Evaluation of performance of XGBoost Regressor and Bayesian Network used for Streamflow Prediction before and after Rocky Fire event near Lower Lake, CA

Arezoo Bybordi , Maryam Akrami
Fall 2020 Data Mining Final Project, Graduate Center CUNY

ABSTRACT

Streamflow is an important measurement in monitoring drought and floods which with applications in fishery industries, agricultures, etc. In this project we aimed to predict daily and annual streamflow values before and after one of the most destructive fire events namely, Rocky fire event which happened near Lower Lake, California , non urban area close to a fishery in the year 2015 using two different models xgboost and bayesian network. Since,the location we are studying is close to fishery prediction of streamflow values, find its application in real life.We used a climate dataset which is provided by the US Geological Survey, which has daily streamflow data from the year 1980 to 2019. These measurements include daily data, streamflow and precipitation from different locations. We have evaluated the performance of XGboost and Bayesian model before and after fire and we observed a deterioration of performance of XGBoost after the fire.This approach comprises classic techniques such as Pattern association, boosting tree regressor(xgboost), (Autoregressive Moving Average Model, ARMA) and novel ones, based on Artificial Intelligent for hydrological research (Bayesian Networks, BNs).

Data Description, Cleaning, Preprocessing

We used a Climate dataset provided by the US Geological Survey which is a daily dataset for 40 years between 1980 -2019 in more than 100 locations in California. We have decided to pick a non-urban location for our analysis to avoid complex trends that come from factors we are not investigating. Thus, the closest non-urban location to the Rocky fire event found by their coordinates is being used, which is outside Napa and it

is close to a fishery.

Deconstruction:

Are there multiple fire events in this location? We have checked the history for this location on the California's Fire Department website and *there has not been any other fire event in this location in the time interval that we are considering (1980-2019)*. There have been other fire events in the locations in that proximity but not in the coordinates that we have considered and because in considering two factors "time" and "space" we have fixed "space" and we are considering factor "time", we are not considering the locations in the proximity of the fire events and we are just considering this location. Also, note that the California's Fire department website seems very complete and exhaustive and it has all of the fire events from almost 1940 to the present year. Also, it has all significant and insignificant fire events so the ones that are not in the website, must have been "too insignificant" to be included in the website.

Problem Definition/ Research Question

Our problem definition is: *"Does the performance of XGBoost Regressor and Bayesian network in streamflow prediction deteriorate after the Rocky Fire event?"*

Which contributes to a deeper research question which is a feature selection question: *"Should measurements related to a fire event such as acres burnt, number of structures destroyed, etc. be used as a part of features used in streamflow prediction?"*

Deconstruction:

Different way to state the research question:

Comparison of performance of XGBoost and Bayesian network on Streamflow prediction before and after Rocky fire event (based on errors (MSE,MAE), R-squared, and ability to predict peaks)

Why do we compare these two models (XGBoost, Bayesian Network)? Because Ni et al. [6] used XGBoost and Molina et al. [2] used this specific type of Bayesian network that we used. And they are two reliable recent papers that have used these two models for their streamflow prediction, and we wanted to figure out the effect of fire using these

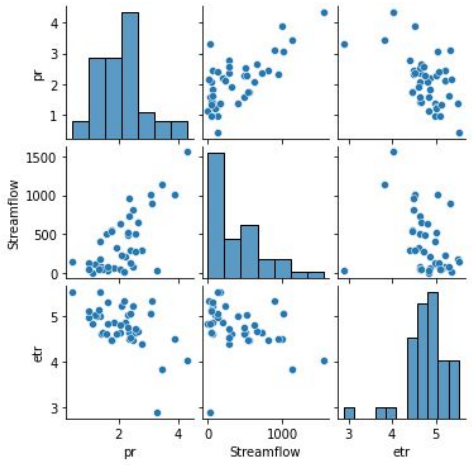
two reliable models that has been used and also we added ARIMA as the baseline model.

Feasibility (Is it on instance of one fire event as discussed, or extensible?) Because the hydrological features of every location is very different than other locations, we have a ***challenge*** in generalizing the results for this analysis to all other locations and we cannot say that. It may suggest that if in future there was another fire event, this may be the possible pattern but this is not extensible to other locations. Also, one of the ***limitation*** we have is that we have used some specific models "XGBoost", "Bayesian network", and "ARIMA" and this result limits to only these models and we may not know if using other models we observe the same change in performance before and after fire. But because these are good models that have been recently used for prediction they can be considered reliable to some extent.

A possible hypothesis as the motivation of this comparison: Our motivation (bigger picture) is to check and see if there are other factors related to fire that impact streamflow prediction that might help with streamflow prediction. Now, if the performance of models that are using only "features that do not include features related to fire" gets worse after a fire event, this might be a *sign* that fire event impacted the ability of model to predict and maybe if we include features related to fire in our prediction, we will get a better streamflow prediction model for after fire. (Although a lot of other factors may impact this, but this is a *sign* that maybe fire features are worth investigating)

Exploratory Data Analysis

In figure 1, bar plot of frequency distribution of each one of the measurements "Streamflow", "Precipitation" and "Evapotranspiration (ETR)" are displayed. This plot shows skewness towards zero. In fact, the skewness coefficient for this variable is 1.16. On the other hand, etr seems to be right-skewed with skewness coefficient -1.6. Also, scatter plots of different measurements are displayed here. By looking



at the scatterplot of streamflow and precipitation, a correlation close to +1 can be expected while there is no sign of a correlation between etr and precipitation or etr and streamflow.

