

Rhys Leahy

Prof. Amir Jafari

Intro to Data Mining

06 December 2021

Introduction:

Our group undertook a project to explore and predict the relationships between physical properties, surface temperature, and ice concentration across the Great Lakes. First, we queried, cleaned, and assembled multiple datasets from NOAA's CoastWatch program and The Great Lakes Environmental Research Laboratory (GLERL). Then we conducted some exploratory data analysis and built three different models: a KNN model to classify properties by lake, a linear regression model to quantify the relationship between surface temperature and ice concentration, and a logistic regression model to classify whether ice concentration would exceed a particular threshold based on surface temperature. Within our group, I led our efforts to query, clean, join, and warehouse multiple datasets from NOAA and GLERL so we could pursue a project that went beyond a prepackaged dataset from Kaggle or similar sites. Instead, we were able to develop a project based on a real-world, messy dataset with implications for climate change.

In addition to the technical process that I describe below, I also managed and maintained our Git repo. Carter, Nongjie and I met one to two times a week every week starting in early November to discuss project planning, data, and modeling approaches. This open and consistent communication enabled us all to learn from each other and be involved across all phases of the project.

Description:

I first identified these datasets through NOAA's public websites, and then Carter, Nongjie and I collaboratively developed SMART questions to drive our exploratory data analysis and modeling approaches. Ultimately, we agreed on three key questions: Based on surface temperature and physical properties, can you predict the percentage ice concentration? Based on surface temperature and physical properties, can you classify whether the percentage ice concentration will exceed a given threshold across the Great Lakes? Lastly, can you predict which lake a set of characteristics (surface temperature, ice concentration, and physical properties) most likely belongs to?

Great Lakes Average Ice Concentration								

Ice Concentration (%)								

Year	Day	Sup.	Mich.	Huron	Erie	Ont.	St.Clr	GL Total

2008	344	2.10	2.12	5.58	0.42	0.24	34.56	2.76
2008	346	2.08	2.29	6.24	0.63	0.27	15.33	2.90
2008	350	3.65	4.24	8.64	7.76	1.05	24.88	5.25
2008	353	4.94	7.66	10.39	6.93	1.40	53.04	6.97
2008	357	5.34	15.50	18.13	13.43	3.06	92.41	11.61
2008	360	7.51	18.18	16.52	16.57	2.63	91.41	12.88
2008	364	4.26	9.29	11.44	9.68	1.65	60.36	7.64
2009	001	6.13	13.52	15.76	9.57	2.85	60.41	10.38

Example .dat file containing ice concentration data from 2008. Data comes from NOAA/GLERL: https://coastwatch.glerl.noaa.gov/statistic/ice/dat/g2008_2009_ice.dat

Since our key questions and modeling efforts hinged on surface temperature and ice concentration, I focused on obtaining and wrangling these datasets which were stored in 26 different .dat files hosted at different URLs. To integrate the ice and surface temperature data, I

leveraged the Python Requests library to query 13 web hosted .dat tables containing daily observations for ice concentration (pictured above) and the 13 web hosted .dat tables containing daily observations for surface temperature (pictured below) going back to 2008 (Chandra). Since each of the .dat files contained messy headers, white spaces, and extra characters, I wrote a function (lines 23-34 of gather_data.py) to query the ice .dat file, strip the headers and white spaces, and return a clean dataframe of observed ice concentration by day and lake. Lines 70-72 of gather_data.py show how we used list comprehension and the Pandas library to query and concatenate each of these data frames into a master ice dataframe (McKinney). Then, I used Pandas' "melt" method to reshape our data from wide to tidy and long format so each row contained "Lake" as part of the relation. Lastly, I added a common id composed of year, day, and lake for each observation to help merge the ice and surface temperature datasets.

Daily Lake Average Surface Water Temperature From Great Lakes Surface Environmental Analysis maps							

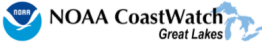
Surf. Water Temp. (degrees C)							
Year	Day	Sup.	Mich.	Huron	Erie	Ont.	St.Clr

2008	001	2.75	3.47	2.44	1.99	3.25	0.20
2008	002	2.74	3.41	2.42	1.92	3.20	0.20
2008	003	2.58	3.26	1.97	1.64	3.20	0.20
2008	004	2.57	3.25	1.96	1.65	3.21	0.20
2008	005	2.57	3.24	1.93	1.68	3.24	0.20
2008	006	2.57	3.19	1.93	1.69	3.24	0.20
2008	007	2.57	3.15	1.92	1.74	3.27	0.20
2008	008	2.56	3.11	1.92	1.78	3.31	0.20
2008	009	2.57	3.09	1.92	1.81	3.35	0.20
2008	010	2.58	3.10	1.91	1.78	3.33	0.20
2008	011	2.59	3.33	2.13	2.26	3.35	0.20
2008	012	2.58	3.33	2.11	2.22	3.30	0.20
2008	013	2.58	3.32	2.09	2.18	3.28	0.20

