# Predicting extreme events with Random Forest and K-Nearest Neighbors

Thomas Krieger

December 2022

## 1 Abstract

In this report, we try to compare two different algorithms for classification. The second goal is to see if these algorithms could be used for predicting extreme events in the United States, like thunderstorm wind, storm hail or winter storm for example, with a simple workflow. We are going to use the Random Forest and the K-Nearest Neighbors with GridSearchCV to tune the hyperparameters.

## 2 Introduction

The United State is a really large territory with lots of different region. Some of the are more victim of extreme events than other. This really different depending on where they are. Knowing that for this type of events is useful to make predictions faster. Especially, because the probability of extreme events could increased, as the intensity, with the global warming. Some authors have already written on the subject using different algorithms to predict the occurrence of the events (Meiyazhagan et al., 2021). In addition protecting people and cities, this could be also used to help agriculture (Gaitan et al., 2020). In our case, we focused on a simple approach to try to predict extreme events in the US, with the use ofK-Nearest Neighbors and Random Forest.

## 3 Data

The dataset come from the National centers for environmental information of the United States (NCEI) in the National Oceanic and Atmospheric Administration (NOAA). This dataset provide different informations about extreme events in the United States for the year 2009. Its dimensions represents the 57398 listed the events considered as exrteme with 51 different features. The dataset is in open access.

### 3.1 Data processing

Among these features, certain could be useful for our project like the states, dates, type of events and other not, like the description of the events and the format file. So a sorting is necessary. We keep only the important features. However, there is lots of Nan value in some features especially for those related to the latitude. To continue, we need to replace the Nan values by 0. We decides to split the data with 30 percent for the testing set and 70 for the training set. After that, we normalized the x of the training and testing data with 'preporcessing' from sklearn librairy, and the 'fit transform' function.

## 4 Methodology

Our main inspiration for the methodology is the first notebook of the first week in the Machine Learning for Earth and Environmental Sciences for K-Nearest NeighborS and GridSearchCV. For Random Forest, we use the third notebook of the second week.

### 4.1 K-Nearest Neighbors

Our first classification in the K-Nearest Neighbor. It is part of the family of the supervised algorithms for machine learning. It means the algorithms work on data that are already labelled, instead of the unsupervised algorithms. The way how the KNN algorithm works is not really complex. A new point take place in a space where other are already here. Those points get a class but not the new one. To know what will be its affiliation, the algorithm attribute the class with the most of points that are labelled in the choosing area. This last thing might be the Euclidian distance between them. To tune the hyperparameters we use GridSearchCV and param grid.

```
knn_clf = KNeighborsClassifier()
```

```
param_grid = {'weights':['uniform', 'distance'],
              'n_neighbors':[3,4,5,6]
              }
```

```
grid_search = GridSearchCV(
    knn_clf,
    param_grid,
    cv = 6,
    verbose = 3)
```

```
grid_search.fit(x_train, y_train)
```

Figure 1: Accuracy for Random Forest

### 4.2 Random Forest Classifier

Random Forest algorithm is ensemble of decision three. the principle is a progression through a range of choice between two options and so on, until arriving at the and and being labelled. For Random Forest, what you have had for several tree is combined to give only one final result.

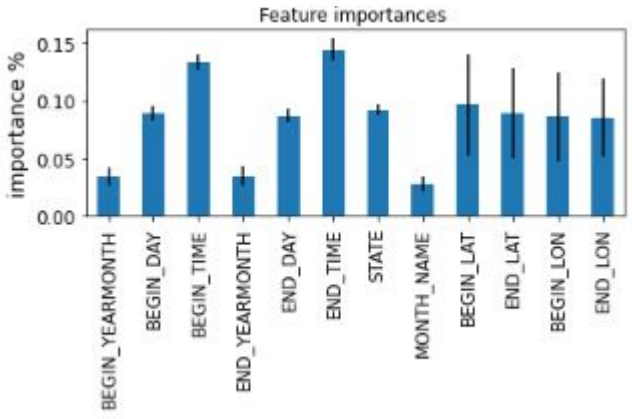


Figure 2: Importance of the features with Random Forest

## 5 Results

The accuracy of the model are roughly similar. Considering the training and testing sets, the Random Forest model is a bit more precise than the K-Nearest Neighbors model. For the first set the results are respectively 99 and 98.99 percent. And for the test set we obtain 76.68 against 73.54 percent. The fact that Random Forest give us a better result was what we were expecting. The model have a better outcome with most input, training has 70 percent of the data.

## 6 Discussion

The accuracy may not be really bad, but our methodology could be used in reality because it is to simple. I f we wanted to keep this methodology, the hyperparameters should be the first thing to change. We tried to maximize The data set maybe were not the best for what we wanted to do. Maybe the Deep Learning with a Convolutional Neural Network would give us better results and is enough complex to take into account the maximum of possible aspects for predicting extreme events(Liu et al., 2016). Another thing that can plays role in results is the split of the data. The distribution between the testing and the training set is probably not the best.

```
# model prediction for train set
y_pred_train = grid_search.predict(x_train)
# model prediction for test set
y_pred_test = grid_search.predict(x_test)

# accuracy score for train set
accuracy_train = accuracy_score(y_train, y_pred_train)
# accuracy score for train set
accuracy_test = accuracy_score(y_test, y_pred_test)

# print the accuracy of the models
print(f'The accuracy of the model with train set is
{accuracy_train:.2%}')
print(f'The accuracy of the model with test set is
{accuracy_test:.2%}')
```

```
The accuracy of the model with train set is 98.99%
The accuracy of the model with test set is 73.54%
```

Figure 3: Accuracy for K-Nearest Neighbors

```
# accuracy score fore the train set
rfc_train_acc = accuracy_score(y_train, rfc_train_preds)
print(f'accuracy score for the train set :{rfc_train_acc:.2%}')

# accuracy score for the train set
rfc_test_acc=accuracy_score(y_test, rfc_test_preds)
print(f'accuracy score for the test set :{rfc_test_acc:.2%}')
```

```
accuracy score for the train set :99.00%
accuracy score for the test set :76.68%
```

Figure 4: Accuracy for Random Forest

# 7 Conclusion

This experiment is very basic. We did not really innovate about the hyperparameters and the model optimization. But, the models seem not really bad with the accuracy percentage. However, representing the reality could ask more elements than we used in this paper and other algorithms.

# 8 Links to code and dataset

The dataset is here
the code is here

# 9 References


Gaitán, C. F. (2020). Chapter 7—Machine learning applications for agricultural impacts under extreme events. Dans J. Sillmann, S. Sippel, S. Russo (Éds), Climate Extremes and Their Implications for Impact and Risk Assessment (pp. 119-138). (S.l.): Elsevier. https://doi.org/10.1016/B978-0-12-814895-2.00007-0

Liu, Y., Racah, E., Prabhat, Correa, J., Khosrowshahi, A., Lavers, D., ... Collins, W. (2016, 4 mai). Application of Deep Convolutional Neural Networks for Detecting Extreme Weather in Climate Datasets. arXiv. Repéré à http://arxiv.org/abs/1605.01156

Meiyazhagan, J., Sudharsan, S., Venkatesan, A., Senthilvelan, M. (2021). Prediction of occurrence of extreme events using machine learning. The European Physical Journal Plus, 137(1), 16. https://doi.org/10.1140/epjp/s13360-021-02249-3