Abstract

Sea ice concentration in the Arctic Circle has been trending downwards over the past century. Data regarding sea ice data is useful in a variety of ways: the concentration of sea ice informs potential sea routes for seafaring vessels; it influences the class of vessels which may traverse a region; and it provides information about the impacts of climate change. This project uses the Random Forests regression model in the skforecast Python library to develop a predictive model of sea ice concentration in the Arctic Circle. Data from the National Snow and Ice Data Center (NSIDC) is used for analysis. The results indicate that sea ice concentration will continue to trend downwards.

1. Introduction

As sea ice in Arctic Regions continues to decline over the years, it is important to understand the rate of decline so as to quantify the possible effects of this change. By utilizing data available from the Geoscience Data Exchange (GDEX) this project aims to predict future sea ice concentrations and to identify trends in sea ice concentration decline.

The data obtained from the GDEX online database, discussed in subsequent sections, will be analyzed to extract trends, and regression techniques will be applied to predict future sea ice concentrations in given regions. Following this, geospatial analysis will be used to produce informative maps to summarize the findings as well as graphs depicting sea ice concentration trends.

2. Methodology

2.1. Libraries

The following Python libraries are used in this project, along with basic libraries such as os:

1. Pandas: A standard Python library for working with dataframes.
2. Xarray: A library for working with multi-dimensional arrays.
3. Matplotlib: A library for visualizing data and creating plots and figures.
4. Numpy: A popular library for mathematical and data functions.
5. Cartopy: A library for plotting and visualizing geospatial data.
6. Scikit-learn: Provides tools for complex and predictive data analysis.
7. Skforecast: Adapts Scikit-learn regressors for time-series forecasting.
8. Scipy: Algorithms for optimization, interpolation, and other tools.

2.2. Data

Data is obtained from the National Snow and Ice Data Center (NSIDC) which hosts data about sea ice concentration, extent, and other related datasets. The data is downloaded as a Python xarray and converted to a Pandas dataframe. The data contains information regarding time and date of measurement, the measurement of concentration in percentage, and the location of the measurement in latitude and longitude coordinates. Calling .info() on the original dataset yields the following:

```
xarray.Dataset {
dimensions:
        time = 1827 ;
        lat = 180 ;
        lon = 360 ;

variables:
        float32 SEAICE(time, lat, lon) ;
                SEAICE:info = sst-ice consistency enforced ;
                SEAICE:units = % ;
                SEAICE:long_name = Sea Ice Concentration ;
        int32 date(time) ;
                date:units = yyyymmdd ;
        float64 date_frac(time) ;
                date_frac:units = yyyymmdd.fraction_of_day ;
        int32 datesec(time) ;
                datesec:units = current seconds of current date ;
        float32 lat(lat) ;
                lat:units = degrees_north ;
                lat:long_name = latitude ;
        float32 lon(lon) ;
                lon:units = degrees_east ;
                lon:long_name = longitude ;
        datetime64[ns] time(time) ;
                time:information = middle of month ;
```

The data is preprocessed using the processData class to remove NaN/np.nan values and outliers are detected and removed using statistical methods (z-scores). Once cleaned, the time column is converted into three separate datetime columns using Pandas to_datetime() function: date, month, and year. All data years before 1950 are dropped due to poor quality data before this threshold year.

2.3. Analysis of Data

The select_year function is called on the instantiated object, and the desired year and month of analysis are passed as integers. The data is searched for data entries matching the desired year and month. This new dataframe is returned. A instance of the visualizeConcentration class is instantiated and the draw_fig function is called.