Deconstruction:

*A **sign** not to use feature "ETR" for prediction*:

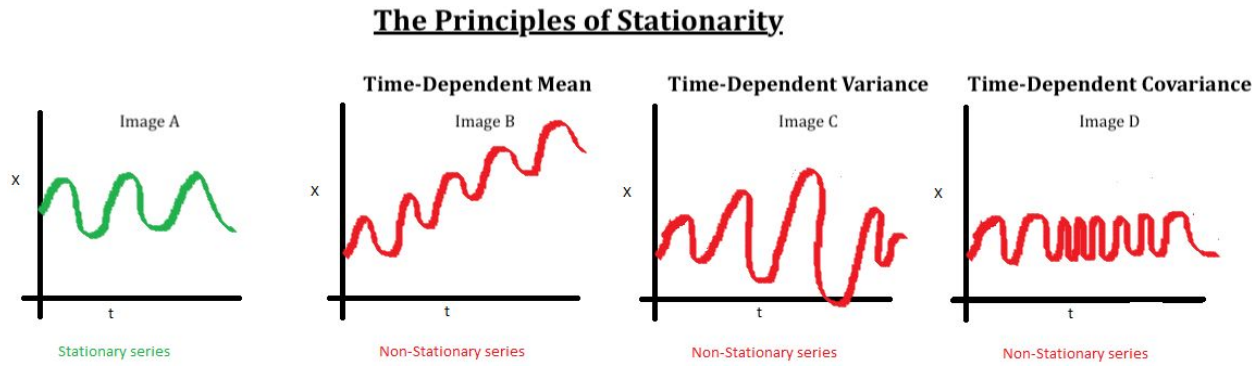
In the related research area, "evapotranspiration/etr" is considered a related feature in streamflow prediction. But for this specific location (fixing "space") by looking at the scatterplot explained above we see that there is no significant correlation between etr and streamflow or etr and precipitation. So this might be a sign that etr for this specific location is not correlated to streamflow and therefore it is not useful to add it into our prediction. In fact, it might add unnecessary complication or irrelevant information. To confirm this fact, in one of our predictions with XGBoost we have done the prediction once including etr ("streamflow", "precipitation", "etr") and once without etr ("streamflow" and precipitation") and we observed a confirmation to this hypothesis. The *performance of XGBoost was better when we did not use "etr*". So we continued using XGBoost without "ETR". For the two other models, ARMA, and Bayesian network, due to the short amount of time we had, we did not have time to make our model in a way that they use other features "precipitation" and "etr". We have only used streamflow for them. In future work, we are planning on including these two features in our predictions with ARMA and Bayesian network.

Deconstruction:

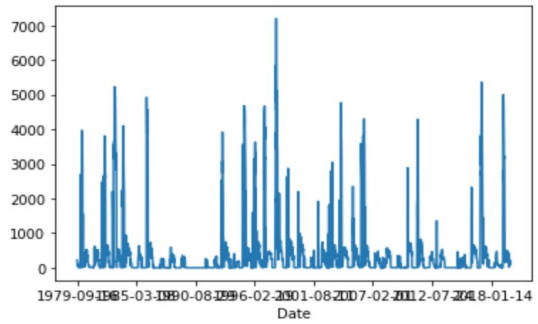
Some signs of the time series being stationary:

1) The plot of non-stationary time series doesn't show stable and almost constant mean and variance all over the time series while a stationary time series shows these statistics almost constant over time. This can be seen in the following

figure



In our case the plots show stationary time series.



2) Mean and variance should be stable in all intervals of time series. We took the first half of the time series and the second half and *we saw that the mean and variance stay almost the same*.

Mean and variance of the two sections for “Streamflow”:

```
mean1=457.167061, mean2=319.673450
variance1=915820.833629, variance2=485785.688170
```

Mean and variance of the two sections for “Precipitation”:

```
mean1=2.154399, mean2=1.992335
variance1=50.789850, variance2=50.804769
```

Mean and variance of the two sections for “ETR”:

```
mean1=4.644795, mean2=4.954748
variance1=8.024533, variance2=9.439436
```

Checking the Stationarity of Time Series (Augmented Dicky-Fuller test)

Stationary time series have constant mean, variance, autocorrelations, etc. over time. This is important because if a time series is not stationary some models cannot be

applied to them and they must be transformed to a stationary time series before the model is applied to them. We have used a statistical test to check the null hypothesis of the following model of the time series having a unit root for the coefficient of y_{t-1}.

