Predictive Modeling of Harmful Algal Blooms in the Western Basin of Lake Erie

Melanie Butler - Springboard Data Science Capstone

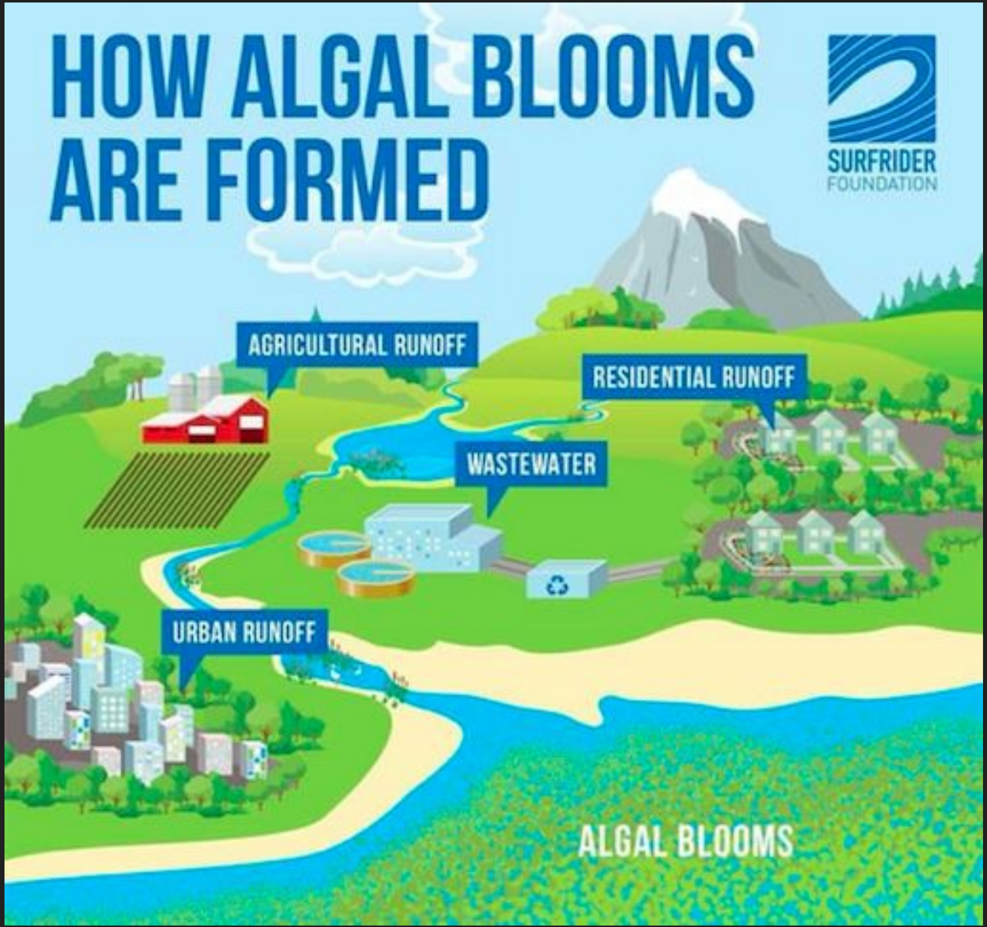
Google Slide Deck

Harmful Algal Blooms (HAB)

- 2014 - 400,000 people in Lake Erie region without drinking water for days
- Toledo - $3 mil to $4 mil annually to treat contaminated water

