Flood risk predictions via machine learning methods

A Preprint

Vladislav Pyzh
MIPT
Moscow, Russia
pyzh.va@phystech.edu

Abstract

In this work we observe extreme weather condition predictions, we particularly focus on floods. We make predictions in short-term periods assuming we have stationary time series and in long-term periods assuming we have not stationary time series. The main feature of this problem is that we want to predict extreme values with high accuracy, whether prediction of low values is not very sufficient.

1 Introduction

Extreme weather events can lead to high economic costs thus it is preferable to predict them. Main goal of our work is to develop model, which can be used in practice to predict these events. Formally speaking we are solving classification task on geo-spatial time series. This task contains two different situations: predictions on stationary time series in short-term periods and not stationary in long-term periods. We assume that weather in period shorter than 5-10 years does not have common trend, so we analyze stationary time series. On the other hand in long run we can assume that weather coditions are likely to change, so we have to take into account that this time series can be not stationary. The main issue on this problem is that events that we can interpret as positive object in our classification task are extremely rare. Speaking more precisely from 1980's there were less than 10 big flood in state California. Thus we use several techniques like SMOTE and under-sampling. Also we should notice that prediction model has to provide low recall, because it is important, not to skip events in this case. Another goal of our work is combining state-of-the-art solutions to improve current result in sphere. We base our decision on works ? and ?. We implement (Подробнее описать какие state-of-the-art методы мы хотим использовать в работе, в чем заключается её новизна?)

2 Problem statement

As mentioned before we solve classification task. Let $Tr = (X_{tr}, y_{tr})$ be our train set, $\widehat{y}(Tr)$ our model, thus we solve optimisation problem

$$y^* = arg\min_{\widehat{y}} \sum_{(X_{ts}, y_{ts})} L(\widehat{y}(X_{tr}, y_{tr}), y_{ts})$$

Lets formally describe what actually means that our model has to achieve low recall (На этой стадии пока не обозначили какой результат именно хотели получить)

3 Planning Experiment

During experiment we aimed to achieve model hyperparameters that improve our known results. Earlier we reached these results:

We use combine linear models, different variations of boosting, random forest and svm with techniques which solves extreme values rarity. Overall we can describe our pipeline: split our data on test and train, use some algorithm to solve class disbalance problem in train, use grid search to tune our model parameters via cross validation on train, measure our prediction accuracy.

Model	SVM and SMOTE	SVM and RU
precision	0.151	0.147
recall	0.367	0.533
accuracy	0.925	0.901
F1-score	0.214	0.230

Таблица 1: Our previous results

In our experiment we use meteorological data across California from 1980-01-01 till 2019-12-31 only in winter. We have 8 features, described below:

- h500 – 500 hPa geopotential height
- ivt - integrated vapor transport
- qv2m- 2-meter specific humidity
- slp - sea level pressure
- tpw -total precipitable water
- uqv – zonal component of integrated vapor transport
- vqv – meridional component of integrated vapor transport
- w500 – 500 hpa vertical pressure velocity

Thus in this experiment we compare a well-known machine learning techniques with our data. Since main goal of out work is to predict events, most effective models we use in Time Series analyze with some extra features like shifted features, mean values and so on.

Final result of this experiment is pivot table, where we can easily compare models.

4 Run basic code and preliminary results

Below we give an example of basic experiment code. Unfortunately google colab resources are too restricted, so now we stucked on this stage (перезапущу еще на персональном компьютере, должно получиться).

```
search_cat = RandomizedSearchCV(
        estimator=CatBoostClasifier(verbose=0),
        param_distributions={
            "reg_lambda": np.logspace(-4, 0, 30),
            "n_estimators" : np.arange(450, 1000, 20),
            "learning_rate" : np.linspace(0.001, 0.02, 40),
            "max_depth": np.arange(4, 20)
        },
        scoring='recall',
        n_iter=25,
        n_jobs=-1
)
search_cat.fit(X_train_smote, y_train_smote)
```

Nevertheless we can obtain example of final result without RandomizedSearchCV on default CatBoostClassifier parameters.

- Accuracy = 0.948
- Precision = 0.167
- Recall = 0.368
- F1 Score = 0.230

For this experiment we also have confusion matrix.

