosting is also interesting because it naturally performs feature selection. We had explored reducing our 18 landcover types into categories. For example, we would only differentiate water, urban, bare areas and areas with dense and flammable vegetation. However, because boosting automatically performs feature selection, we did not use any dimensionality reduction techniques.

The following table shows the variables that were predicted to be the most important by our boosting model. Variable importance is calculated as the mean decrease in mean squared error resulting from a split made on that variable over all trees.

We see how meaningful the `mean_monthly_fires` variable and `RH` we added are. We see as well that the landcover judged to be most important is urban landcover `lc16`, coherent with the fact that human activity has a high impact on increasing the number of wildfires.

	var <chr>	**rel.inf** <dbl>
mean_monthly_fires	mean_monthly_fires	61.22076842
year	year	2.99363176
RH	RH	2.80760704
thermal_radiation	thermal_radiation	2.16171472
potential_evaporation	potential_evaporation	2.06600932
lc16	lc16	2.03309339
precipitation	precipitation	1.94119221
Wspeed	Wspeed	1.76402398
evaporation	evaporation	1.57116867
dew_temperature	dew_temperature	1.54493690

Figure 36: Table of variable importance for GBM model with added variable `mean_monthly_fires`

The final model for CNT was a GBM model using 10,000 trees of depth 6, trained on all available data with added feature `mean_monthly_fires`.

6 Extreme Value Theory

The aim of this section is to study the burnt areas (BA) of larger fires, more explicitly, fires with burnt areas greater than a certain threshold u. The general perception is that the small number of large fires account for the most damage, and therefore it is crucial that we examine these cases in order to predict the burnt area with more accuracy. Indeed, several studies state that 1% of the fires account for 90% of total damages by wildfires [12][6]. Moreover, in the Data Challenge, predicting larger fires hold a greater weight when calculating the scores. Thus, it is of importance that extremes of burnt area are taken into account when predicting wildfires.

6.1 Theory

Extreme value theory (EVT) can be applied to study specific cases from a sample that deviate extremely from the mean of the data and thus, it is generally used to study the the tail behavior of distributions. There are two main methods used in extreme value theory: the peaks-over-threshold (POT) method and the block maxima method [1]. In particular, the POT method models the extreme events that exceeds a certain threshold value u. Since we consider burnt area extremes to be those over a set threshold, the POT method will be used to study the tail distribution of our data.

The formulation of the POT method is as follows. Let $X_1, X_2, ..., X_n$ be a sequence of independently and identically distributed random variables with distribution function F. Then, the stochastic behaviour of extreme events can be described by the following conditional probability:

$$P\{X > u + y | X > u\} = \frac{1 - F(u + y)}{F(u)}, \quad y > 0$$

It is given that for a large enough u, $\{X > u + y | X > u\}$ follows the distribution function:

$$H(y) = \begin{cases} 1 - (1 + \frac{\xi y}{\tilde{\sigma}})^{-1/\xi} & \text{if } \xi \neq 0, \\ 1 - exp(-\frac{y}{\tilde{\sigma}}) & \text{otherwise} \end{cases}$$

defined on $\{y : y > 0, 1 + \frac{\xi y}{\tilde{\sigma}} > 0\}$, where ξ is the shape parameter, σ the scale parameter, and $\tilde{\sigma} = \sigma + \xi(u - \mu)$. The distribution defined above is called the generalized Pareto distribution (GPD). In the GPD, the shape parameter ξ is dominant in determining the behaviour of the generalised Pareto distribution. If $\xi = 0$, then the distribution follows an exponential distribution, and if $\xi < 0$, the distribution has an upper bound while if $\xi > 0$, there's no upper limit. Therefore, the bigger the shape parameter, the heavier the distribution.