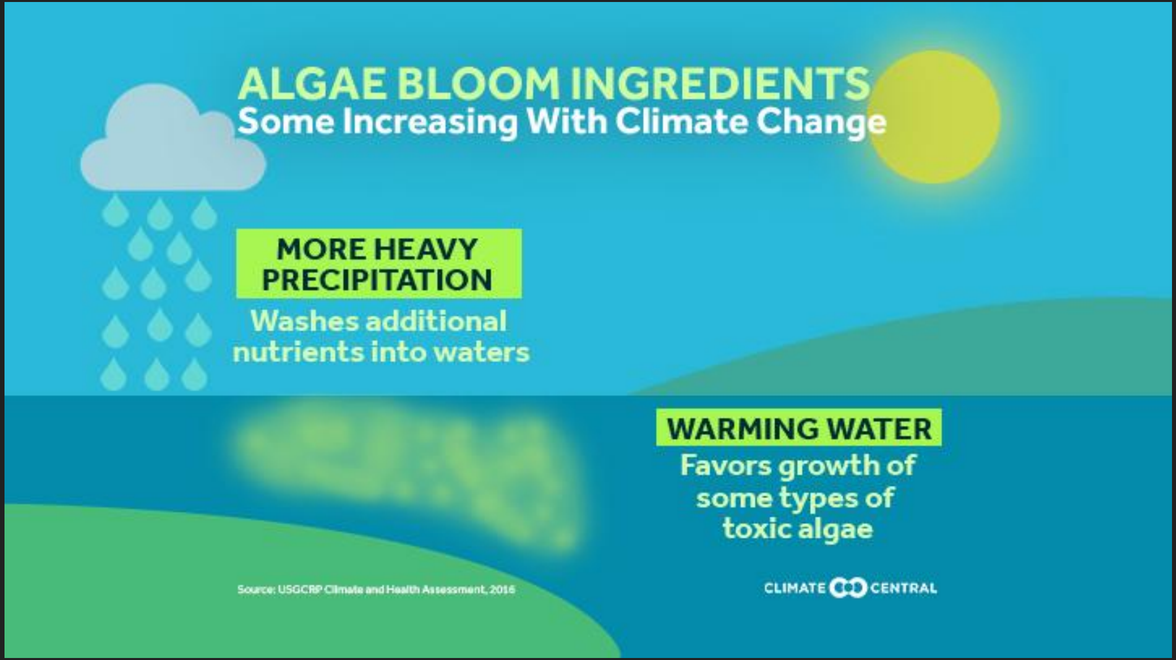
HAB - Causes

- Nitrogen
- Phosphorus
- Oxygen
- Warm Temperatures



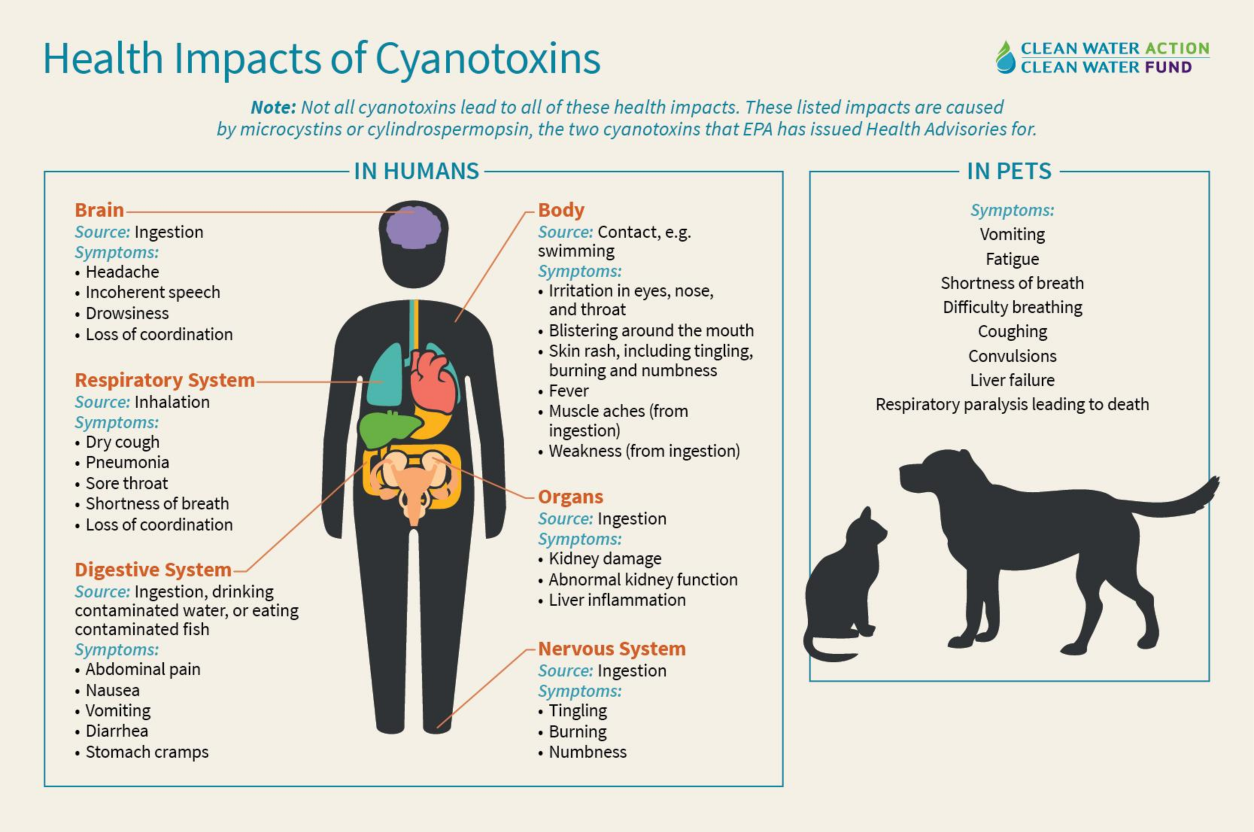
HAB - Causes

- Nitrogen
- Phosphorus
- Oxygen
- Warm Temperatures



HAB Effects

1. Consume oxygen - eutrophication
2. Produce toxins - microcystin



Lake Erie

Drinking water for 11-12 million

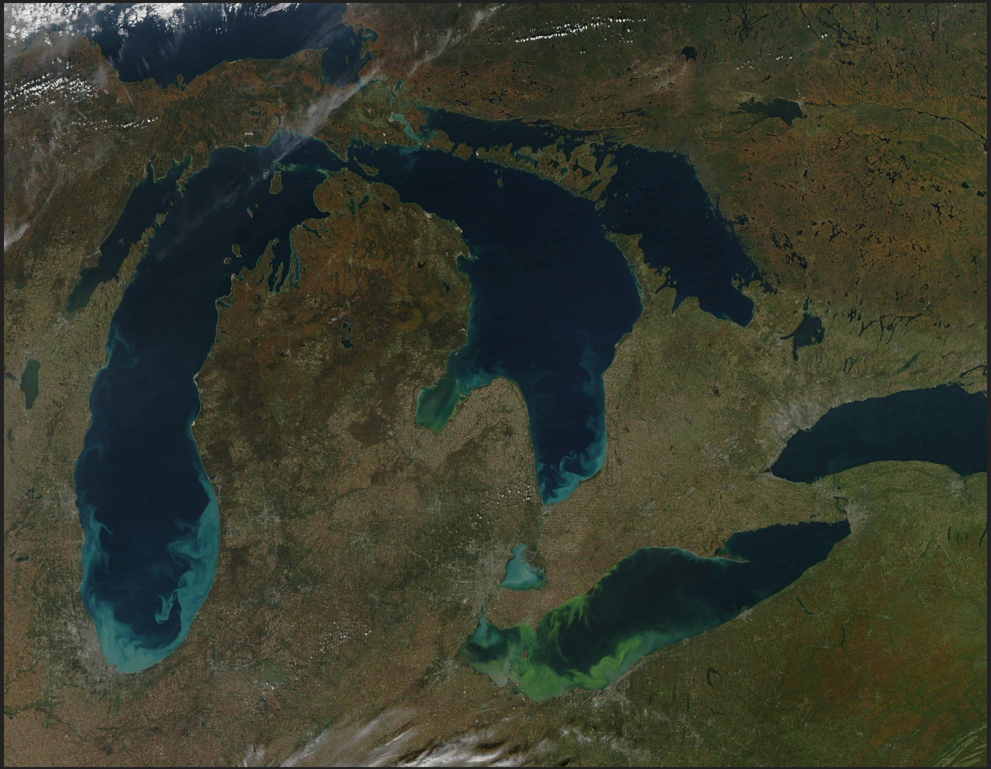
Shallow - Warming

Nutrient Influx -

Agricultural Runoff

Industrial Loading

Invasive Species



https://www.epa.gov/greatlakes/lake-erie#:~:text=Excessive%20algal%20growth%20in%20Lake,in%20the%20U.S.%20and%20Canada.

Guiding Questions

1. What is the typical spatial and temporal profile of nutrients and conditions in Lake Erie?
2. Which water quality features are most predictive of an HAB event?
3. What are the capabilities and limitations of a predictive model for microcystin concentration in Lake Erie?

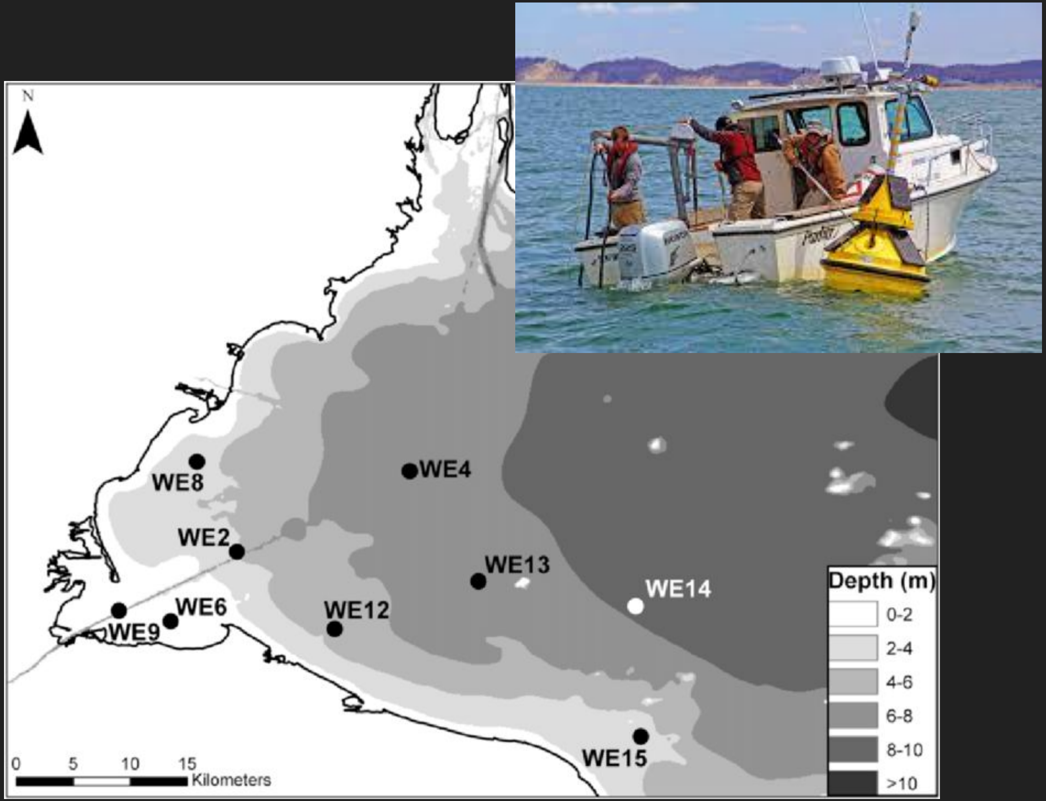
Data Source

Great Lakes Environmental Research Laboratory (GLERL)

Bi-weekly / weekly sampling

Spring - late summer

2012 - 2019



https://www.glerl.noaa.gov/res/HABs_and_Hypoxia/
https://www.researchgate.net/publication/331230140_Spatial-temporal_variability_of_in_situ_cyanobacteria_vertical_structure_in_Western_Lake_Erie_Implications_for_remote_sensing_observations

Data Wrangling Challenges

Spatial Variability:

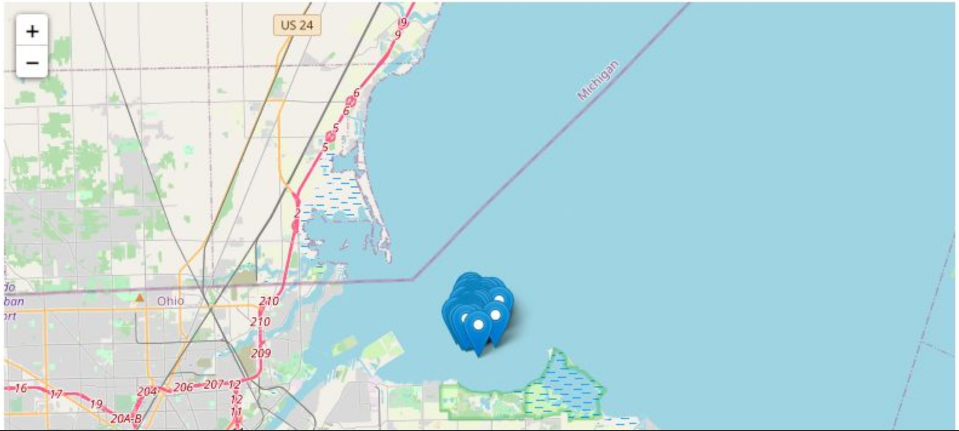
Inconsistent site labels

Solution:

Coordinate filtering and relabeling

```python
#Select closely grouped 'WE6' observations
we6 = surface[(surface.Site == 'WE6') & (surface['Latitude (decimal deg)'] < 41.79)
              & (surface['Longitude (decimal deg)'] > -83.45)]
```

```python
#Confirm that measurements are within one area
mapsites(we6)
```

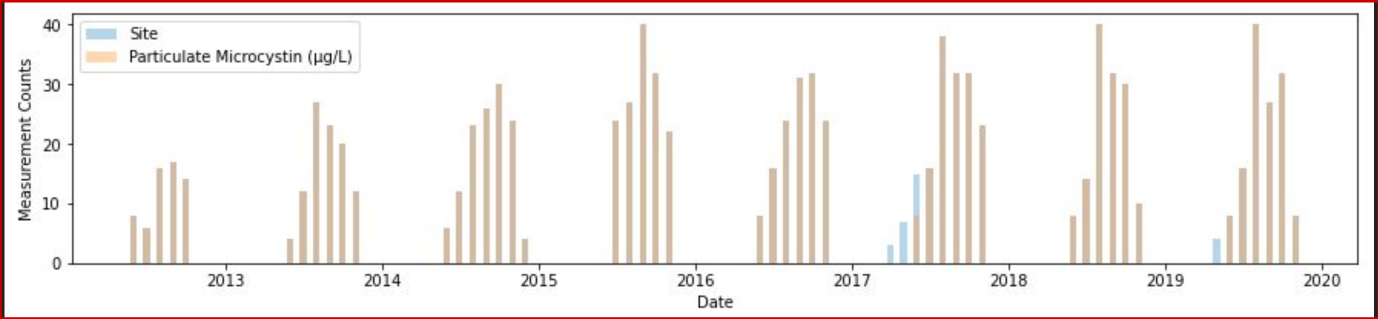


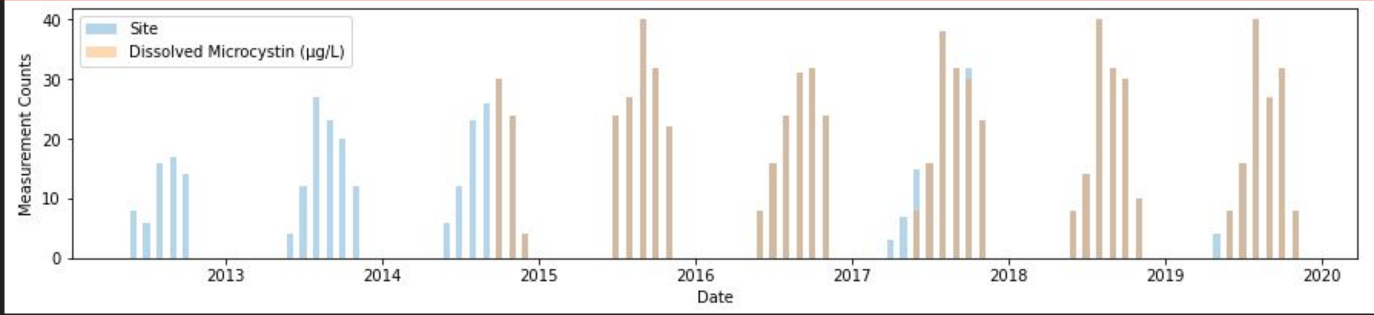
Data Wrangling Challenges

Particulate Microcystin data more complete than dissolved > PM will be target model output