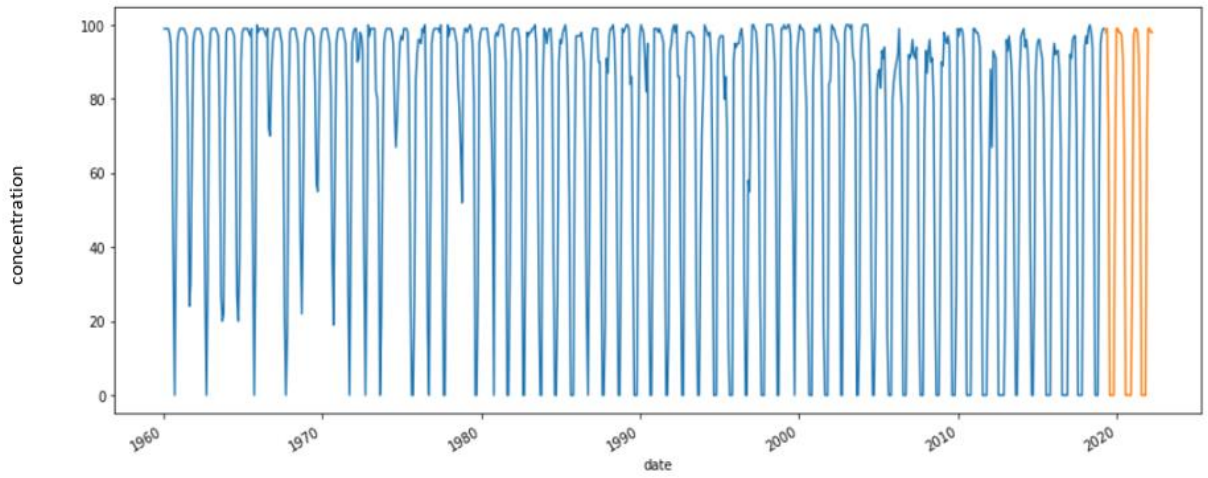
Within this function, the Cartopy library is used to set the map projection to one of the Arctic Circle, called ccrs.NorthPolarStereo(). Various methods are called to zoom the map to the correct region and to visualize land and coastline shapes. Numpy and Scipy are then used to create the data for a heatmap. With a spacing of 50[1], numpy.linspace() generates an array of numbers between the minimum and maximum latitude, and another array between the minimum and maximum longitude. Numpy.meshgrid is used to create a coordinate matrix from these arrays. Scipy.griddata() is next applied to the data. Griddata works by passing the data point coordinates (the latitude and longitude), the data values (concentration values), and points at which to interpolate the data (the generated meshgrid values). The function interpolates between these points, allowing for a heatmap of values.

Matplotlib's pcolormesh is passed the interpolated values along with the meshgrid values. A heatmap of the region is returned. Note that the transform ccrs.PlateCarree() is called to match the coordinate system of the data to that of the map.

This process is repeated with a second year, and the two maps are saved to be displayed in the generated report for visual comparison.

[1] 500 is a preferable spacing for a more detailed visualization; however, due to runtime constraints, 50 was adopted for the purposes of demonstration.

The next step is to conduct the predictive analysis of the data in order to estimate Arctic ice concentration at a given time in the future. The predictIce class is instantiated. The data is split into training and testing sets, with the last 12 months used for testing. The split is not randomized as a randomized training set will no longer contain the temporal aspect of the data. Matplotlib is used to generate plots of the training and testing set for visual inspection.



The forecaster function is called on the instantiated object to begin the forecasting using skforecast. The selected regressor is the Random Forests regressor available from the sklearn library. This was selected based on splitting the data into training and testing sets and then running various algorithms: Linear regression, neural networks, k-nearest neighbors, random forests, and support vector regression. Using sklearn's cross_val_score function, the algorithms were compared, yielding the following:

Box Plot comparing Potential algorithms

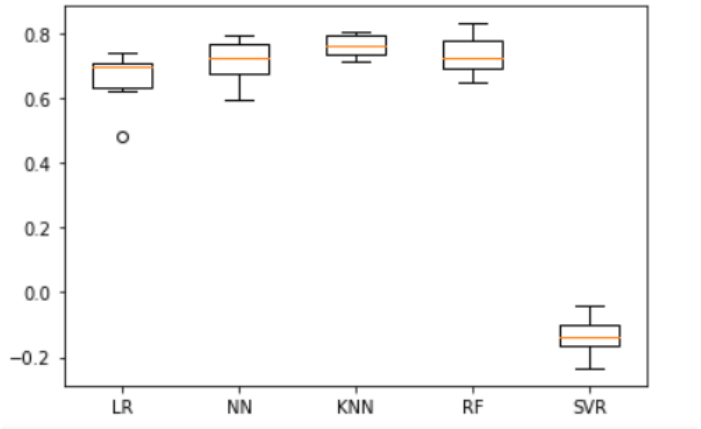


Figure: Potential algorithms compared.

