ID2221 – Data Intensive Computing

My Contribution to Arctic Meltdown

By:

Nagasudeep Vemula<vemula@kth.se>

Akhil Yerrapragada<akhily@kth.se>

Problem description

To analyse the factors causing the Arctic Ice to melt down and predict the upcoming emission rates using existing data. Here, we are intended to provide quantitative analysis on Methane, Co2 and other harmful gasses responsible for the cause and calculate how much an individual is responsible for these emissions per year. Also, the calculated data will be given as an input to predictive analysis to forecast upcoming emission rates.

Data

The data used for this project contains levels of Carbon Dioxide [2], Methane, Nitrous Oxide emitted, temperature raise, and the quantity of ice melted in arctic over the years. This data is unbounded, structured, available in JSON format and can be accessed by using Global Warming API [1].

Tools

1) Scala [3] – Programming Language
2) Play Framework [4] – Web Framework to build MVC applications
3) Cassandra [5] – NoSQL database to store data.
4) Apache Spark [6] – For batch data processing.
5) ARIMA [8] – Time series forecasting
6) MLlib [7] – Library with ML Algorithms
7) vegas-viz [9] – Matplot Library to convert dataframe into visualizable Json
8) vega-lite [10] – For visualizing the charts

Process

Application Architecture:

The application (MVC) contains a build file(build.sbt), routes file, controller (HomeController.scala), service(DataProcessing.scala) and 2 views(calculations.scala.html and home.scala.html). Let us look into the calculations part in detail:

1) HomeController.scala – This file is responsible to hold all the GET end points required to trigger required calculation. It contains end points for CO2Emissions, Methane, NOEmissions, Polar Ice and Temperature calculations.

2) DataProcessing.scala – This file is responsible to hold all the calculations required for the above mentioned. Two types of calculations are implemented:

   2.1) Individual contribution to emission: The amount of gas emitted per year is received in units, ppm (parts per million). Firstly, the received ppm value is converted to kilograms per acre-foot. Here, 1ppm = 1.233 kilograms per acre-foot [11]. Secondly, Kg per Acre is converted to Kg per Sqkm. Third, Kg per Sqkm is converted to Tons per Sqkm. Finally, Tons per Sqkm is divided by population per Sqkm.

   2.2) Time series forecasting: Time series forecasting is implemented using ARIMA (Auto Regressive Integrated Moving Average). The received calculated output is auto fitted into ARIMA and is forecasted for next 20 observations using **.forecast** API.

Data Processing Implementation:

Step 1-:

The required maven dependencies are added to build the .sbt file.
Defined a spark session and established connection to Cassandra.

Step 2-:

The Json data is batch processed using Spark SQL and is received in the form of a dataframe. Following this, the dataframe is then exploded into to get the required hierarchy from root.

Step 3-:

The required values in exploded data frame are converted to RDD.

Step 4-:

The RDD contains 12-month data per year. Therefore, **reduceByKey** is used and the overall value obtained per key is then divided by number of months to obtain average. The overall result is then stored in the RDD.

Step 5-:

All calculations are performed and are received in an RDD. The Result is then pushed to Cassandra using **saveToCassandra.** Below image shows the calculations performed on RDD's

```
// Ppm to Kg per acre foot
var ppmTokgRDD = averagedRDD.map(x =>( x._1, x._2 * 1.233))

// Kg per acre to Kg per sqkm conversion
var acreTosqkmRDD = ppmTokgRDD.map(x =>( x._1, (x._2 * 1000000)/4047 ))

// Kg per sqkm to tons per sqkm
var kgToTonsRDD = acreTosqkmRDD.map(x =>( x._1, x._2 / 1000 ))

//Tons per sqkm / population per sqkm
var finalRDD = kgToTonsRDD.map(x =>( x._1, x._2 / 27.2 )).sortBy(_._2)

finalRDD.saveToCassandra("environmental_calculations", "co2", SomeColumns("year", "tonnes"))
```

Step 6-:

To perform predictions, we give the calculated data as an input to ARIMA. ARIMA uses **.autofit** API to define a model from input. The model is later fit into the **.forecast API** which generates the prediction results.

Step 7-:

The results are then zipped to years RDD and given to vegas-viz plotter to generate the json.