*Example .dat file containing surface temperature data in Celcius from 2008. Data comes from NOAA/GLERL:
https://coastwatch.glerl.noaa.gov/ftp/glsea/avgtemps/2008/glsea-temps2008_1024.dat*

Then, I developed a similar function (lines 36-51 of `gather_data.py`) to query and clean the 13 .dat files containing observed surface temperature. However, since these files are transferred through https and also through ftp in each request, this function was especially prone to slow run times and connection errors, even after implementing a try and except block and increasing the timeout between calls to three minutes. In PyCharm's interactive console, I was eventually able to call the function on each of the URLs point to a .dat file and concatenate the returned objects together through repeated requests, but to make the data more accessible for future analyses I also hosted a copy of the full dataset on Google Cloud storage. Once I queried and cleaned the full surface temperature dataset, I used Pandas to perform an outer join on our ice dataframe and temperature DataFrame. Using Pandas again, I dropped the duplicated columns after the merge and stored a copy of the clean, fully merged DataFrame with 31,126 rows and six features for initial testing.

To account for the potential impact of a lake's physical characteristics on ice formation in our models (e.g. how depth vs. surface area or wind speeds might affect freezing), I also created a Pandas DataFrame (lines 131- 141) drawing on NOAA statistics about each lake's properties. I then performed an outer join between our lake properties DataFrame and our DataFrame containing surface temperature, ice concentration, year, day, and lake in long and tidy format. The resulting dataset contained a total of 31,126 rows with 16 variables. However, in further preprocessing we filtered observations where ice concentration was NA, resulting in 8,855 rows for training, testing, and model development. This follows the logic that ice measurements are only available for about a quarter of the year during the winter months, whereas surface temperature measurements are collected throughout the year.


[Home](#) · [GLSEA](#) · [MODIS](#) · [Ocean Color](#) · [In-Situ](#) · [Statistics](#)

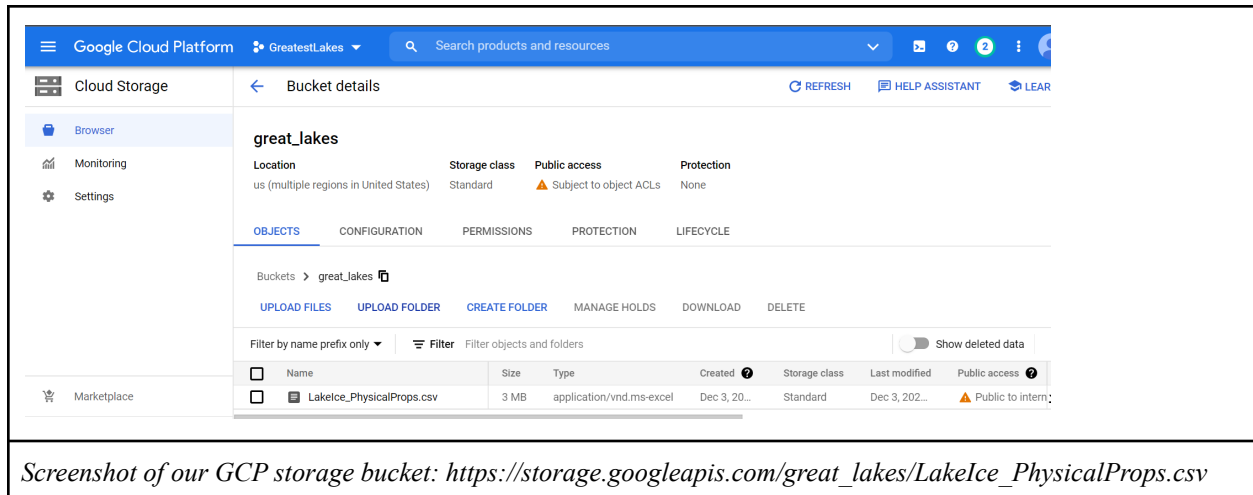
Physical Characteristics of the Great Lakes

		Superior	Michigan	Huron	Erie	Ontario	Totals
Elevation ^a	(feet)	600	577	577	569	243	
	(metres)	183	176	176	173	74	
Length	(miles)	350	307	206	241	193	
	(kilometres)	563	494	332	388	311	
Breadth	(miles)	160	118	183	57	53	
	(kilometres)	257	190	245	92	85	
Average Depth ^a	(feet)	483	279	195	62	283	
	(metres)	147	85	59	19	86	
Maximum Depth ^a	(feet)	1,332	925	750	210	802	
	(metres)	406	282	229	64	244	
Volume ^a	(cu. miles)	2,900	1,180	850	116	393	5,439
	(km ³)	12,100	4,920	3,540	484	1,640	22,684
Water Area	(sq. mi.)	31,700	22,300	23,000	9,910	7,340	94,250
	(km ²)	82,100	57,800	59,600	25,700	18,960	244,160
Land Drainage Area ^b	(sq. mi.)	49,300	45,600	51,700	30,140	24,720	201,460
	(km ²)	127,700	118,000	134,100	78,000	64,030	521,830
Total Area	(sq. mi.)	81,000	67,900	74,700	40,050	32,060	295,710
	(km ²)	209,800	175,800	193,700	103,700	82,990	765,990
Shoreline Length ^c	(miles)	2,726	1,638	3,827	871	712	10,210 ^d
	(kilometres)	4,385	2,633	6,157	1,402	1,146	17,017 ^d
Retention Time	(years)	191	99	22	2.6	6	
Outlet	St. Marys River Straits of Mackinac St. Clair River Niagara River/ Welland Canal St. Lawrence River						