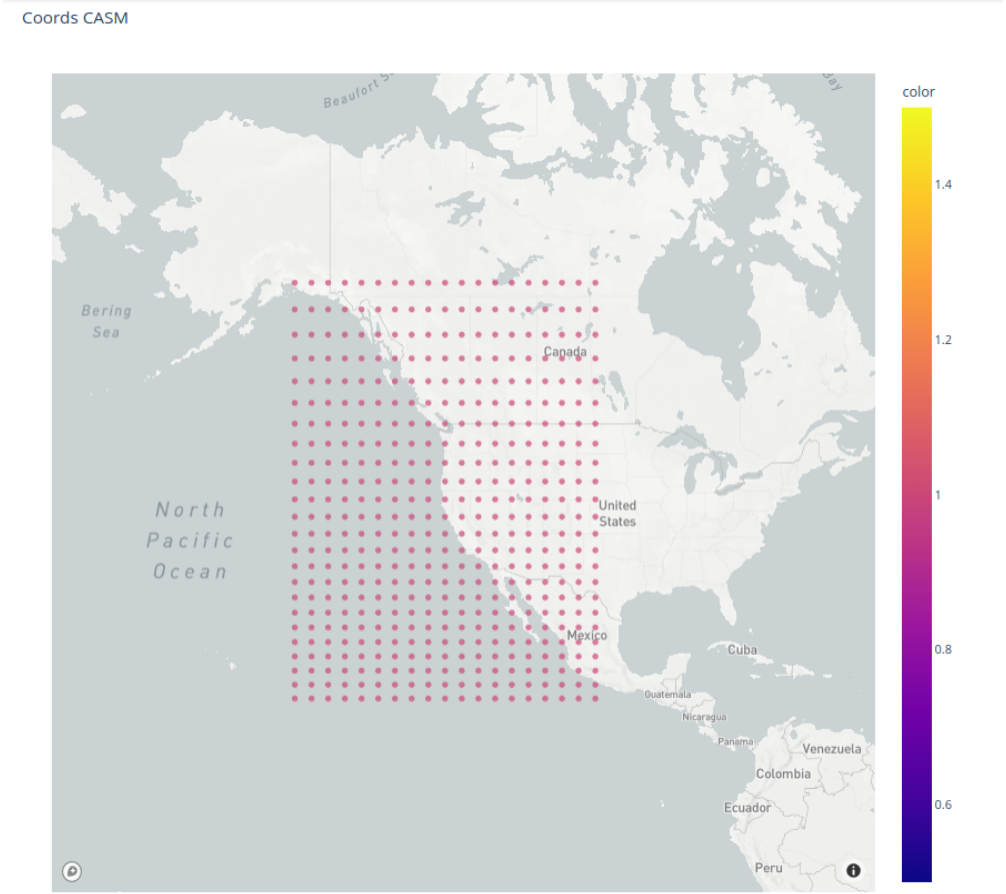
True Negative = 848	False Positive = 35
False Negative = 12	True Positive = 7

Таблица 2: Confusion matrix

5 Experiment description

During this experiment we tested CatBoost, SVC with linear and rbf kernels, RandomForest and Logistic regression. To nullify big class disbalance we used SMOTE and RandomUndersampling. We used geo-spatial dataset with data across west coast of the United States, we illustrated it here. It has approximately 400 rows about every point in winter only.

Dataset points on USA map



6 Experiment results

Dataset points on USA map

	accuracy	precision	recall	conf_mat	name	sampling
0	0.980044	1.000000	0.052632	[[883, 0], [18, 1]]	CatBoost	None
1	0.949002	0.170732	0.368421	[[849, 34], [12, 7]]	CatBoost	Smote
2	0.733925	0.066667	0.894737	[[645, 238], [2, 17]]	CatBoost	Under
3	0.978936	0.000000	0.000000	[[883, 0], [19, 0]]	SVC-linear	None
4	0.829268	0.090909	0.789474	[[733, 150], [4, 15]]	SVC-linear	Smote
5	0.770510	0.076577	0.894737	[[678, 205], [2, 17]]	SVC-linear	Under
6	0.978936	0.000000	0.000000	[[883, 0], [19, 0]]	SVC-rbf	None
7	0.746120	0.066116	0.842105	[[657, 226], [3, 16]]	SVC-rbf	Smote
8	0.427938	0.015595	0.421053	[[378, 505], [11, 8]]	SVC-rbf	Under
9	0.977827	0.333333	0.052632	[[881, 2], [18, 1]]	LogReg	None
10	0.833703	0.093168	0.789474	[[737, 146], [4, 15]]	LogReg	Smote
11	0.745011	0.069388	0.894737	[[655, 228], [2, 17]]	LogReg	Under
12	0.980044	1.000000	0.052632	[[883, 0], [18, 1]]	RandomForest	None
13	0.952328	0.184211	0.368421	[[852, 31], [12, 7]]	RandomForest	Smote
14	0.747228	0.069959	0.894737	[[657, 226], [2, 17]]	RandomForest	Under

7 Plot list

In this paper we use these plots:

- Data visualisation
- Results pivot table
- (Consider using this ??) Results comparation with lines?
- (Code is not done yet) DL techonologies visualistion