Running the code:

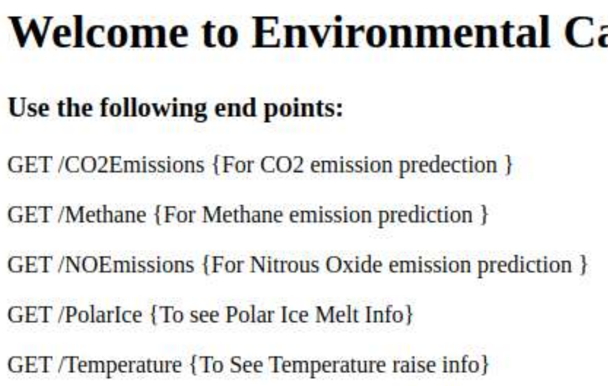
1) Use the command **sbt run** in root of the project initiate AkkaHTTPServer. Info appears as mentioned in the below image:

```
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/nanda/.ivy2/cache/ch.qos.logback/logback-classic/jars/logback-classic-1.2.3.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/nanda/.ivy2/cache/org.slf4j/slf4j-log4j12/jars/slf4j-log4j12-1.7.16.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [ch.qos.logback.classic.util.ContextSelectorStaticBinder]
--- (Running the application, auto-reloading is enabled) ---

[info] p.c.s.AkkaHttpServer - Listening for HTTP on /0:0:0:0:0:0:0:0:9000

(Server started, use Enter to stop and go back to the console...)
```

2) Use **localhost:9000** url to view the index page. The index page appears as below:



Welcome to Environmental Calculations Pannel

Use the following end points:

GET /CO2Emissions {For CO2 emission predection }

GET /Methane {For Methane emission prediction }

GET /NOEmissions {For Nitrous Oxide emission prediction }

GET /PolarIce {To see Polar Ice Melt Info}

GET /Temperature {To See Temperature raise info}

3) Navigate to suggested routes to initiate calculations as per required.

4) The obtained output can be viewed in 2 ways:

4.1) Cassandra:

Use command **cqlsh** in terminal to open Cassandra command palette.

- All the tables are created in **environmental_calculations** keyspace. It can be accessed using command **use environmental_calculations;**
- Co2 Emissions calculations are available in **co2** table**.** Accessible by using the command **SELECT * from co2;**
- Methane Emissions calculations are available in **methane** table**.** Accessible by using the command **SELECT * from methane;**
- Nitrous Oxide Emissions calculations are available in **noemissions** table**.** Accessible by using the command **SELECT * from noemissions;**
- Polar Ice Area remaining calculations are available in **polaricevalues** table**.** Accessible by using the command **SELECT * from polaricevalues;**
- Temperature changes calculations are available in **temperaturechanges** table**.** Accessible by using the command **SELECT * from temperaturechanges;**
- Prediction calculations are available in **predictioncalculations** table**.** Accessible by using the command **SELECT * from predictioncalculations;**

4.2) Vega-lite:

The generated Json on the screen after performing the calculations can be visualized using vega-lite.

Results

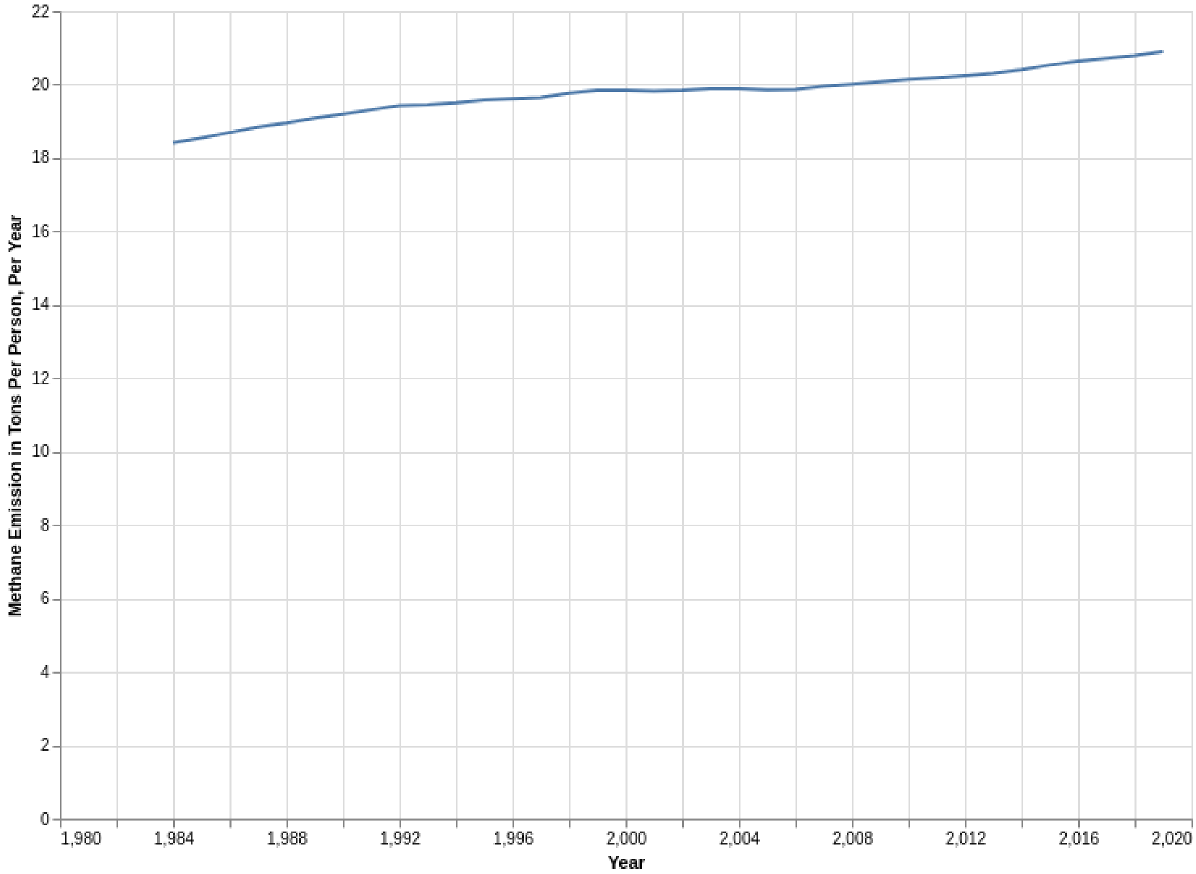
1 Actuals:

1.1 Methane Emissions:

Calculated results in Cassandra:

year	tonnes
1992	19.42272
1999	19.84836
2019	20.90127
2014	20.40842
1983	9.16251
2003	19.89317
1988	18.95227
2009	20.07238
1995	19.57954
2010	20.13959
2005	19.85956
2006	19.87077
2012	20.2404
2017	20.71085
2007	19.94917
1998	19.76995
2015	20.53163
2001	19.82596
2002	19.84836
1990	19.1987
2000	19.84836
2020	10.48424
1984	18.41462
1985	18.54903
1989	19.08669
2011	20.1844
1991	19.31071
1993	19.44512
1996	19.61314
1994	19.50113
2004	19.89317
2018	20.78926
1987	18.84026
1986	18.69465
2016	20.63244
1997	19.64674
2013	20.30761
2008	20.00518

Results in Vegas-Lite:

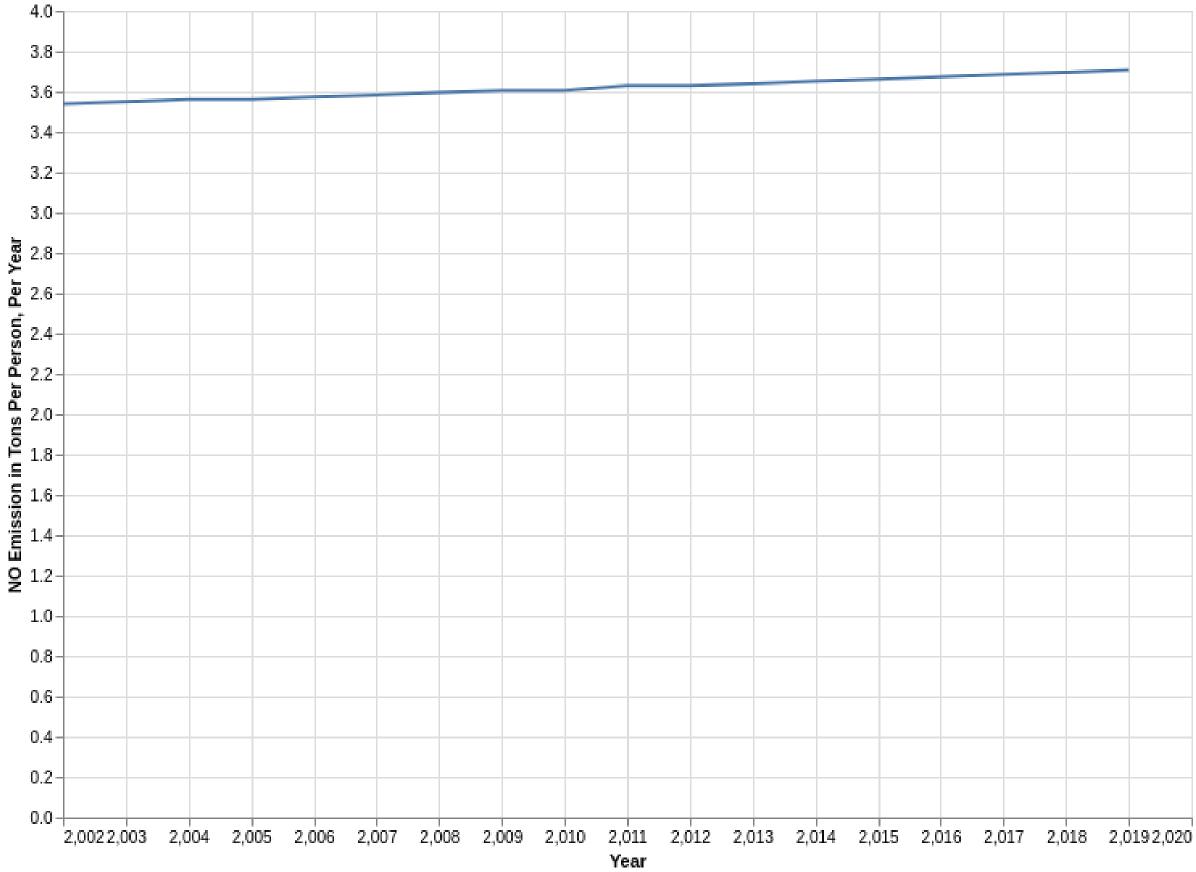