Once the threshold u is determined, the parameters can be estimated by maximum likelihood method. If $y_1, ..., y_k$ are the k excesses of a threshold u, then the log-likelihood for the generalised Pareto distribution parameters is given by the following:

$$l(\sigma, \xi) = \begin{cases} -k \log(\sigma) - (1 + 1/\xi) \sum_{i=1}^{k} \log(1 + \xi y_i / \sigma), & \text{if } \xi \neq 0 \\ -k \log(\sigma) - \sigma^{-1} \sum_{i=1}^{k} y_i, & \text{if } \xi = 0 \end{cases}$$

6.2 Application to US wildfire BA data

In this section, for the purpose of ensuring sufficient extremes, a total data set of odd years between 1993-2005 was taken from the given US wildfires dataset, where 75% of it were used as the training set and 25% as the test set. This resulted in 220,680 observations for the training set and 73,563 observations for the validation set. In this section, the cases with CNT values and without CNT values will both be considered separately.

Due to the heavy-tailed shape of the distributions of wildfires, it has been suggested that different types of Pareto distribution models (such as the generalized Pareto, tapered Pareto) are suitable for modeling large fires over a threshold [12][3]. Using the GPD, we aim to model the distribution of the excesses over the set threshold u to better predict the probabilities $\mathbb{P}(BA < u_{BA,i})$, where $u_{BA,i} \in \mathcal{U}_{BA}$ is given in the description of the challenge (see section EDA).

Predicting the probabilities $\mathbb{P}(BA < u_{BA,i})$ must be followed by a few steps; the following probabilities must be calculated.

- The probability that the burnt area exceeds the threshold u:

$$p_1 = \mathbb{P}(BA > u)$$

- The conditional probability that the burnt area is smaller than $u+v$ given that it exceeds the threshold:

$$p_2 = \mathbb{P}(BA \leq u + v | BA > u)$$

- The probability that the burnt area is between u and $u+v$. By the law of total probability, this is obtained by multiplying p_1 and p_2:

$$p_3 = \mathbb{P}(u < BA \leq u + v) = \mathbb{P}(BA \leq u + v | BA > u) \cdot \mathbb{P}(BA > u) = p_1 \cdot p_2$$

- The probability that the burnt area is smaller than $u+v$ is then given by:

$$p_4 = \mathbb{P}(BA \leq u + v) = \mathbb{P}(BA < u) + \mathbb{P}(u \leq BA < u + v) = \mathbb{P}(BA < u) + p_3$$

  Note that $\mathbb{P}(BA < u)$ was predicted using the generalized boosted regression model, which performed the best in predicting distributions of BA (see previous sections). Here, v is the exceedance to be modeled.

6.2.1 Selection of the threshold

When treating the extremes, the choice of the threshold amounts to the trade-off between bias and variance; if the threshold is too high, there will be too few exceedances which leads to bias while if it is too low, it is likely to violate the asymptotic behavior resulting in high variance. There are two main graphical methods that aid with the selection of the threshold: the mean residual plot and the parameter stability plot.

The mean residual plot graphs u against the 'mean excess' : $\mathbb{E}[X - u | X > u] = \frac{\sigma_u}{1-\xi}$. The GPD model becomes valid at the threshold u if the plot is linear above u. The following figures (figure 37)is the mean residual plot of the US wildfire data (odd years) generated with function **mrlplot()** in the package `'evmix'`. It plots the mean excess across the threshold as well as the 95% confidence interval that allows the consideration of the effects of estimation uncertainty. It

also estimates the parameters at the 90th quantile, which in this case, is $u = 32$, with parameters $\hat{\sigma} = 79$ and $\hat{\xi} = 1.4$. In the plot, it seems appropriate to choose a threshold in the broad range of 15,000-50.000. However, it is difficult to choose one single threshold from this graph. Like this, it is not always simple to interpret the mean residual life plots [1]. An alternative for determining the threshold is looking that the threshold stability plot.