Temporal Variability:

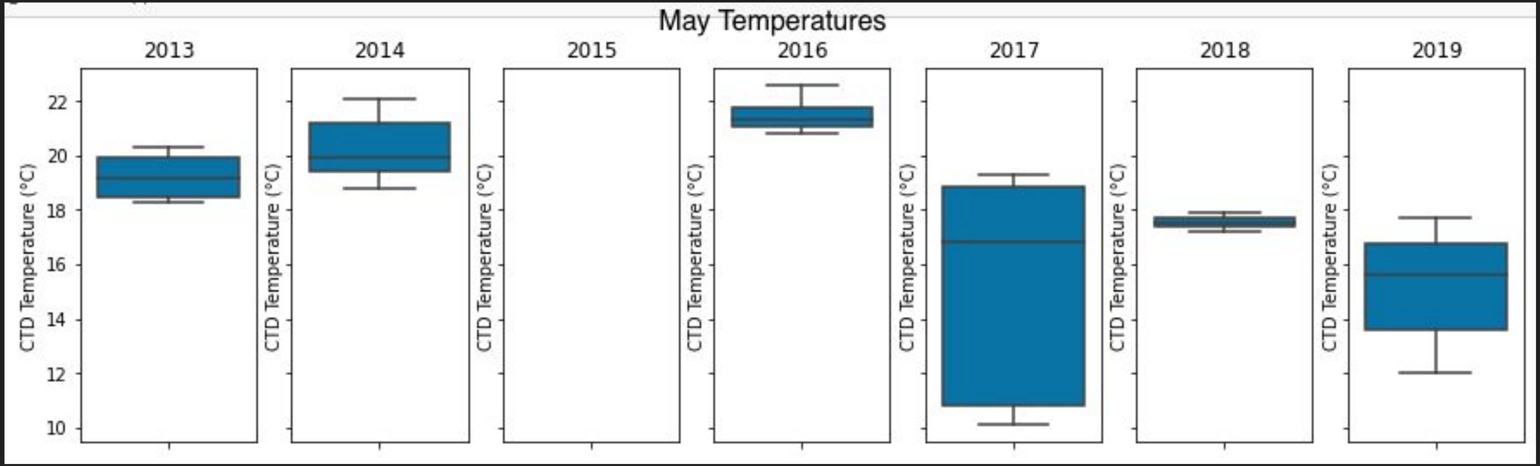
Irregular sampling and missing values





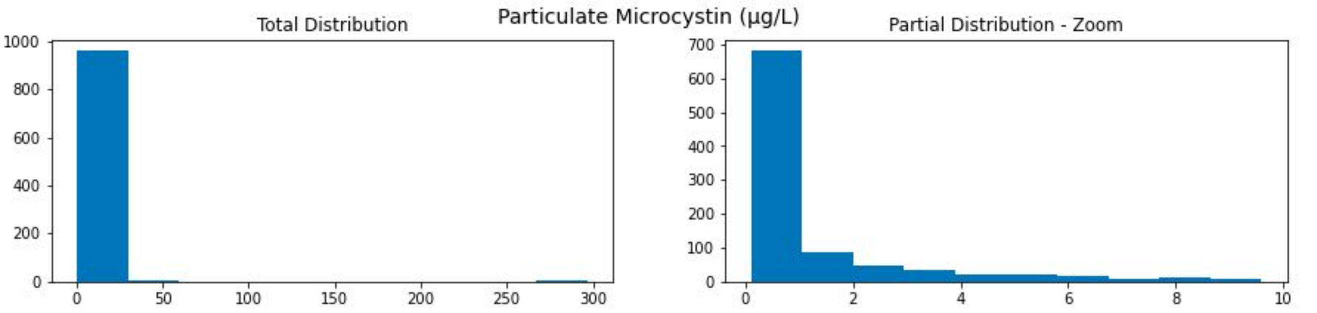
Data Wrangling Challenges

Solution: Impute using year/month average, linear interpolation (more in feature engineering)