Statistical table from NOAA/GLERL: <https://coastwatch.glerl.noaa.gov/statistic/physical.html>

Statistical table from NOAA/GLERL: <https://coastwatch.glerl.noaa.gov/statistic/physical.html>

Finally, to ensure that the dataset was fully accessible to Carter and Nongjie for modeling, and to ensure that our code could be replicated by anyone, I ultimately hosted a copy of our final dataset on Google Cloud Storage. Since our gather_data.py file was prone to connection and timeout errors (even after extensive debugging sessions and error handling revisions), keeping a copy of the data on Google Cloud allowed us to have a reliable and replicable final dataset.



Results:

As a result of my data wrangling, warehousing, and preprocessing efforts, our group was able to develop not one, but three models and GUIs exploring the relationship between ice concentration, surface temperature, and physical properties of the Great Lakes. Carter's KNN model significantly outperformed the naive baseline. Likewise, Nongjie's logistic regression model accurately predicted whether ice coverage would exceed the long term average based on surface temperature.

Our final results of Carter's KNN model show that the model performed best with a k-value of 23. While the model only classified observations at about 40%, this is significantly better than the naive baseline of 16.7%. Since Carter's model performed best on lake's with the most distinct physical properties (Lake St. Clair) and worst on the lakes with the most similar physical properties (it performed poorly distinguishing between Lake Superior vs. Lake Ontario), we can conclude that these physical properties are significant features.

Our final results of Nongjie's Logistic Regression model show that it was very accurate at

predicting whether ice coverage will exceed the long term average ice coverage in the Great lakes. The test set accuracy was approximately 93%, which was almost identical to the training set score, so we can reasonably conclude that there is no over-fitting. While this model was most prone to False Negatives, it still maintained a precision rate of 99%.

Ultimately, our modeling efforts revealed that you can predict ice concentration based on surface temperature and physical properties. Furthermore, Nongjie's logistic regression model showed that you can classify whether the percentage ice concentration will exceed a given threshold set by long term averages across the Great Lakes. Lastly, Carter's KNN model demonstrated that you can predict which lake a set of characteristics (surface temperature, ice concentration, and physical properties) most likely belongs to.

Summary and Conclusions:

In this project I learned how to query files and webpages using Python's Requests library. I also applied our lessons on preprocessing, string operations, and Pandas to clean and structure messy text data into a useful table format. Furthermore, I also learned how to transform and reshape datasets to different formats, and about the different types of joining and merging operations that Pandas offers that closely resemble some SQL operations. Most importantly, I learned how to effectively perform preprocessing and data wrangling to facilitate streamlined model development. By aggregating, exploring, and preparing the data,

In a future project, with fewer time and resource constraints, I would dedicate more effort to improve error handling and the connection timeout errors in `gather_data.py`. After testing and debugging this script across multiple days, I often found that it would run best on the evenings and weekends, when NOAA and GLERL servers are likely under a lighter load. Since these

factors were beyond our control, this proved a useful opportunity to learn a little bit about cloud computing and how GCP's storage solutions can integrate into data mining and modeling projects.

Individual Code Contribution:

I wrote 153 lines of code in total. Approximately 25 of these lines were based on package documentation. For example, I drew on Pandas documentation to “melt” the different datasets into long and tidy form. I also drew on Pandas documentation to perform different types of joins and merging operations. Since our aggregating and preprocessing routines were highly customized to this project, I always modified any example code in the documentation to fit our use case. I modified every line of code, but approximately 10 lines closely resemble snippets I found in package documents. $(25-15)/(25 + 128) * 100 = 5\%$.

Works Cited

Chandra, R. V., & Varanasi, B. S. Python requests essentials. Packt Publishing Ltd, 2015.

Great Lakes CoastWatch Program Overview.

<https://coastwatch.glerl.noaa.gov/overview/cw-overview.html>. Accessed 5 Dec. 2021.

Great Lakes Physical Characteristics. <https://coastwatch.glerl.noaa.gov/statistic/physical.html>.

Accessed 5 Dec. 2021.

Great Lakes Statistics. <https://coastwatch.glerl.noaa.gov/statistic/statistic.html>. Accessed 5 Dec.

2021.

McKinney, W., et al. Data structures for statistical computing in python. In Proceedings of the 9th Python in Science Conference (Vol. 445, pp. 51–56), 2010.