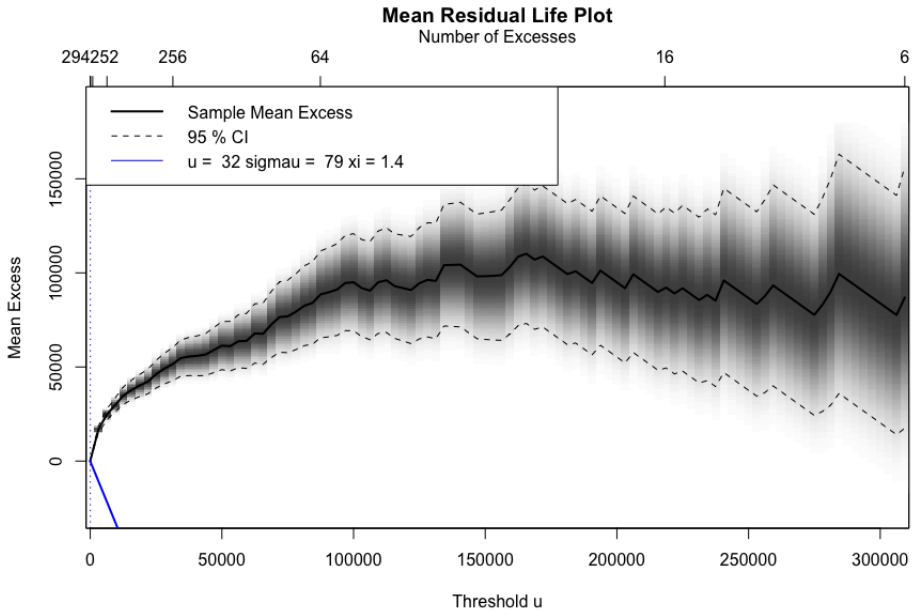

Figure 37: Mean residual life plot (for odd years data)

The threshold stability plot graphs the maximum likelihood estimates of the GPD shape and modified scale (subtracting of the shape times the threshold) parameters against possible threshold values. If GPD model is appropriate for excess above a threshold u, then it should also be suitable for thresholds higher than u. Therefore, above a suitable u, the parameters should be constant. Figure 38 shows the threshold stability plot of wildfire data for thresholds between 0 and 10,000. It was plotted using the function **tcplot()** in the package `'evmix'`.

As the threshold changes, the estimates of the shape and modified scale parameter progress almost oppositely to each other. The shape parameter increases over the range of thresholds, and the modified scale decreases. Again, it is difficult to pinpoint the threshold at which the parameters stabilizes just with the plots.

Another common practice when it comes to selecting the threshold of extremes is using the quantiles of the data. The 99th quantile of the burnt area was 1513.42, thus the threshold of 2,000 acres was selected, which amounts to 4,427 excesses in the odd years dataset. In the following parts, modeling will be performed using this threshold.