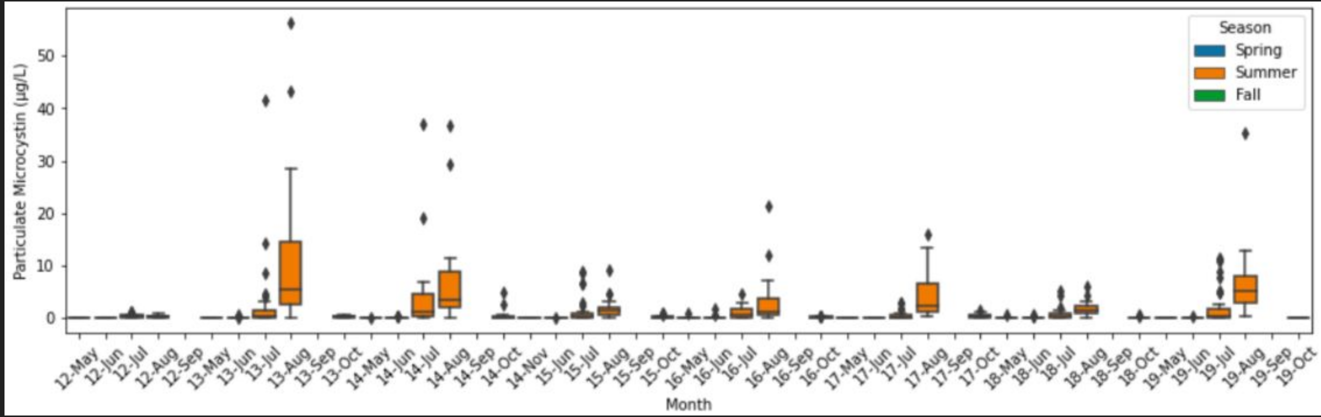


EDA - Particulate Microcystin

Usually below 1.6μg/L drinking limit



Higher in late summer months

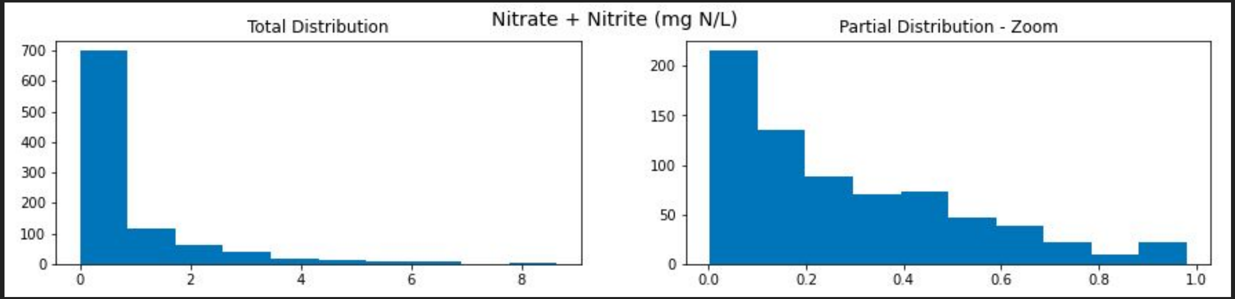


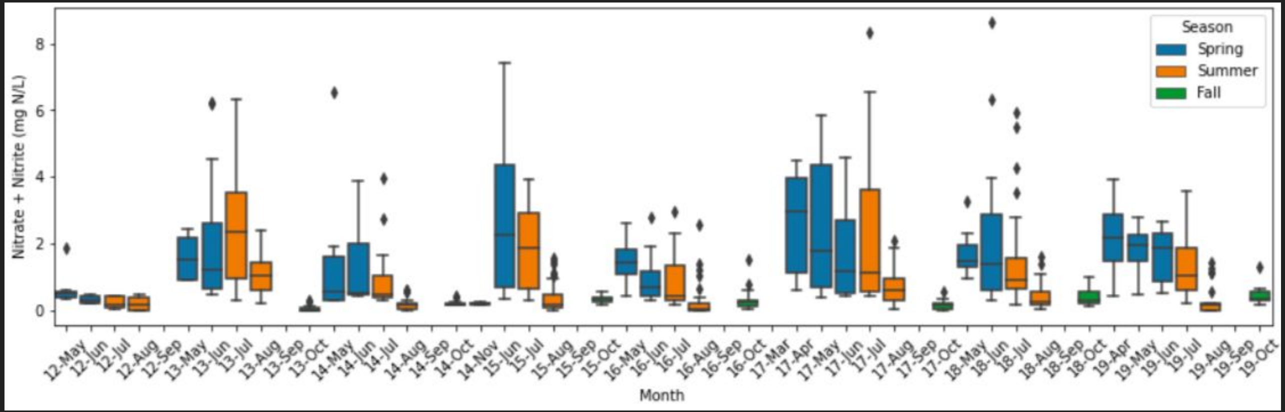
EDA - Nitrogen, Nitrites, Nitrates

Higher / more variable in spring months

Rainfall → nutrient runoff

Late summer → algae consume

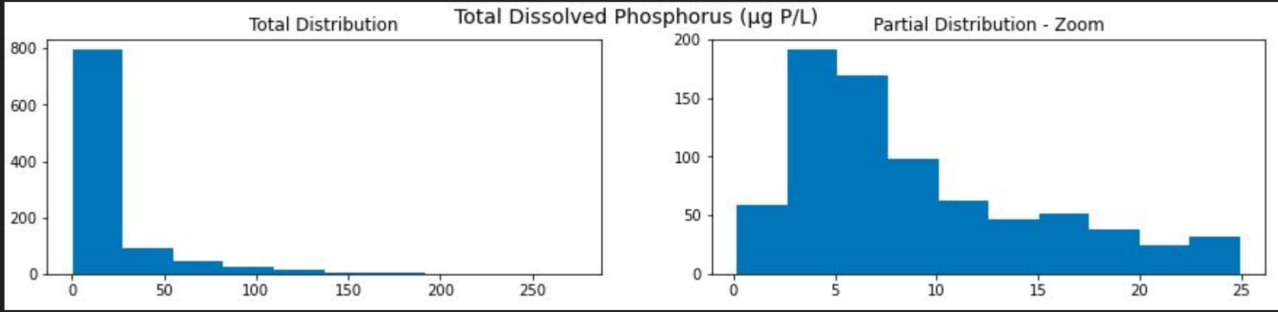


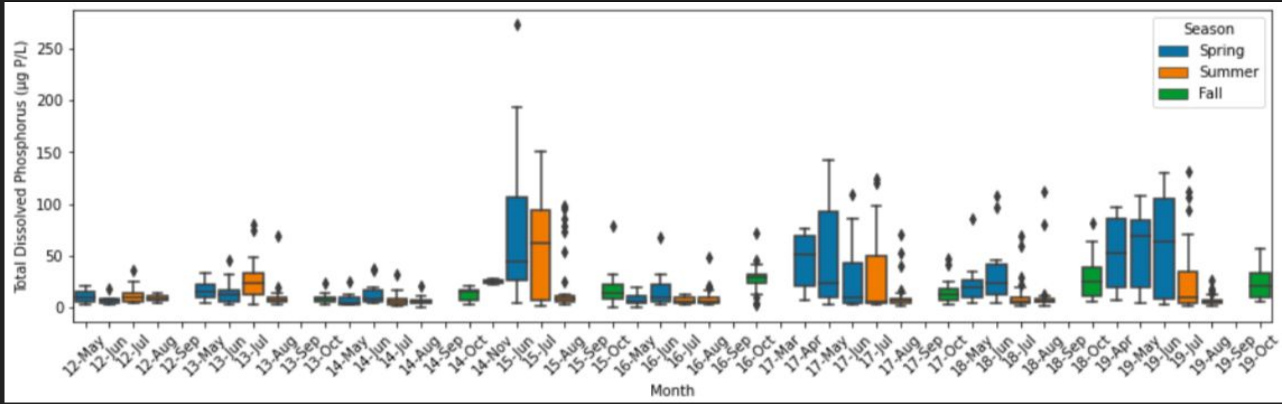


EDA - Phosphorus-containing Compounds

Similar pattern to nitrogen

Spring loading → Summer / fall consumption?

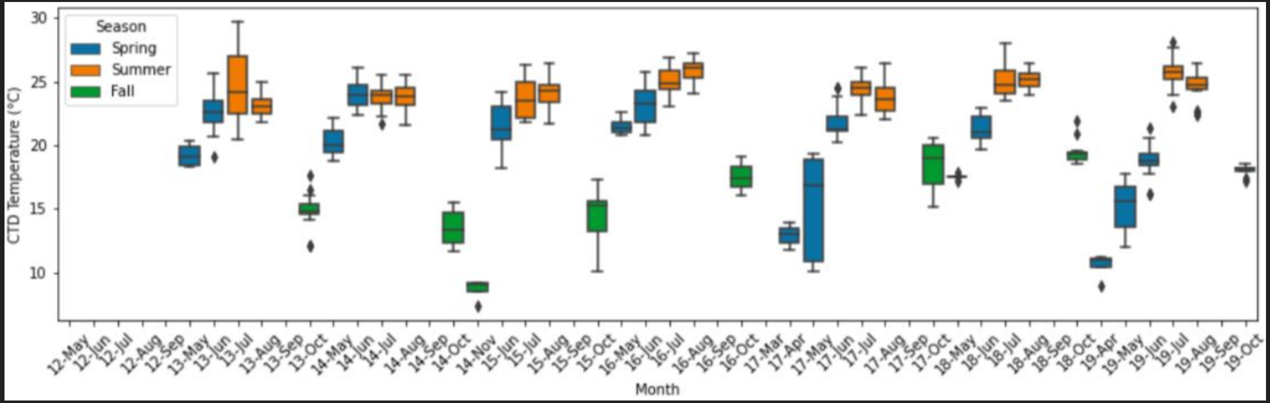


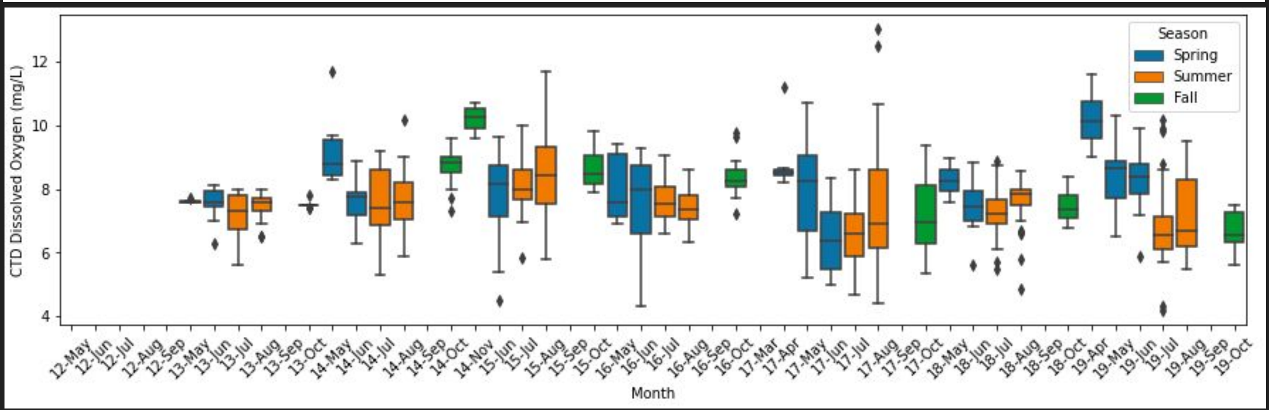


EDA - Physical Factors

Temp: as expected (upper limit around 26°C)

Dissolved Oxygen: low point late summer → HAB consumption?

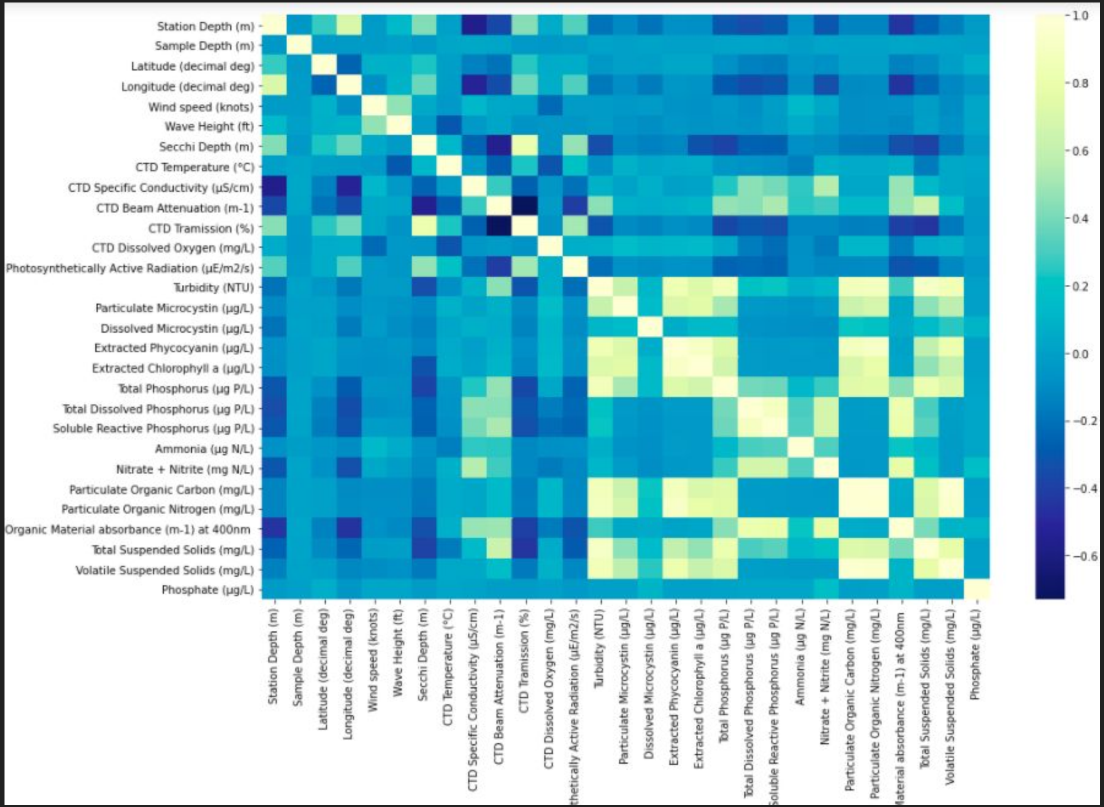




EDA - Pearson Correlations

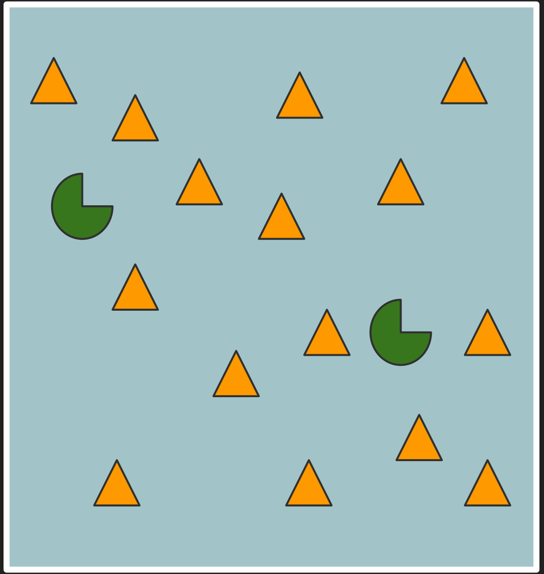
Many logical correlations (e.g. transmission / beam attenuation)

No strong correlation between nutrients and microcystin!