$$\Delta y_t = \alpha + \beta t + \gamma y_{t-1} + \delta_1 \Delta y_{t-1} + \cdots + \delta_{p-1} \Delta y_{t-p+1} + \varepsilon_t$$

This also can be formulated as $y_t = D_t + z_t + \varepsilon_t$

Where D_t is the deterministic component, (trend, seasonality) and z_t the stochastic component. If in the first formula there is a unit root for the time series this means that there is a stochastic trend in the time series that is unpredictable. In other words, considering the first equation if there is a unit root it means that there is a stochastic coefficient for the previous lags that makes the series non-stationary. A lag p is introduced and then the higher order formulation of the ADF will be made. Then it performs a test on t-values of the coefficients for previous lags. In our case the null hypothesis was rejected and our time series is stationary. (statistic value ~-11, p-value =0)

More explanation in the appendix attached to the end.

Reason for using XGBoost Tree

A type of decision tree regressor, finds the most important features to check at the beginning and checks the rest in the order of their importance. In our case, using a 10 day window, for example, finds out if the information of 5 days ago is more important to predict the streamflow of the 11th day or the 6th day.

Deconstruction:

Why did we choose XGBoost? In Ni et al. [6], which is a very recent paper published in the most reputable journal in hydrology, they have used a XGBoost model and also a Gaussian Mixture XGBoost model that had a very good performance on predicting the streamflow. We are aware that this may not be the best approach for our location as the hydrological properties of different locations are different, but we tried to compare the performance of XGBoost before and after fire because we considered XGBoost as a probable good model to predict streamflow.

Approach 1: XGBoost Regression Tree

Our dataset is relatively large, therefore we decided to implement a machine learning method similar to the tree regression approach discussed in class. The XGBoost first makes an initial prediction and then it fits a regression tree to the residuals. It uses a similarity score as a criteria for comparing each node. Threshold decision and splitting criteria for tree is a measure called gain. The output of the initial tree will be calculated and a fraction of it will be summed with the previous prediction and will be given to the model again to be fitted to a new improved regression tree.

Deconstruction:

We did the XGBoost first with normalized data and then without normalized data and we chose the one without normalization: The performance of XGBoost using normalized data was very poor in comparison to the one without normalization. So we continued doing the analysis and hypothesis testing using the one without normalization.

We did the XGBoost with features "Streamflow", "Precipitation", "ETR" and another one with only "Streamflow", "Precipitation". Here is why: When we were not using "ETR" we had lower error values and higher R squared values over all 15 years that we did the prediction. So we decided to continue with the one that has better performance without ETR. This is a confirmation on what we expected before, because considering the scatterplots, we saw that "ETR" didn't have correlation with other two variables (for this specific location) and therefore it is adding extra unnecessary complexity and irrelevant information to the model.

MSE in 15 years (before and after fire) ***with*** ETR: ~310
MAE in 15 years (before and after fire) ***with*** ETR: ~121
R squared in 15 years (before and after fire) ***with*** ETR: 0.78
MSE in 15 years (before and after fire) ***without*** ETR: ~285
MAE in 15 years (before and after fire) ***without*** ETR: ~109
R squared in 15 years (before and after fire) ***without*** ETR: 0.81

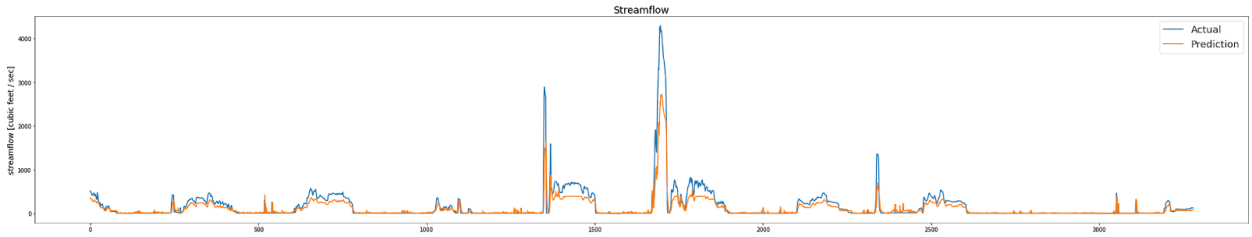
Evaluation XGBoost Regression Tree before and after Fire

We have used 10 day windows of data to predict the 11th day. For training we have used daily data from 1980 - 2005, predicted daily values of 2005 - 2014 before fire and

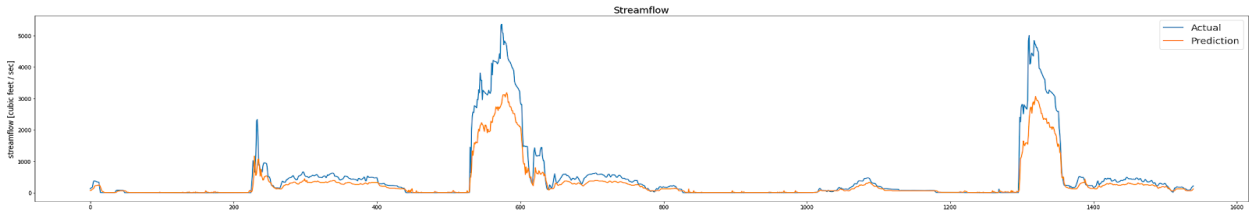
daily values of 2015 - 2019 after fire. We compared the performance in terms of accuracy and peak predictions. Our observation is an increase in error and R squared after fire. Also, peak prediction is an important application of streamflow prediction and we can see that peaks are predicted better before the fire.