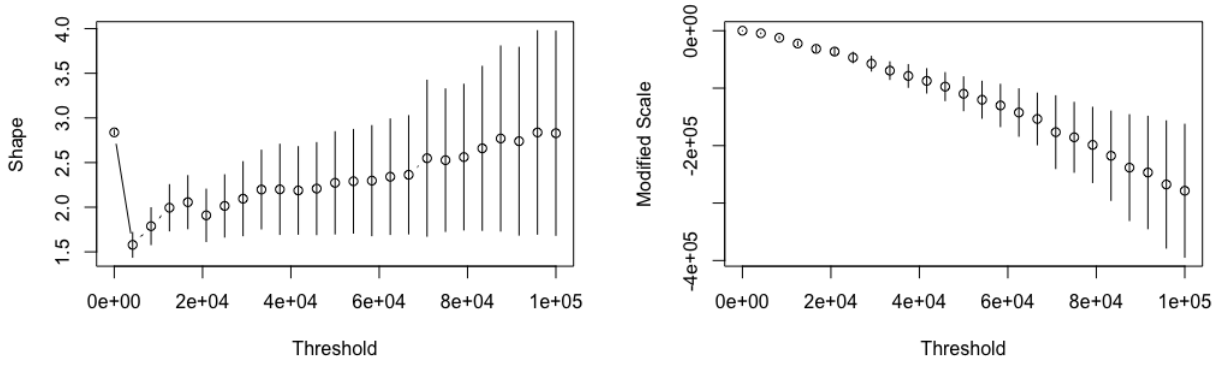

Figure 38: Parameter stability plot

6.2.2 Classification for event {BA > threshold}

Before applying extreme value theory, the cases where BA exceeds the threshold must first be predicted. To do so, several methods of classification will be discussed, namely logistic regression with generalized linear models (GLM) and generalized boosted models (GBM), random forests and naive Bayes classification. There were also attempts at logistic regression with generalized additive models (GAM), and classification with random forests. However, it performed poorer than GLMs so details won't be discussed.

To carry out this classification, a dummy variable of $\mathbb{1}_{\{\text{BA} > \text{threshold}\}}$ was created for every voxel, and was named `overthreshold`.

GLM : logistic regression

Logistic regression is a class of generalized linear models that models categorical binary dependent variables. Logistic regression does not model the dependent variable directly, but rather models the probabilities of the success by applying the logit transformation. Suppose that there are n different independent variables denoted by vector $X^t = (x_1, ..., x_n)$, and the ratio of cases where Y is a success is denoted by $\mathbb{P}(Y = 1|X) = p(X)$. The logistic function is used for the regression model [4]:

$$p(X) = \frac{e^{\beta_0+\beta_1 x_1+\beta_2 x_2+...+\beta_n x_n}}{1 + e^{\beta_0+\beta_1 x_1+\beta_2 x_2+...+\beta_n x_n}} \tag{1}$$

where $\beta_0, ..., \beta_n$ are model parameters. Now, assuming a linear relationship between the predictor variables and the logit of the event that $Y = 1$, it can be written in the following mathematical form

$$\text{logit}(p(X)) = \log(\frac{p(X)}{1 - p(X)}) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_n x_n. \tag{2}$$

Model fitting requires the estimation of model parameters, which is generally obtained by maximum likelihood [4].

To predict $\mathbb{P}(BA > u)$, we can perform logistic regression using generalized linear models using the **glm()** function with the binomial family. The following is the implementation in R:

```
#without CNT
overthreshold_glm <- glm(formula = overthreshold ~.-CNT-BA, family = binomial,
    data = train)
#with CNT
overthreshold_glm <- glm(formula = overthreshold ~.-BA, family = binomial,
    data = train)
```

To assess the logistic regression model, the confusion matrix was generated to calculate the accuracy.

BA > threshold	Predicted		Actual	Percent correct
	no	yes		
no	70943	456	72,925	0.973
yes	1,982	182	638	0.285
Overall accuracy				0.967

Table 5: Confusion matrix for the logistic regression (without CNT)

BA > threshold	Predicted		Actual	Percent correct
	no	yes		
no	70,985	419	72,925	0.973
yes	1,940	219	638	0.343
Overall accuracy				0.968

Table 6: Confusion matrix for the logistic regression (with CNT)

Here, the percent correct and the overall accuracy was calculated by taking

$$\% \text{ Correct} = \frac{\text{true positives/negatives}}{\text{actual positives/negatives}}, \quad \text{Overall accuracy} = \frac{\text{true positives} + \text{true negatives}}{\text{total number of cases}}$$

Although the overall accuracy is very high for both cases, we can observe that the number of predicted positive cases is very low compared to the actual ones, resulting in a very low correct percentage of cases with BA over the threshold. Because the predicting larger fires holds greater importance, we aim to improve the accurate percentage of actual positives in the following few classification methods to be discussed.

Moreover, it can be noted from above that CNT as a covariate in the logistic regression is of importance, especially in predicting the true positives.