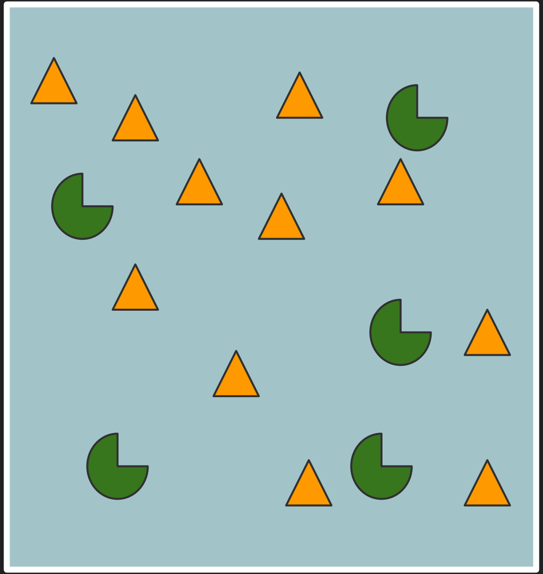


Where are the environmental correlations? Hypothesis:

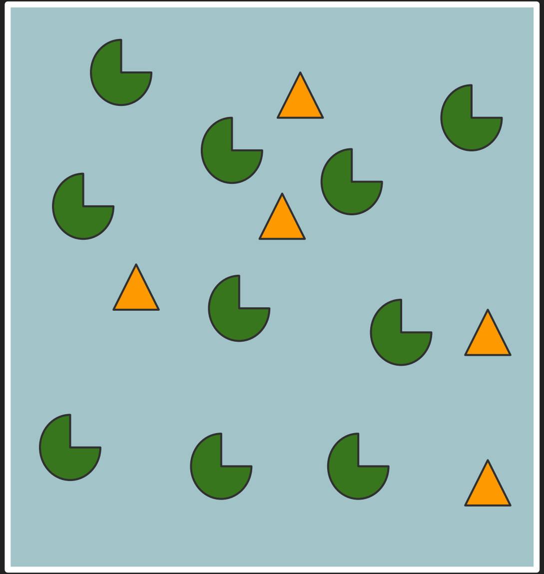
Early in the spring - cold → low algae rain → nutrient runoff



Warmer weather, nutrient-rich environs → algae grow, consume



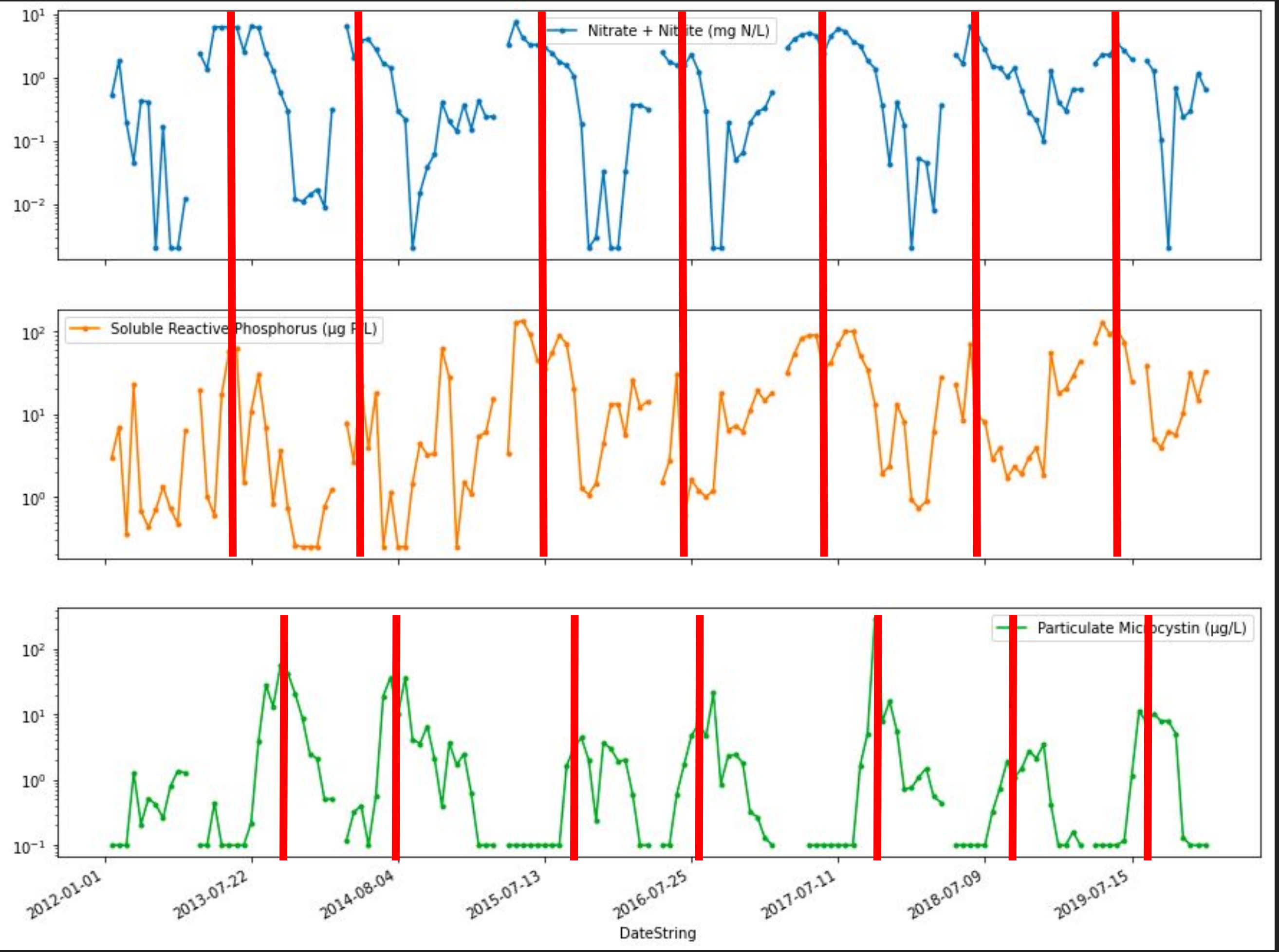
Continued, uncontrolled growth → HAB, nutrient consumption



Time Lag?