MSE **before** fire	183.917912	MSE **after** fire	428.618875
MAE **before** fire	72.842713	MAE **after** fire	187.210463
R Squared **before** fire	0.787544	R Squared **after** fire	0.812068

Line Chart *Before* fire (Predicted: Orange, Blue: Real values):



Line Chart *After* fire (Predicted: Orange, Blue: Real values)



Prediction using Autoregressive Moving Average (ARIMA) model:

We did an Autoregressive model as a baseline model in addition to our previous models. We have only used “Streamflow” as the feature for prediction because we didn’t have time to code a more complex model that includes more features (“precipitation” and “etr”) but in our future work, we plan to include them. Similar to the context of the *Normalization*: We have normalized the streamflow because ARIMA works with normalized dataset and assumes they have Gaussian distribution.

Results of comparing the AIC, BIC, p-values, log likelihood on residuals for different lags (the reasoning behind model selection):

In choosing the best length of lag for the ARIMA model, we decided to check different lags of ARIMA for their AIC, BIC, their variables' p-values and the log likelihood of the residuals. We compared lags 2, 5, 10, 15 and we observed almost the same performance for each model. All of them have AIC ~ -9167 , BIC ~ -9127 , log likelihood ~ 4500. Among these different lengths of lags, we decided to go with p = 5 , because it seemed that more than 5 order it is unnecessary complexity that we are adding to the model and also we looked at the p-values of the coefficient of each individual AR variable in different lags and for higher than 5 order lags, we see a high p-value for the coefficients of higher order lags. So we decided to not choose a model with order higher than 5. Also, model order 5 had slightly better AIC value in comparison to model order 2 so it means that we are doing a slightly better model selection choosing 5 order model because model order 5 has lower AIC. Here is the summary of the evaluation of ARIMA with p=2, 5, 10 , 15.

ARIMA(2,1,0): (small and good individual p-values)

Model:	ARIMA(2, 1, 0)	Log Likelihood	4499.214
Method:	css-mle	S.D. of innovations	0.178
Date:	Sat, 19 Dec 2020	AIC	-8990.427
Time:	22:50:59	BIC	-8960.062
Sample:	09-17-1979	HQIC	-8980.340
	- 10-14-2019		

	coef	std err	z	P>|z|	[0.025	0.975]
const	4.875e-07	0.002	0.000	1.000	-0.004	0.004
ar.L1.D.Streamflow	0.2966	0.008	36.112	0.000	0.280	0.313
ar.L2.D.Streamflow	-0.1123	0.008	-13.676	0.000	-0.128	-0.096

ARIMA(5,1,0): (small and relatively good individual p-values ,AIC smaller than order 2)

Model:	ARIMA(5, 1, 0)	Log Likelihood	4503.247
Method:	css-mle	S.D. of innovations	0.178
Date:	Sat, 19 Dec 2020	AIC	-8992.493
Time:	22:51:13	BIC	-8939.353
Sample:	09-17-1979 - 10-14-2019	HQIC	-8974.841

	coef	std err	z	P>|z|	[0.025	0.975]
const	4.561e-07	0.002	0.000	1.000	-0.004	0.004
ar.L1.D.Streamflow	0.2971	0.008	35.948	0.000	0.281	0.313
ar.L2.D.Streamflow	-0.1154	0.009	-13.389	0.000	-0.132	-0.099
ar.L3.D.Streamflow	0.0096	0.009	1.107	0.268	-0.007	0.027
ar.L4.D.Streamflow	-0.0240	0.009	-2.780	0.005	-0.041	-0.007
ar.L5.D.Streamflow	0.0106	0.008	1.286	0.198	-0.006	0.027

ARIMA (10,1,0): (AR 10 with relatively high and bad p-value: 0.77)

Model:	ARIMA(10, 1, 0)	Log Likelihood	4518.069
Method:	css-mle	S.D. of innovations	0.178
Date:	Sat, 19 Dec 2020	AIC	-9012.138
Time:	22:48:51	BIC	-8921.042
Sample:	09-17-1979 - 10-14-2019	HQIC	-8981.878

	coef	std err	z	P>|z|	[0.025	0.975]
const	5.057e-07	0.002	0.000	1.000	-0.003	0.003
ar.L1.D.Streamflow	0.2973	0.008	35.968	0.000	0.281	0.313
ar.L2.D.Streamflow	-0.1160	0.009	-13.454	0.000	-0.133	-0.099
ar.L3.D.Streamflow	0.0085	0.009	0.983	0.326	-0.008	0.026
ar.L4.D.Streamflow	-0.0253	0.009	-2.921	0.003	-0.042	-0.008
ar.L5.D.Streamflow	0.0120	0.009	1.390	0.165	-0.005	0.029
ar.L6.D.Streamflow	-0.0084	0.009	-0.970	0.332	-0.025	0.009
ar.L7.D.Streamflow	-0.0376	0.009	-4.343	0.000	-0.055	-0.021
ar.L8.D.Streamflow	0.0182	0.009	2.104	0.035	0.001	0.035
ar.L9.D.Streamflow	-0.0219	0.009	-2.540	0.011	-0.039	-0.005
ar.L10.D.Streamflow	-0.0024	0.008	-0.289	0.772	-0.019	0.014