GBM

Boosted trees (see section 4) can also be used for binary classification using the function **gbm()** of the package 'gbm' in R by setting `distribution = "bernoulli"` :

```
#without cnt
gbm_overthreshold1 <- gbm(overthreshold ~.-CNT-BA, distribution = "bernoulli",
    data = train, n.trees = 1000)
gbm_predict1 <- predict(gbm_overthreshold1, newdata = test, type = "response",
    n.trees = 1000)
#with cnt
gbm_overthreshold2 <- gbm(overthreshold ~.-BA, distribution = "bernoulli",
    data = train, n.trees = 1000)
gbm_predict2 <- predict(gbm_overthreshold2, newdata = test, type = "response",
    n.trees = 1000)
```

The correctly predicted number of observations over the threshold increased, improving the correct percentage of BA $>$ threshold. However, the number of false positive greatly increased as well.

BA $>$ threshold	Predicted		Actual	Percent correct
	no	yes		
no	70,635	2290	72,925	0.969
yes	412	226	638	0.354
Overall accuracy				0.997

Table 7: Confusion matrix for GBM model (without CNT)

BA $>$ threshold	Predicted		Actual	Percent correct
	no	yes		
no	70,162	2,763	72,925	0.962
yes	303	335	638	0.525
Overall accuracy				0.958

Table 8: Confusion matrix for GBM model (with CNT)

Naive Bayes Classification

Naive Bayes is a simple probabilistic machine learning model based on Bayesian classification. Bayesian classification searches for the probability of a label given some features. It uses Bayes' theorem that describes the relationship of conditional probabilities of events:

$$\text{Bayes' theorem} : \mathbb{P}(A|B) = \frac{\mathbb{P}(B|A)\mathbb{P}(A)}{\mathbb{P}(B)}$$

and combines it with the assumption that the value of a particular feature is independent of the value of any other feature.

The implementation of the naive Bayes model in R is shown in the code below using the **naive_bayes()** function in the package 'naivebayes':

```
train$overthresholdf <- as.factor(train$overthreshold)
test$overthresholdf <- as.factor(test$overthreshold)
naivebayes <- naive_bayes(overthresholdf ~.-BA-CNT-overthreshold, data = train,
    usekernel = T)
p_naivebayes <- predict(naivebayes, test, 'prob')
test$naivebayes <- p_naivebayes[,2]
train$overthresholdf <- NULL
test$overthresholdf <- NULL
```

The predictions did not change with or without CNT as a variable in the Naive Bayes model. In the confusion matrix, we can see that the falsely predicted cases significantly decreased, and the percentage of correctly predicted observations is the greatest so far. Thus, it seems appropriate to use the naive Bayes model (without CNT) for the classification of $\mathbb{1}_{\{\text{BA} > \text{threshold}\}}$.

BA > threshold	Predicted		Actual	Percent correct
	no	yes		
no	72,886	301	72,925	0.9994
yes	39	337	638	0.528
Overall accuracy				0.995

Table 9: Confusion matrix for the Naive Bayes classification (without and with CNT)

6.2.3 Distribution for excesses over threshold

Figure 39 shows the histogram of BA $>$ 2000 acres and the GP distribution with parameters $\sigma = 3121.9, \xi = 0.9852$. These parameters were estimated by modeling the excesses using maximum likelihood (see below for implementation in R).

```
gpdmodel <- evir::gpd(data = data_scaled$BA, threshold = 2000, method = "ml",
    information = "observed")
gpdmodel$par.ests
```