Apparent offset between nutrient peaks and particulate microcystin peaks

Magnitude - 3 to 8 weeks



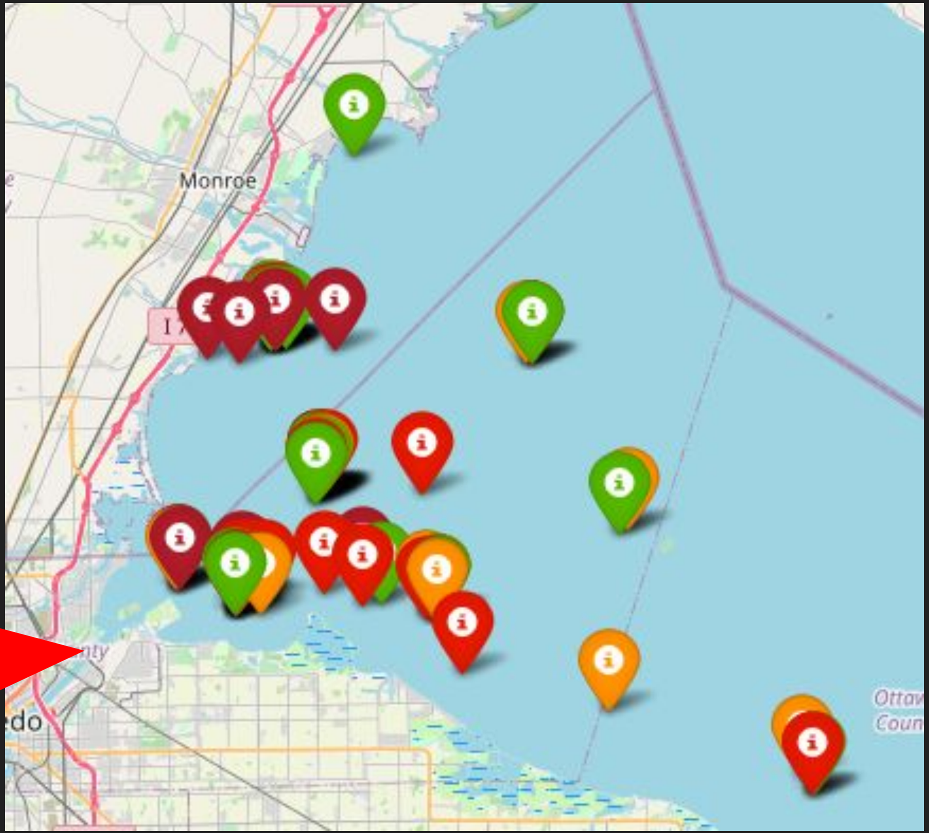
Spatial Correlations

Dark red - top quartile microcystin

Green - bottom quartile microcystin

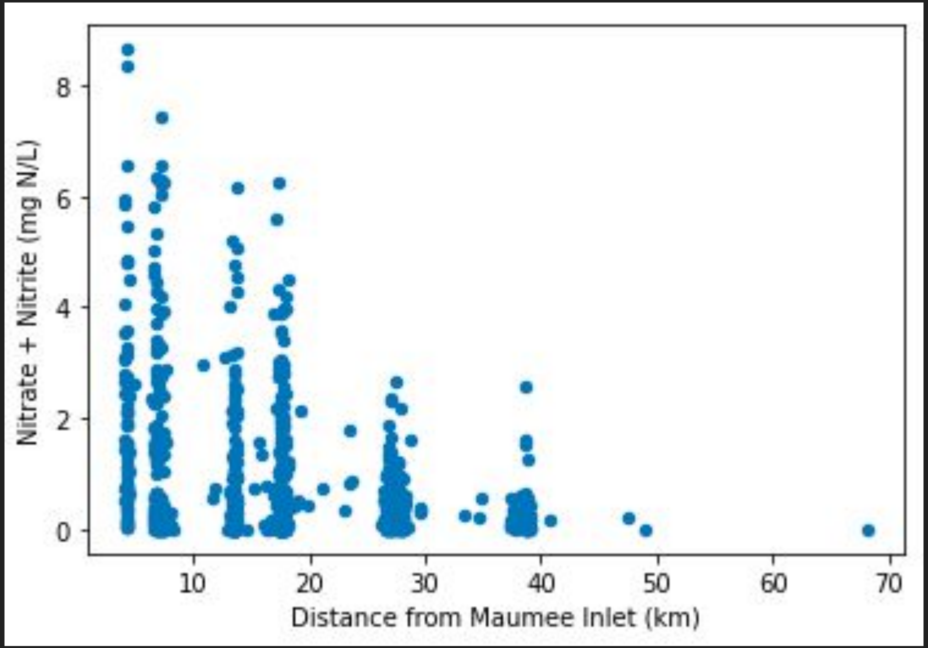
More concentrated measurements:

1. Close to shore
2. Close to Maumee Inlet



Spatial Correlations

- ➢ Shallower
- ➢ Warmer
- ➢ Less diluted runoff
- ➢ Nutrient loading from river



Feature Engineering

Need to create time-shifted features:

For each feature, observation should include feature 2wks prior, 4wks prior, etc.

Challenge: irregular sampling intervals.

Solution: resampling and linear interpolation

```
interpolated_features = pd.DataFrame()
cols = ['CTD Temperature (°C)', 'CTD Dissolved Oxygen (mg/L)',
       'Total Dissolved Phosphorus (µg P/L)','Ammonia (µg N/L)',
        'Total Phosphorus (µg P/L)', 'Soluble Reactive Phosphorus (µg P/L)',
        'Nitrate + Nitrite (mg N/L)', 'N:P Mass Ratio']
shift_dict = {'2wks':14, '4wks':28, '6wks':42}


for site_name in site_names:
    df = features[features['Site']==site_name].set_index('Date', drop=True)
    site_interpolated = pd.DataFrame()

    #resample daily and perform linear interpolation per site, per year
    for year in df.index.year.unique():
        temp = df.loc[df.index.year==year].resample('D').interpolate(method='linear')

        #for each relevant column create a new column shifted by the periods defined in shift_d
        for col in cols:
            for key in shift_dict.keys():
                periods=shift_dict[key]
                temp[col+'_'+key] = temp[col].shift(periods=periods).copy().fillna(method='bfil
        temp = temp.resample('W').mean()

        #once finished interpolating and shifting for one year, concatenate to site_interpolate
        site_interpolated = pd.concat([site_interpolated,temp])

    site_interpolated['Site'] = site_name

    #concatenate finished site df to final df
    interpolated_features = pd.concat([interpolated_features, site_interpolated])

#fill remaining null values with median
interpolated_features.fillna(interpolated_features.median(), inplace=True)
```

Feature Engineering

Select → one site, one year.

Upsample → daily, linear interpolation.

```
interpolated_features = pd.DataFrame()
cols = ['CTD Temperature (°C)', 'CTD Dissolved Oxygen (mg/L)',
       'Total Dissolved Phosphorus (µg P/L)','Ammonia (µg N/L)',
        'Total Phosphorus (µg P/L)', 'Soluble Reactive Phosphorus (µg P/L)',
        'Nitrate + Nitrite (mg N/L)', 'N:P Mass Ratio']
shift_dict = {'2wks':14, '4wks':28, '6wks':42}

for site_name in site_names:
    df = features[features['Site']==site_name].set_index('Date', drop=True)
    site_interpolated = pd.DataFrame()

    #resample daily and perform linear interpolation per site, per year
    for year in df.index.year.unique():
        temp = df.loc[df.index.year==year].resample('D').interpolate(method='linear')

        #for each relevant column create a new column shifted by the periods defined in shift_d
        for col in cols:
            for key in shift_dict.keys():
                periods=shift_dict[key]
                temp[col+'_'+key] = temp[col].shift(periods=periods).copy().fillna(method='bfil
        temp = temp.resample('W').mean()

        #once finished interpolating and shifting for one year, concatenate to site_interpolate
        site_interpolated = pd.concat([site_interpolated,temp])

    site_interpolated['Site'] = site_name

    #concatenate finished site df to final df
    interpolated_features = pd.concat([interpolated_features, site_interpolated])

#fill remaining null values with median
interpolated_features.fillna(interpolated_features.median(), inplace=True)
```

Feature Engineering

Select → one site, one year.

Upsample → daily, linear interpolation.

Create new time-shifted features → 2wks, 4wks, 6wks

```
interpolated_features = pd.DataFrame()
cols = ['CTD Temperature (°C)', 'CTD Dissolved Oxygen (mg/L)',
        'Total Dissolved Phosphorus (µg P/L)','Ammonia (µg N/L)',
         'Total Phosphorus (µg P/L)', 'Soluble Reactive Phosphorus (µg P/L)',
         'Nitrate + Nitrite (mg N/L)', 'N:P Mass Ratio']
shift_dict = {'2wks':14, '4wks':28, '6wks':42}


for site_name in site_names:
    df = features[features['Site']==site_name].set_index('Date', drop=True)
    site_interpolated = pd.DataFrame()

    #resample daily and perform linear interpolation per site, per year
    for year in df.index.year.unique():
        temp = df.loc[df.index.year==year].resample('D').interpolate(method='linear')

        #for each relevant column create a new column shifted by the periods defined in shift_d
        for col in cols:
            for key in shift_dict.keys():
                periods=shift_dict[key]
                temp[col+'_'+key] = temp[col].shift(periods=periods).copy().fillna(method='bfil
        temp = temp.resample('W').mean()

        #once finished interpolating and shifting for one year, concatenate to site_interpolate
        site_interpolated = pd.concat([site_interpolated,temp])

    site_interpolated['Site'] = site_name

    #concatenate finished site df to final df
    interpolated_features = pd.concat([interpolated_features, site_interpolated])

#fill remaining null values with median
interpolated_features.fillna(interpolated_features.median(), inplace=True)
```

Feature Engineering

Select → one site, one year.

Upsample → daily, linear interpolation.

Create new time-shifted features → 2wks, 4wks, 6wks

Resample to weekly frequency, recombine data

```
interpolated_features = pd.DataFrame()
cols = ['CTD Temperature (°C)', 'CTD Dissolved Oxygen (mg/L)',
       'Total Dissolved Phosphorus (µg P/L)','Ammonia (µg N/L)',
        'Total Phosphorus (µg P/L)', 'Soluble Reactive Phosphorus (µg P/L)',
        'Nitrate + Nitrite (mg N/L)', 'N:P Mass Ratio']
shift_dict = {'2wks':14, '4wks':28, '6wks':42}


for site_name in site_names:
    df = features[features['Site']==site_name].set_index('Date', drop=True)
    site_interpolated = pd.DataFrame()

    #resample daily and perform linear interpolation per site, per year
    for year in df.index.year.unique():
        temp = df.loc[df.index.year==year].resample('D').interpolate(method='linear')

        #for each relevant column create a new column shifted by the periods defined in shift_d
        for col in cols:
            for key in shift_dict.keys():
                periods=shift_dict[key]
                temp[col+'_'+key] = temp[col].shift(periods=periods).copy().fillna(method='bfil
        temp = temp.resample('W').mean()

        #once finished interpolating and shifting for one year, concatenate to site_interpolate
        site_interpolated = pd.concat([site_interpolated,temp])

    site_interpolated['Site'] = site_name

    #concatenate finished site df to final df
    interpolated_features = pd.concat([interpolated_features, site_interpolated])

#fill remaining null values with median
interpolated_features.fillna(interpolated_features.median(), inplace=True)
```

Feature Engineering

New Feature:

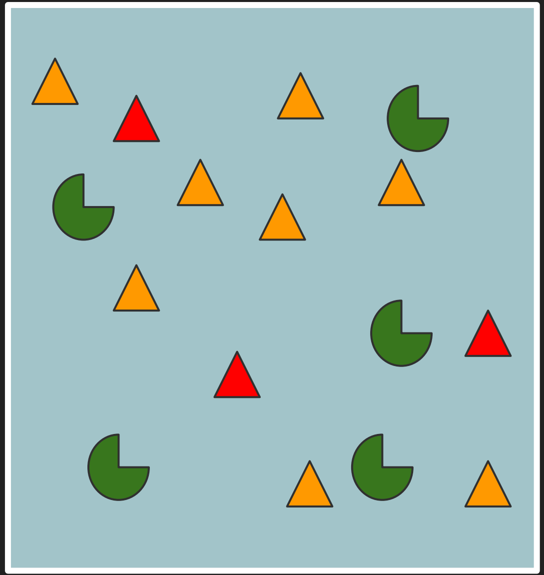
N:P mass ratio

Nitrogen:

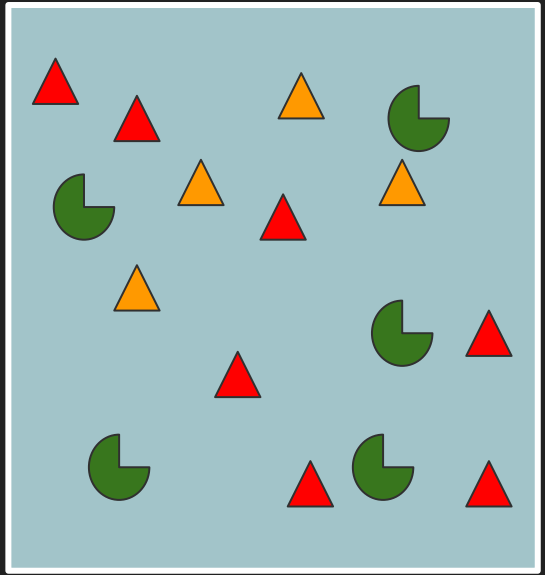
Phosphorus:

Algae:

Enough P, not enough N
→ nitrogen-limited
→ low N:P ratio



Enough N, not enough P
→ phosphorus-limited
→ high N:P ratio



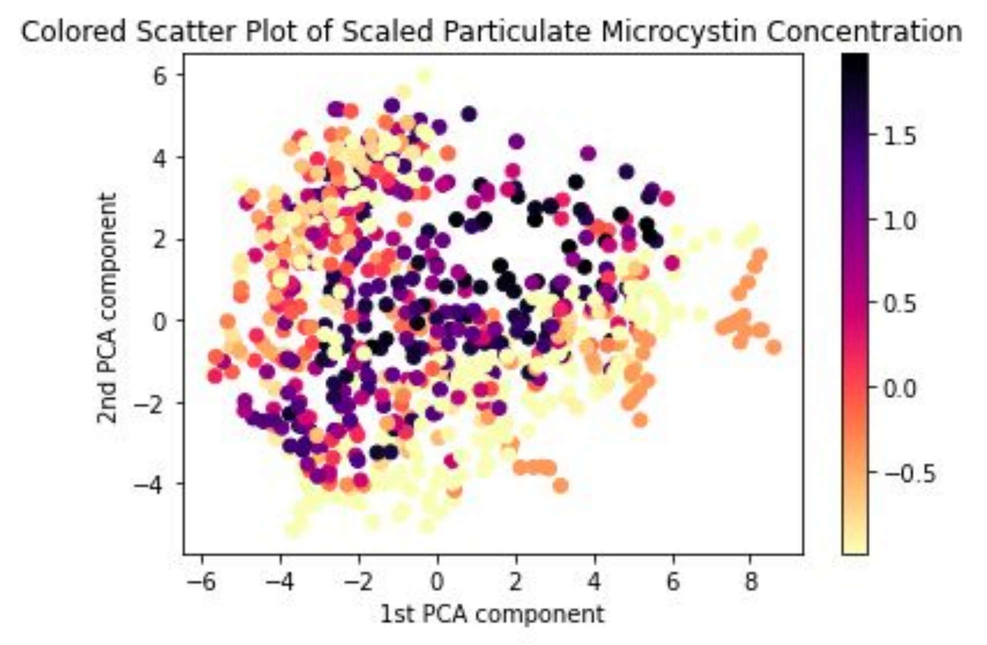
Feature Engineering

Selected Features:

- ➢ Distance from shore, inlet
- ➢ Month label
- ➢ Nutrients (NOx, Total / Dissolved Phosphorus, Ammonia)
- ➢ Temperature, Dissolved Oxygen

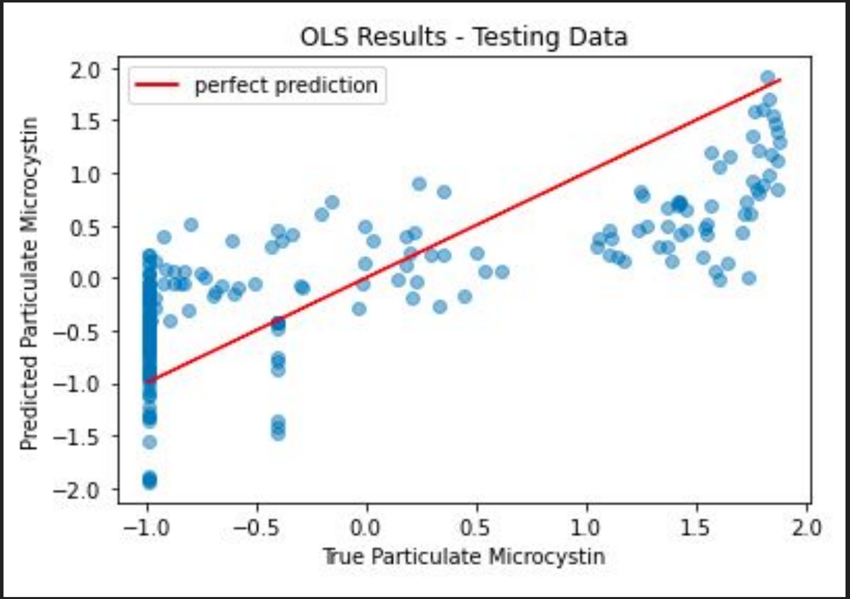
Original and power transformed data

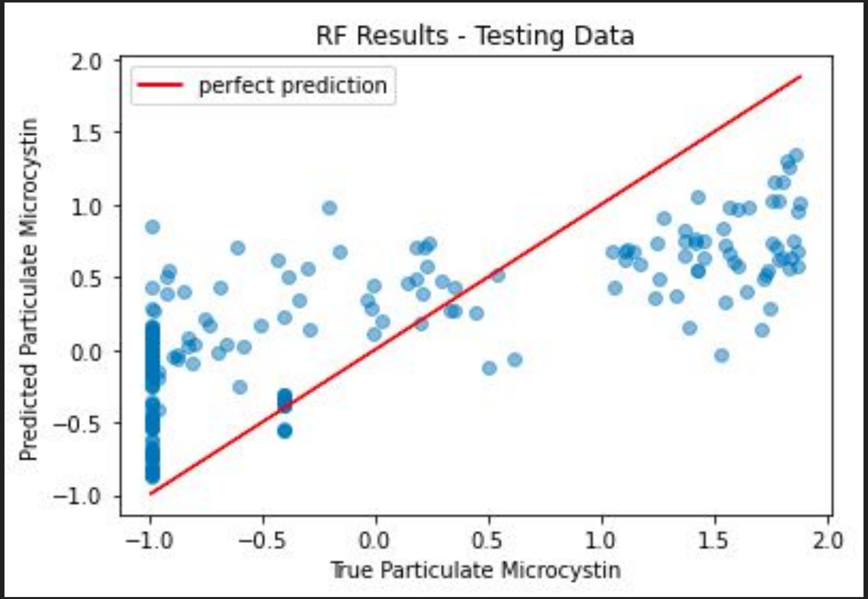
PCA plot shows some separation > selected features should have some predictive ability



Modeling - Regression

OLS, RF, XGBoost - Poor performance of optimized models. Could not distinguish high / low concentration.





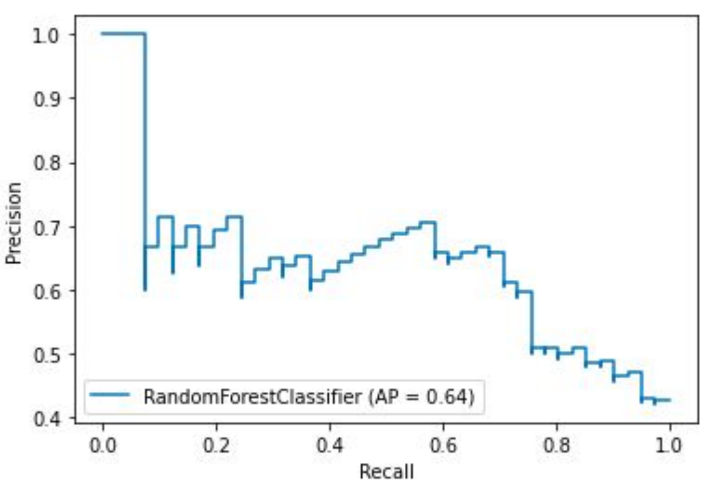
Modeling - Classification

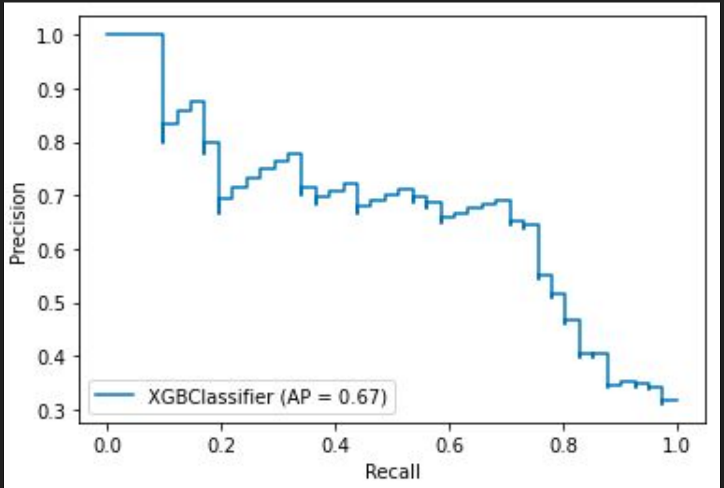
EPA drinking limit (1.6µg/L)

- ➢ Positive class weight 0.2
- ➢ Hyperparameter tuning with balanced accuracy scoring (RF and XGB)
- ➢ Time series-split for CV and train / test split