ARIMA(15,1,0): (AR 10 to 15 high and bad p-values)

Model:	ARIMA(15, 1, 0)	Log Likelihood	4529.307
Method:	css-mle	S.D. of innovations	0.178
Date:	Sat, 19 Dec 2020	AIC	-9024.614
Time:	22:50:06	BIC	-8895.561
Sample:	09-17-1979 - 10-14-2019	HQIC	-8981.745

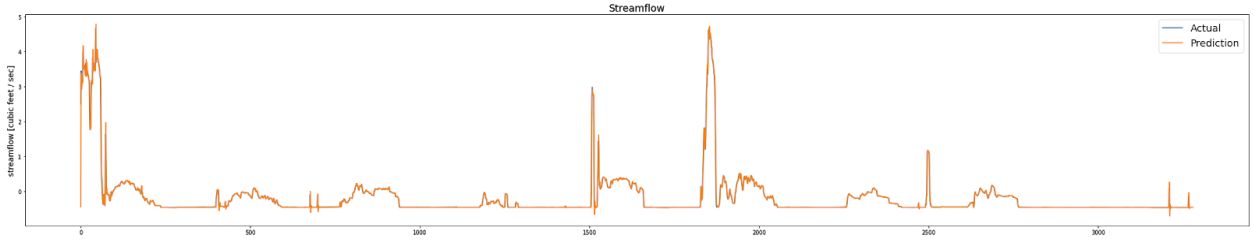
	coef	std err	z	P>|z|	[0.025	0.975]
const	6.73e-07	0.002	0.000	1.000	-0.003	0.003
ar.L1.D.Streamflow	0.2970	0.008	35.938	0.000	0.281	0.313
ar.L2.D.Streamflow	-0.1154	0.009	-13.393	0.000	-0.132	-0.099
ar.L3.D.Streamflow	0.0088	0.009	1.012	0.312	-0.008	0.026
ar.L4.D.Streamflow	-0.0253	0.009	-2.917	0.004	-0.042	-0.008
ar.L5.D.Streamflow	0.0116	0.009	1.341	0.180	-0.005	0.029
ar.L6.D.Streamflow	-0.0079	0.009	-0.908	0.364	-0.025	0.009
ar.L7.D.Streamflow	-0.0388	0.009	-4.480	0.000	-0.056	-0.022
ar.L8.D.Streamflow	0.0179	0.009	2.060	0.039	0.001	0.035
ar.L9.D.Streamflow	-0.0206	0.009	-2.381	0.017	-0.038	-0.004
ar.L10.D.Streamflow	-0.0046	0.009	-0.534	0.594	-0.022	0.012
ar.L11.D.Streamflow	0.0055	0.009	0.635	0.525	-0.011	0.022
ar.L12.D.Streamflow	0.0041	0.009	0.478	0.633	-0.013	0.021
ar.L13.D.Streamflow	0.0101	0.009	1.164	0.244	-0.007	0.027
ar.L14.D.Streamflow	-0.0355	0.009	-4.122	0.000	-0.052	-0.019
ar.L15.D.Streamflow	-0.0053	0.008	-0.642	0.521	-0.021	0.011

Evaluation ARIMA(5,1,0) before and after fire:

	MSE	MAE	R squared	p-value
Before fire	183.917912	72.842713	0.787544	**0.9**
After fire	428.618875	187.210463	0.812068	**0.9**

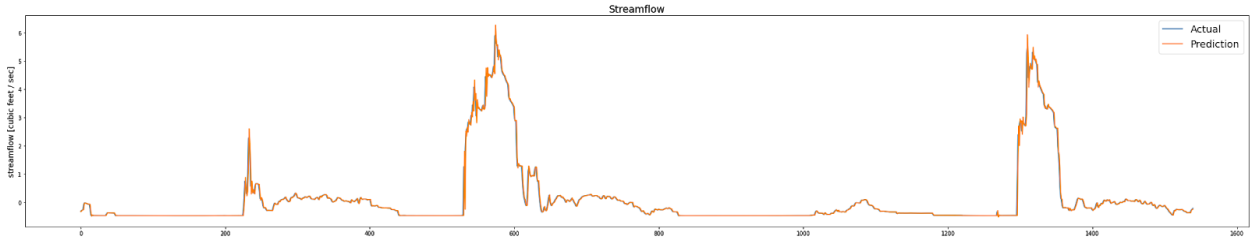
Plots of predictions vs actual values of streamflow before fire:

(Orange: prediction, Blue: actual, almost a match in every point)



Plots of predictions vs actual values of streamflow after fire:

(Orange: prediction, Blue: actual, almost a match in every point)



Conclusion:

We see that even in the ARIMA model, our errors have increased after fire and the R squared decreased after fire. The same pattern we saw in the XGBoost and it is a probable confirmation of our hypothesis that a chaos probably caused by fire disrupted the correct prediction of streamflow.