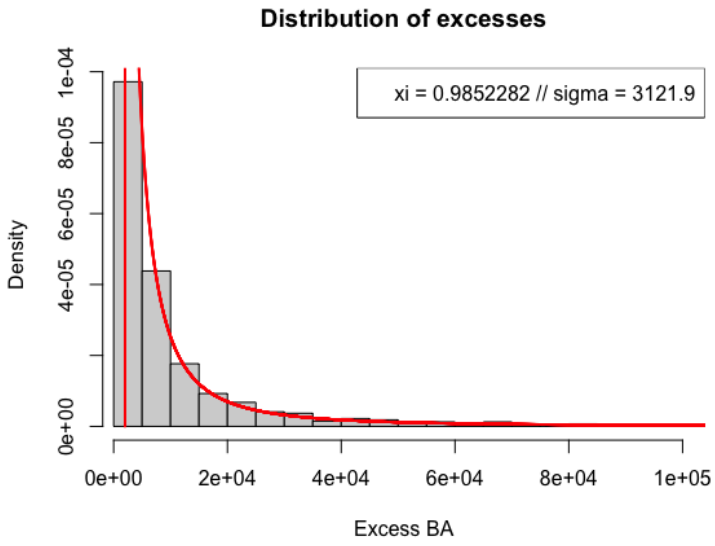

Figure 39: Distribution of BA $>$ 2000

To fit GP distributions with covariates the **vglm()** function in the 'VGAM' package was used. The following two models were used to fit the excesses of BA on the cases that was predicted with naive Bayes in section 5.2.2.

```
#without count
modelvglm1 <-  vglm(BA~.-mean_monthly_CNT-CNT, family = gpd(threshold = threshold),
    link = "log", data = train, subset = BA > threshold)

# with count
modelvglm2 <-  vglm(BA~.-mean_monthly_CNT, family = gpd(threshold = threshold),
    link = "log", data = train, subset = BA > threshold)
```

The following table shows the first scale and shape parameters for the first few cases. The shape parameter is constant with $\xi = 0.7508032$ and the scale parameter varies with mean of 5522. This implies that the shape of the distribution stays the same as the spread varies among different cases; the greater the scale parameter, the bigger the spread.

scale σ	shape ξ
2583.933	0.7508032
3520.953	0.7508032
6273.917	0.7508032
3150.891	0.7508032
2406.263	0.7508032

Table 10: Estimated GPD parameters (scale and shape)

Using the estimated parameters, probability distribution of each case can be calculated (and thus $p_2 = \mathbb{P}(BA \leq u + v | BA > u)$).

6.2.4 Final model : Generating the probability matrix with GBM

To generate the final probability matrix, the cases where CNT values were available were considered separately from cases where CNT values were masked. Classification methods was used and compared to predict cases where BA exceeds the threshold. In these cases, two GPD models were fitted above the threshold depending whether the CNT value was available in the voxel. To model the burnt area under the threshold, a simple GBM was used with log-normal distribution of BA+1.

Table 11 compares the three different methods considered above to classify the event {BA > threshold} by computing their scores. The scores were calculated using the scoring function provided in the description of the EVA Data Challenge. These three methods were also compared with a base GBM model without considering extremes of burnt area, to make sure that considering the extremes improved the model. We can see that the naive Bayes classification method for extremes yielded the best score with significant improvement compared to the improvements made with other methods.

	Without extremes	Classification of BA > threshold		
		GLM (logistic)	GBM	Naive Bayes
Without CNT (as a covariate)	0.03966	0.03962	0.04006	0.03594
With CNT	0.03492	0.03459	0.03451	0.03008

Table 11: Table of scores for each classification method

Finally, a different thresholds were considered, because we only relied on intuition and quantiles to select the threshold of 2,000 in the beginning. Table 12 compares the scores calculated for thresholds between 500 and 5,d00. It can be seen that 1,500 has the lowest score, so the final probability matrix was generated with the threshold of 1,500 acres.

Threshold	500	1000	1500	2000	5000
Without CNT (as a covariate)	0.03590	0.03521	0.03428	0.03594	0.03586
With CNT	0.03200	0.03091	0.02990	0.03008	0.03087

Table 12: Table of scores for different thresholds

7 Conclusion and results of the competition

We resume here the final models we submitted for the competition, as well as how we performed.

- For CNT, we used a GBM trained on all non-masked data, using 10,000 trees, interaction depth of 6, poisson loss function and default learning rate of 0.1.
- For BA, we also used a GBM with log-gaussian distribution trained using the same parameters as the ones for the CNT model. The GBM model was combined with the GPD model for the extreme burnt areas over a threshold of 1,500 acres.