Performance metric: **Recall**

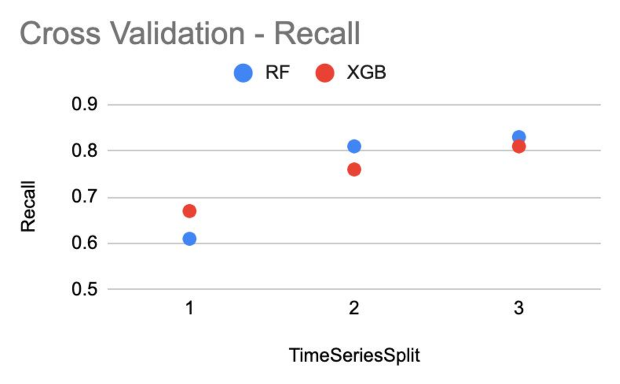
- ➢ False positive = annoyance
- ➢ False negative = missed HAB
- ➢ **Threshold tuning to maximize recall with reasonable precision**

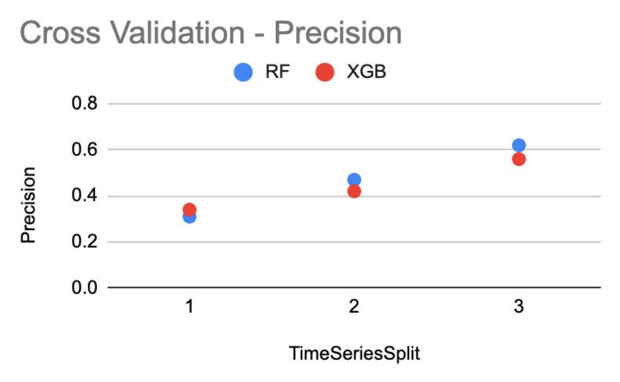




Modeling - Classification Results

Classifier	Precision - Test Set	Recall - Test Set
Random Forest - default threshold	0.79	0.52
XGBoost - default threshold	0.72	0.54
Random Forest - adjusted threshold	0.69	0.71
XGBoost - adjusted threshold	0.65	0.71

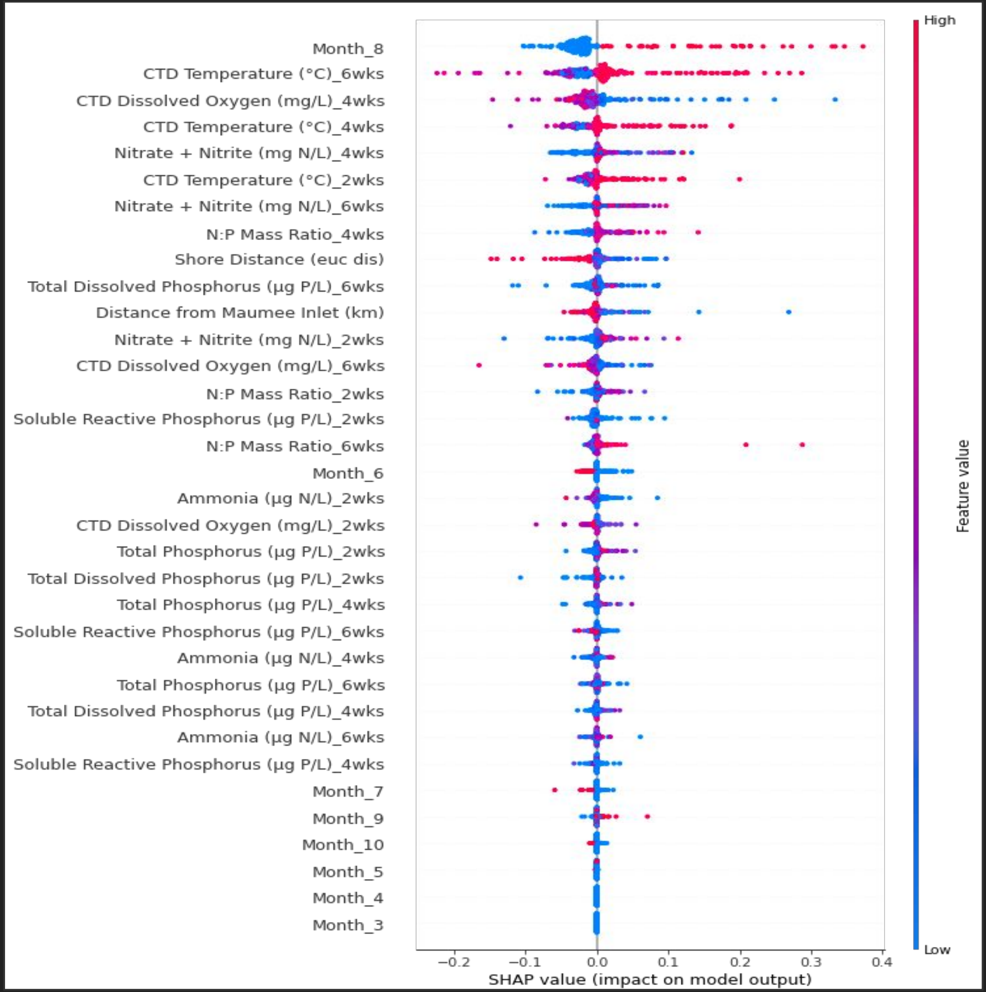




Feature Importance

Optimized RF model SHAP value plots:

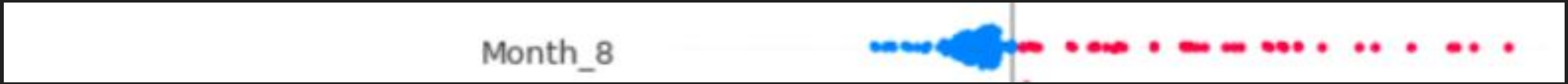
- Dot represents feature importance for one sample
- Right → feature increased probability of positive label
- Left → feature decreased probability of positive label
- Red → high feature value
- Blue → low feature value



Feature Importance

Month -

HAB much more likely in August, September to lesser extent.



Temperature -

Higher temp, HAB more likely, *especially many weeks prior*.

Feature Importance

Dissolved Oxygen -

Low oxygen increases probability of HAB, *especially in prior weeks* - eutrophication due to algae consuming oxygen?

Location -

Close to shore, Maumee increases probability of HAB

Feature Importance

Nitrogen / N:P Ratio -

Higher nutrient concentration in past → greater probability of HAB

Earlier measurements more influential

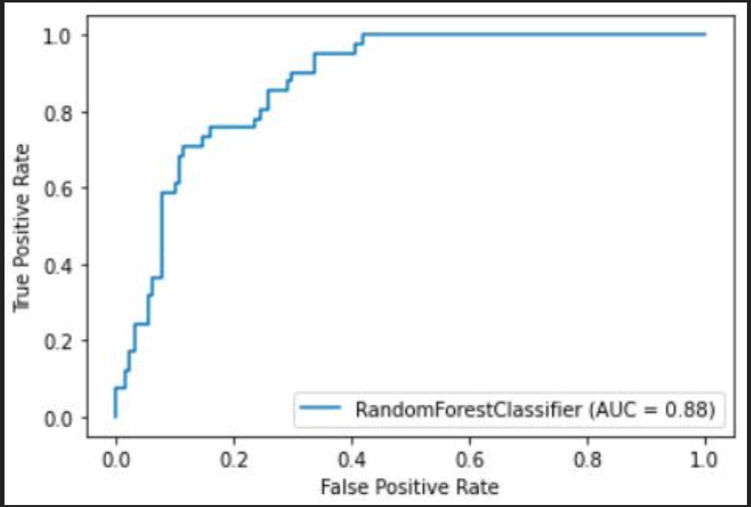
More important than phosphorus

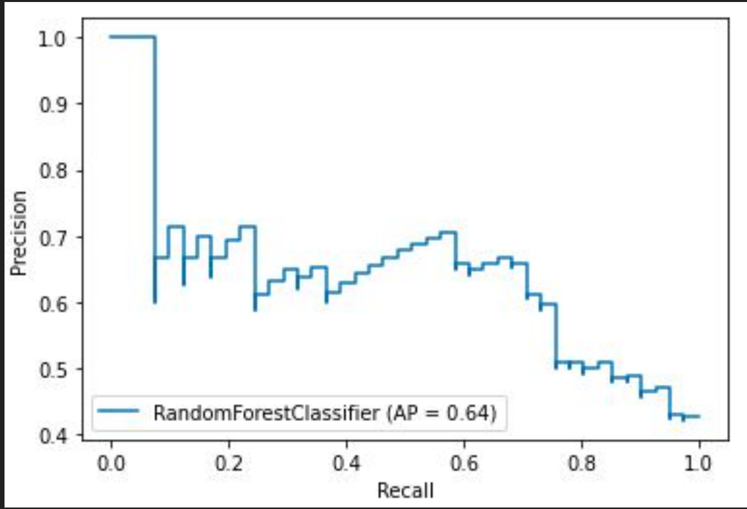
Higher N:P → HAB. Does this imply nitrogen-limited system?

Model Recommendation

Random Forest Binary Classifier with tuned threshold

- ➢ 60% - 70% Recall
- ➢ 60% - 70% Precision
- ➢ Promising considering restriction of features
- ➢ Early warning - treatment with carbon, alum chlorine to remove microcystin





https://cen.acs.org/articles/92/i32/Danger-Microcystins-Toledo-Water-Unclear.html

Conclusion and Future Work

Further Improvements

- ➢ Extend predictions to sites without automated sampling using interpolation techniques
- ➢ Develop model based only on shore data:
  - Rainfall (runoff data)
  - Heidelberg Tributary Loading Inlet Data

Automated sampling station at Coldwater Creek at Coldwater, Ohio

https://ncwqr.org/monitoring/

Conclusion and Future Work

Climate Change:

- ➢ Model shows influence of temperature
- ➢ HAB danger will only increase
- ➢ 68% of USians rely on surface drinking water sources - vulnerable to changing conditions / HAB invasion

https://www.ewg.org/interactive-maps/2019_microcystin/