Approach 2: Bayesian Network

Deconstruction:

Why did we choose Bayesian Network? We used Bayesian Network for the prediction of streamflow fluctuation because the reputable papers we read worked on this approach while working with hydrological series.

A summary of overall steps:

For implementing our bayesian network we followed the exact steps done in a research paper mentioned in the reference section. Primarily, we created a synthetic streamflow time series dataset , implemented apriori to find association patterns , only the highest 2% patterns were selected.

Generating Synthetic Data using ARMA(1,1)

The Autoregressive Moving average model (p,q) predicts values using their previous correlated lags. We have used p=1 which is one year lag. We normalize the dataset because the ARMA model assumes Gaussian distribution for variables. ARMA(1,1) is a good candidate because it has developed with the highest freedom degree and it allows relationships between years (nodes) not to be conditioned before entering the Bayesian network. In this way, it allows the Bayesian network to detect the connections between years itself. 200 time series dataset were generated using this method.

Deconstruction: According to our observation while comparing the difference of aic and bic using arima with different orders ,model selection with any of these orders has

similar performance. Predictions are done recursively. The prediction of one value is put as the input value of the next ARMA prediction. This ARMA usage is not for prediction. Just to generate 200 new time series such that we feed it as data to the bayesian network. Since the Bayesian network is a population based method, we need to generate multiple time series to feed it. Here we have generated 200 time series using this recursive ARMA generation.

Finding Association Patterns of the Years (Nodes of Bayesian Network)

Inorder to construct our Bayesian Network we need to find out the patterns and relation between each node. This step was done by implementing Apriori algorithm on our synthetic annual streamflow dataset.

First we have discretized the annual synthetic time series into 4 categories [0,25%]: 1, [25,50%]: 2, [50%,75%]:3 , [75, 100%]:4. Using Apriori algorithm, association patterns were found and only Only 2% of all the patterns with Lift > 1.2 were chosen.Thus, we were able to find almost 40 patterns and connected the nodes (years) with lift> 1.2

Deconstruction:

Limitation: This may cause an issue later and we keep it in mind that this may not be the best way to discretize the dataset and for future work we plan to find better ways to discretize it.

Assumptions

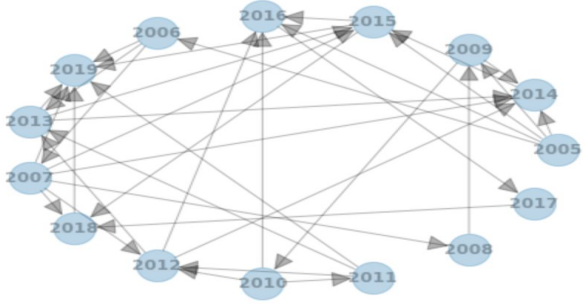
Discretizing into (min , 25%,50%,max) intervals to categorize streamflow values might not be the best discretization. Also, we only connected the highest 2% lift. Maybe this is not enough and we are not capturing enough patterns (not connecting enough nodes).

Deconstruction:

Finding the best criteria to choose how many percent of maximum lifts is appropriate to choose as the most associated patterns is one of our ***challenges***. For future work, we plan to use Markov network instead of Bayesian network so that the model picks the important associations itself and we do not need an outer criteria to choose the association patterns.

Bayesian Network Graph

The Bayesian approach is populated and



trained from 200 synthetic annual streamflow series that were previously generated through an ARMA model. The BN model automatically generates probabilistic distributions of streamflow data for each year, as well as a logic structure, according to its internal dependencies relationships. The dependence between consecutive years seems obvious but the identification of time order dependencies larger than one (non-consecutive years) is less trivial which is found by the Bayesian network.

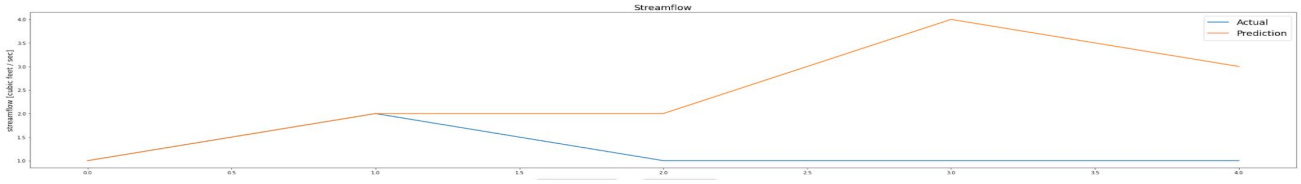
Deconstruction:

Each node represents one year. First we have connected the nodes using the patterns the apriori algorithm introduced to us. Then we had the graph and only needed to find the conditional probabilities. We found the conditional probabilities using Bayes formula and the population of each set. For example conditional probability of value of streamflow in year 2000 being between in its minimum and 25 percentile (category 1) given the last year being in its minimum and 25 percentile (category 1) can be calculated using Bayes formula and the populations of those sets and the population of the intersection of those sets. Our code calculates all conditional probabilities and completes the Bayesian network.