In the competition, we ended up 7th place with 13 participating teams in total. In particular we were 2nd place for CNT and 11th for BA. What we were very surprised by is that we performed worse for BA during the final predictions, where we used extreme value theory, a larger training set and a model using CNT for the predictions, than during the preliminary predictions. A possible explanation is that we used only a random set of odd years as a testing set for the extreme values and that the masked data taken out contained more extremes. Hopefully, we will get a better understanding once the masked data is revealed.

8 Acknowledgments

We would like to thank Professor Anthony Davison for supervising the project and for his valuable guidance and support. We really enjoyed exploring and learning about different topics for the project. We really appreciate the recommendations of articles, books and presentations, as well as the directions provided, all the while having the liberty to come up with our own ideas. Not all the ideas we explored on our own were fruitful in the context of this project, but they will certainly give us important knowledge that could be used in the future.

Also, special thanks to Jonathan Koh and Ophelia Miralles for assisting our weekly meetings and presentations, for providing useful guidance and new insights.

References

[1] Stuart Coles. *An introduction to statistical modeling of extreme values*. Springer Series in Statistics. London: Springer-Verlag, 2001. ISBN: 1-85233-459-2.

[2] Sergi Costafreda-Aumedes, Carles Comas, and Cristina Vega-Garcia. "Human-caused fire occurrence modelling in perspective: a review". In: *International Journal of Wildland Fire* 26.12 (2017), pp. 983–998. URL: https://doi.org/10.1071/WF17026.

[3] K.F. Turkman J.M.C. Pereira. "Handbook of Environmental and Ecological Statistics". In: 1st ed. Chapman and Hall/CRC., 2019. Chap. Statistical models of vegetation fires: Spatial and temporal patterns, pp. 401–414.

[4] Gareth James et al. *An Introduction to Statistical Learning: With Applications in R*. Springer Publishing Company, Incorporated, 2014. ISBN: 1461471370.

[5] Haiganoush K.Preisler. "Probability based models for estimation of wildfire risk". In: (2004).

[6] Shu Li and Tirtha Banerjee. "Spatial and temporal pattern of wildfires in California from 2000 to 2019". In: *Scientific Reports* 11.1 (2021), p. 8779. DOI: 10.1038/s41598-021-88131-9. URL: https://doi.org/10.1038/s41598-021-88131-9.

[7] NWCG. *Fire Weather Index (FWI)*. URL: https://www.nwcg.gov/publications/pms437/cffdrs/fire-weather-index-system.

[8] Thomas Opitz. *Wildfires and their extremes: a global challenge*. 2021. URL: https://www.maths.ed.ac.uk/school-of-mathematics/eva-2021/competitions/data-challenge.

[9] Volker C. Radeloff et al. "Rapid growth of the US wildland-urban interface raises wildfire risk". In: *Proceedings of the National Academy of Sciences* 115.13 (2018), pp. 3314–3319. ISSN: 0027-8424. DOI: 10.1073/pnas.1718850115. eprint: https://www.pnas.org/content/115/13/3314.full.pdf. URL: https://www.pnas.org/content/115/13/3314.

[10] Simon N. Wood. *Generalized additive models : an introduction with R*. fre. Second edition. Chapman Hall/CRC texts in statistical science. Boca Raton: CRC Press/Taylor Francis Group, 2017. ISBN: 1498728332.

[11] Dexen D.Z. Xi et al. "Statistical Models of Key Components of Wildfire Risk". In: *Annual Review of Statistics and Its Application* 6.1 (2019), pp. 197–222. DOI: 10.1146/annurev-statistics-031017-100450. eprint: https://doi.org/10.1146/annurev-statistics-031017-100450. URL: https://doi.org/10.1146/annurev-statistics-031017-100450.

[12] P. de Zea Bermudez et al. "Spatial and temporal extremes of wildfire sizes in Portugal (19842004)". In: *International Journal of Wildland Fire* 18.8 (2009), pp. 983–991. URL: https://doi.org/10.1071/WF07044.