Thus, due to the high score of the model as well as familiarity with the model, random forests was selected as the algorithm for this problem.

The training data is fit to the model and a dataframe is created from the predictions. In order to update the index of the dataframe to have a datetime format, the Pandas DateOffset function is called to add dates

at the given interval from the end of the data to the predicted date. The created dates and the predictions are saved as .csv and .hdf5 files for each coordinate set in the region.

Using the generated prediction data, another visualizeConcentration object is instantiated and the process of producing a heatmap is repeated. This figure is also saved for insertion into the html report. The final step is instantiating an htmlReportGenerator object with all the generated data and images, thus producing the html report.

2.4. Validation and Optimization

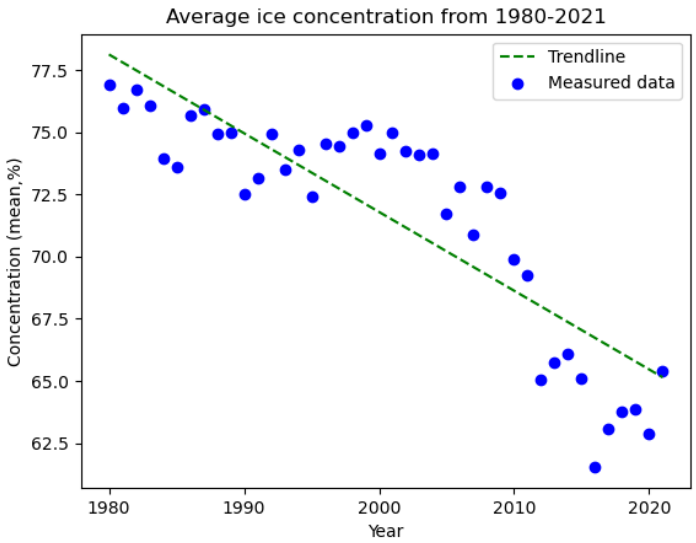
There are several methods for validating and optimizing models.

To optimize the model, the skforecast grid_search_forecaster function is applied to the fitted model. By varying the number of lags[2] and estimators, the optimal model is returned by reducing the mean squared error and/or mean absolute error. Thus, the model with the best configuration has n_estimators equal to 500 and 12 lags.

To validate the model, the actual concentrations and the predicted concentrations for the test set are compared based on mean absolute error, mean squared error, and R^2 score. The scikit-learn metrics class contains functions to determine these values out of the box. For mean absolute error and mean squared error, a lower value indicates a better fit, with a value of 0 indicating a "perfect" (and usually overfit) model. The closer an R^2 score is to 1, the better the fit.