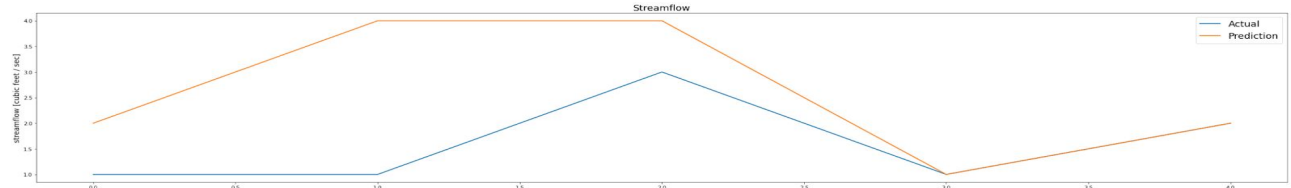
Evaluation Bayesian Network before and after fire

MSE **before** fire	1.673320	MSE **after** fire	1.483240
MAE **before** fire	1.200000	MAE **after** fire	1.000000
R Squared **before** fire	-16.500000	R Squared **after** fire	-2.437500

Line Chart *Before* fire (Predicted: Orange, Blue: Real values):



Line Chart *After* fire (Predicted: Orange, Blue: Real values)



Evaluation using p-value:

*Considering these models are **not** trained in the same circumstances. So at the time of comparison we must note that (comparing apples with apples!)*

- Bayesian network and ARMA use normalized data. XGBoost does't use normalized data.
- Bayesian network uses yearly data, ARMA and XGBoost use daily data.
- Bayesian network uses discretized values. ARMA and XGboost use exact values.
- We used both "Streamflow" and "Precipitation" for XGBoost and only "Streamflow" for ARMA and Bayesian network.

*An addition to **comparison** between the performance of models to the extent that they are comparable and also showing **p-values**:*

Model	XGBoost	Bayesian Network	ARIMA(5,1,0)
Normalization? (yes/no)	No	Yes	Yes
Yearly/Daily	Daily	Yearly	Daily
Discretization	No	Yes	No
Features used	Streamflow/Precipitation	Streamflow	Streamflow
MSE before fire	183.917912	1.673320	0.104759
MAE before fire	72.842713	1.200000	0.022615
R squared before fire	0.787544	-16.500000	0.976682
P-value before fire	3.659856497231594e-10	0.05983787551928	0.9
F before fire	39.40730018050218	4.79999999999999	
MSE after fire	428.618875	1.483240	0.154511

MAE after fire	187.210463	1.000000	0.40402
R squared after fire	0.812068	-2.437500	0.9827679
P-value after fire	4.31760009931759 1e-09	0.20293396786290	0.9
F after fire	34.6744554431153 5	1.92307692307692	

Discussion /Future Work

When using the XGboost method we observed that the MSE, MAE, increased after the fire event. Our observation for the Bayesian network is improvement of errors results after the fire event. Model implemented by XGboost seems to be more predictable before fire whereas Bayesian network is more predictable after the fire event. Xgboost performs significantly better in predicting the streamflow fluctuation in comparison with bayesian networks. We plan to change the Bayesian network to a Markov model so that we wouldn't need to worry about finding a better way to connect the nodes to each other, since the Markov model finds them itself and there will be no need to find a threshold for the lift to make a decision for edges between nodes of network.

Deconstruction:

Poor performance of ARIMA model according to its ***p-value***: If we look at the p-value of each three model, we see that p-value of ARIMA model (0.9) is worse than XGBoost (0.0000) and even the Bayesian network (0.05 and 0.2 after fire).

One of the reasons on why ***ARIMA seems to have lower error value****s in comparison to XGBoost*:

ARIMA uses normalized dataset so the scale of error in predicted values have been removed from the values. This may be one of the reasons that ARIMA has smaller error values. Although it seems to have generally better performance because it has a much higher R squared value (0.97) and seems almost perfect. Also, it shows better performance in terms of peak prediction in comparison with XGBoost.

A probable confirmation of our hypothesis: For all three models, ARIMA, XGBoost, Bayesian network we see a decrease in performance of model after fire. So this is a probable confirmation of our hypothesis that the fire event affects and disrupts the streamflow prediction. We acknowledge that there are some ***limitations*** to this hypothesis. For example, we cannot be sure that this reduction in performance is due to fire specially or that it could be extended to other locations.

Why we chose these three metrics for comparison over a p-value:

MAE, MSE and R-Squared are metrics that were frequently used in almost all the papers we found in analysis of temporal behavior dependencies. For instance, in the article “Innovative Analysis of Runoff Temporal Behavior through Bayesian Networks” ,only these metrics have been used. We Also need to consider the fact that our models are trained in different circumstances as we discussed earlier; they are yearly, daily, normalized or unnormalized. So at the time of comparison we must note that p-value might not make much sense .