3. Results

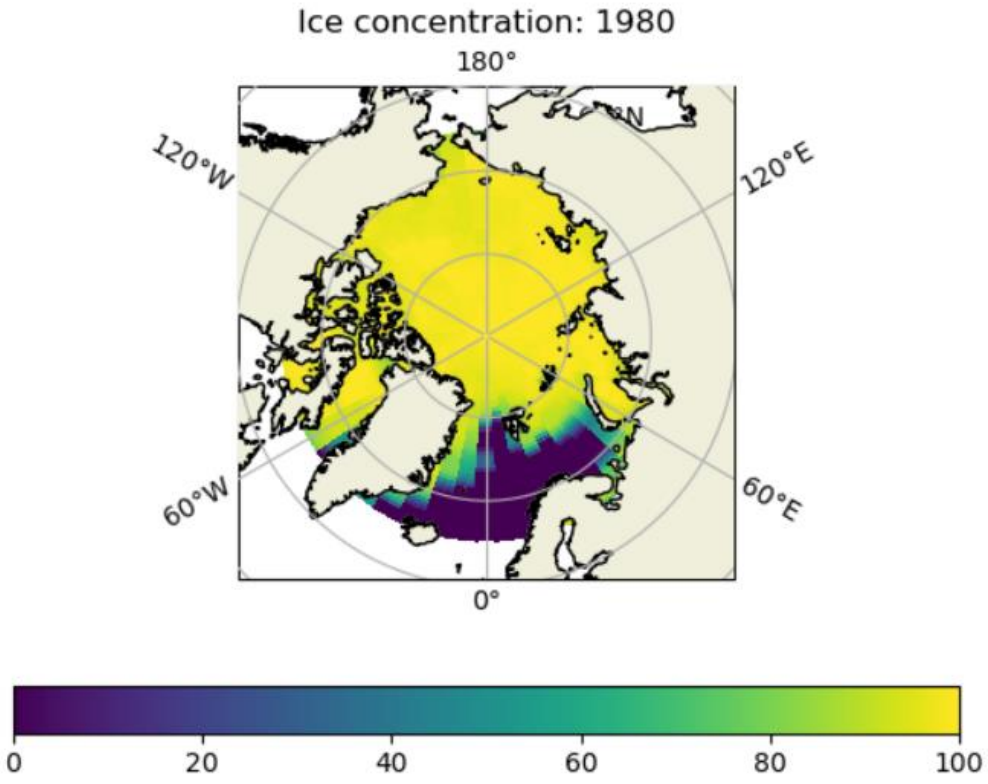
The generated data and figures are presented below.



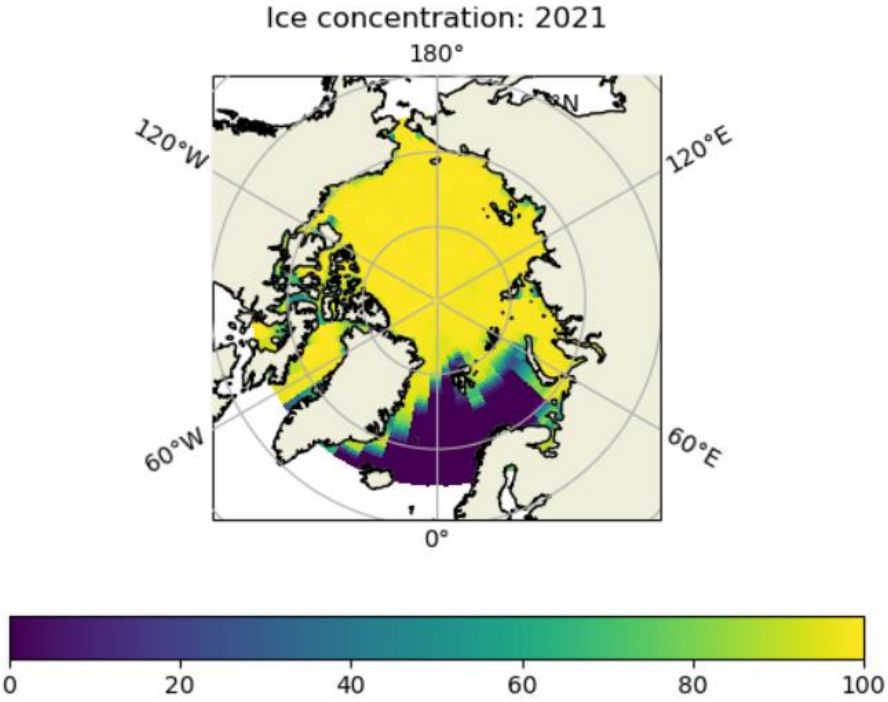
[2] A lag

First, average ice concentration for each year is plotted to explore the data and understand trends in the data. Evidently, there is a downward trend in average yearly sea ice concentration. Thus, we proceed with making predictions.