References

1. The location not being urban
2. Innovative Analysis of Runoff Temporal Behavior through Bayesian Networks
3. How to Check if Time Series Data is Stationary with Python
4. Salas, J.; Delleur, J.; Yevjevich, V.; Lane, W.L. *Applied Modeling of Hydrologic Time Series*, 1st ed.; Water Resources Publications: Littleton, CO, USA, 1980; p. 484. [**Google Scholar**]
5. Chen, T., and C. Guestrin. "XGBoost: A scalable tree boosting system. arXiv 2016." *arXiv preprint arXiv:1603.02754*.
6. Lingling Ni, Dong Wang, Jianfeng Wu, Yuankun Wang, Yuwei Tao, Jianyun Zhang, Jiufu Liu, Streamflow forecasting using extreme gradient boosting model coupled with Gaussian mixture model, Journal of Hydrology, Volume 586, 2020, 124901, ISSN 0022-1694, https://doi.org/10.1016/j.jhydrol.2020.124901.

Appendix: Augmented Dicky Fuller test

"Augmented Dicky Fuller" test is very similar to "Dicky Fuller" test. So I explain "Dicky Fuller" test because it is simpler and then I write the mode improved one "Augmented Dicky Fuller" test that we have used in our final project.

Dicky Fuller Test

The test is for checking if a time series is stationary or non-stationary assuming that our time series is only dependent of its previous lag (only first order). In the "Augmented" version which we brought later we discuss the same test in presence of higher order dependencies (depending on more than one previous lag). We test a null hypothesis: that our time series is non-stationary. The idea with the test is that we start with an first order AR (Autoregressive) process so we have

$$Y_t = \alpha + \rho Y_{t-1} + \epsilon_t$$

We try to estimate ρ using a least squares method for regression.

Here Xt is the value of time series in time t. And epsilon is the error process. The null hypothesis here is

$$H_0 : \rho = 1$$

Because if this value equals to 1 we have a non-stationary time series.

And the alternative hypothesis is

$$H_1 : \rho < 1$$

Because if this value is less than one, we have proved that our time series or the AR process is stationary. We can write the original equation in this way

$$Y_t - Y_{t-1} = \alpha + (\rho - 1)Y_{t-1} + \epsilon_t$$
$$\Delta Y_t = \alpha + \delta Y_{t-1} + \epsilon_t$$

where

$$\delta = \rho - 1$$

If our null hypothesis happens and we have $H_0 : \rho = 1$ this means that $\delta = 0$ and the equation becomes $\Delta Y_t = \alpha + \epsilon_t$. Now the left hand side is stationary and we don't have non-stationary variables on the right hand side. On the other hand, if $H_1 : \rho < 1$ then $\delta \neq 0$ and we have $\Delta Y_t = \alpha + \delta Y_{t-1} + \epsilon_t$ and we have a non-stationary term in our right hand side.

How do we check whether we have a unit root (because a unit root is when we have $\rho = 1$ then $\delta = 0$)?

We calculate an ordinary t-statistic on δ in $\Delta Y_t = \alpha + \delta Y_{t-1} + \epsilon_t$. Or more specifically we calculate a t-statistic on estimated value of δ which we can denote with $\hat{\delta}$. And then we compare that t-statistic with a t-distribution, that would help us to determine if we had a stationary or a non-stationary time series.

If $\rho < 1$ and $\delta \neq 0$ and the format of equation is $\Delta Y_t = \alpha + \delta Y_{t-1} + \epsilon_t$, (the time series is non-stationary), then the central limit theorem for large values doesn't work and we cannot do this test for asymptotic values. To solve this issue, "Dickey" and "Fuller" made a table for the asymptotic distribution of the least squares estimator for δ under the null hypothesis of being a unit root. We can compare our ordinary t-statistic with the values of this table (which they called it the Dicky-Fuller) distribution and if the t is less than a value coming from Dickey-Fuller table called "DF critical value" then we reject the null hypothesis.

The "Augmented" Dickey-Fuller test

Very similar to the previous Dickey Fuller test but instead of using *first order AR* here we use *higher order AR processes*.

In the previous Dickey- Fuller test we ran a regression on the first order AR process $\Delta y_t = \alpha + \delta Y_{t-1} + \epsilon_t$ and the null hypothesis was $\delta = 0$ against the alternative $\delta < 0$. Now we want to test for the unit root in the presence of higher order dependencies. This is how the equation looks like in the presence of higher order dependencies. We do the regression for this equation

$$\Delta Y_t = \alpha + \delta Y_{t-1} + \sum_{i=1}^{p} \beta_i \Delta Y_{t-i} + \epsilon_t$$

The rest is the same. Null hypothesis is $H_0 : \delta = 0$ and time series is non-stationary. And the alternative is that we have a stable order p AR process when $\delta < 0$ and our time series is stationary.