To visualize this trend, ice concentration for the entire Arctic in 1980 is plotted onto a map. This geospatial analysis is given below and shows, using a heatmap, the concentration of sea ice at each coordinate pair:

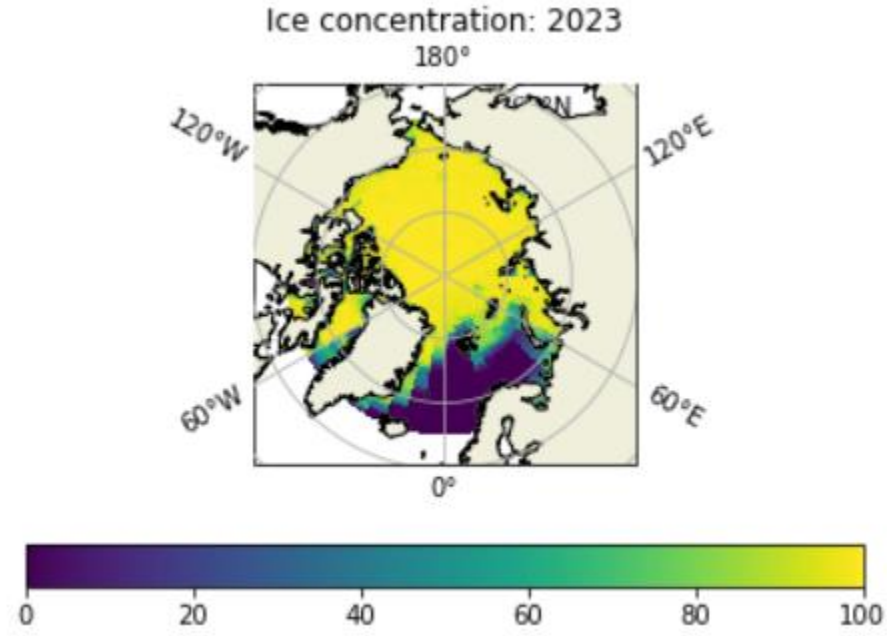


This was repeated for the year 2021. Note that the month of March was used for both maps for illustration purposes only.



A quick visual inspection shows that the ice concentration, particularly surrounding coastal areas, has receded in the decades between 1980 and 2021.

Continuing with our analysis, we obtain the following heatmap from predicted values for March in the year 2023:



Again, evidence of sea ice concentration reduction is visible from the above plot. Further analysis can be done with the .csv and .hdf5 data files generated in the code.

4. Discussion

As expected, ice concentration continues to trend downwards. Ice concentration for the following year is expected to reduce in areas surrounding land masses, thus seeing ice extent in coastal areas recede. If this trend continues, there will be significant reductions in ice concentration in the Arctic Region.

Several improvements could be made to this model, as there were some limitations. The main limitation was the computing power available, thus requiring reducing the size of the dataset used in order to run the algorithms required for analysis. As well, hyperparameters could be further tuned in order to identify optimal model conditions; for example, capturing temporality requires choosing the optimal train/test split.

5. Conclusion

The results of this project indicate that, as expected, sea ice concentration is reducing and sea ice is therefore thinning. This model accurately tracks the downwards trend of sea ice concentration and fits a model which predicts the continuing decline of sea ice.

Appendix A: Required Project Tasks Checklist

- ✓ Read data from Pandas
- ✓ Data computations using numpy, Scipy
- ✓ Data computations using additional package(s)
    - Scikit-learn, Skforecast
- ✓ Processed data written as .csv/HDF5 files.
- ✓ Visualize data using Matplotlib
- ✓ Visualize data using another tool
    - Cartopy
- ✓ Figures need titles, labels, annotations
- ✓ Four+ classes:
    - processData, predictIce, visualizeConcentration, htmlReportGenerator, modelOptimization
- ✓ Import classes into main code.
- ✓ Create github repository.