1.2 Nitrous Oxide Emissions:

Calculated results in Cassandra:

year	tonnes
2019	3.70757
2014	3.65156
2003	3.55075
2009	3.60676
2010	3.60676
2005	3.56195
2006	3.57315
2012	3.62916
2017	3.68516
2007	3.58435
2015	3.66276
2001	2.94589
2002	3.53955
2020	1.85938
2011	3.62916
2004	3.56195
2018	3.69637
2016	3.67396
2013	3.64036
2008	3.59556

Results in Vegas-Lite:

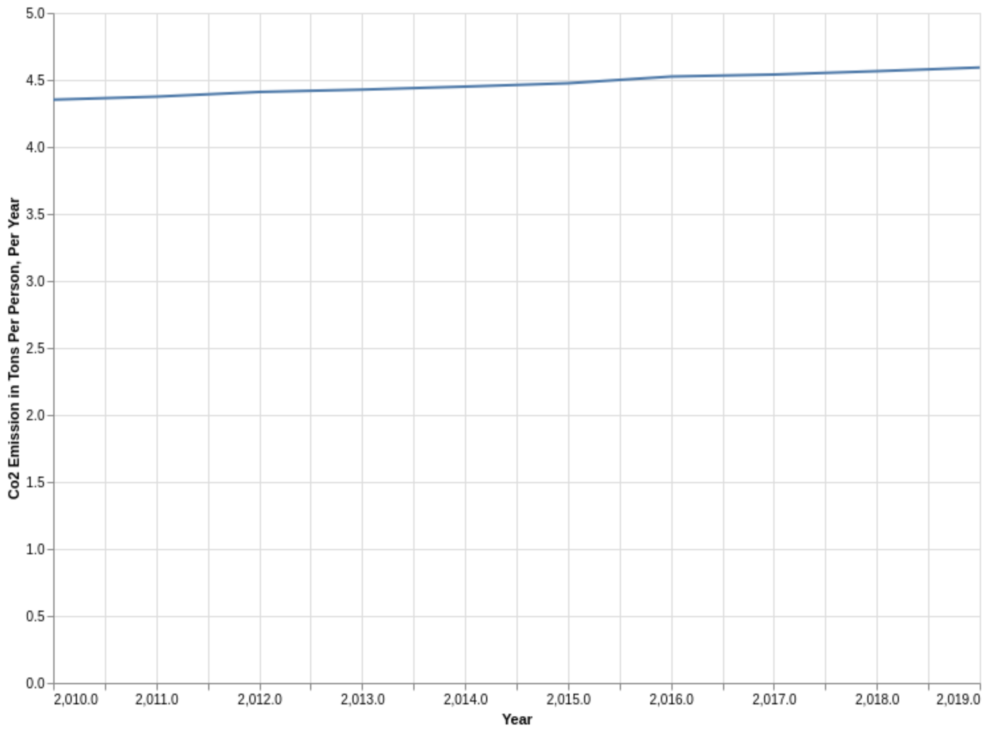


1.3 Co2 Emissions:

Calculated results in Cassandra:

year	tonnes
2019	4.5932
2014	4.451
2010	4.35247
2012	4.40914
2017	4.5397
2015	4.47577
2020	3.70826
2011	4.37509
2018	4.56465
2016	4.52439
2013	4.42853

Results in Vegas-Lite:

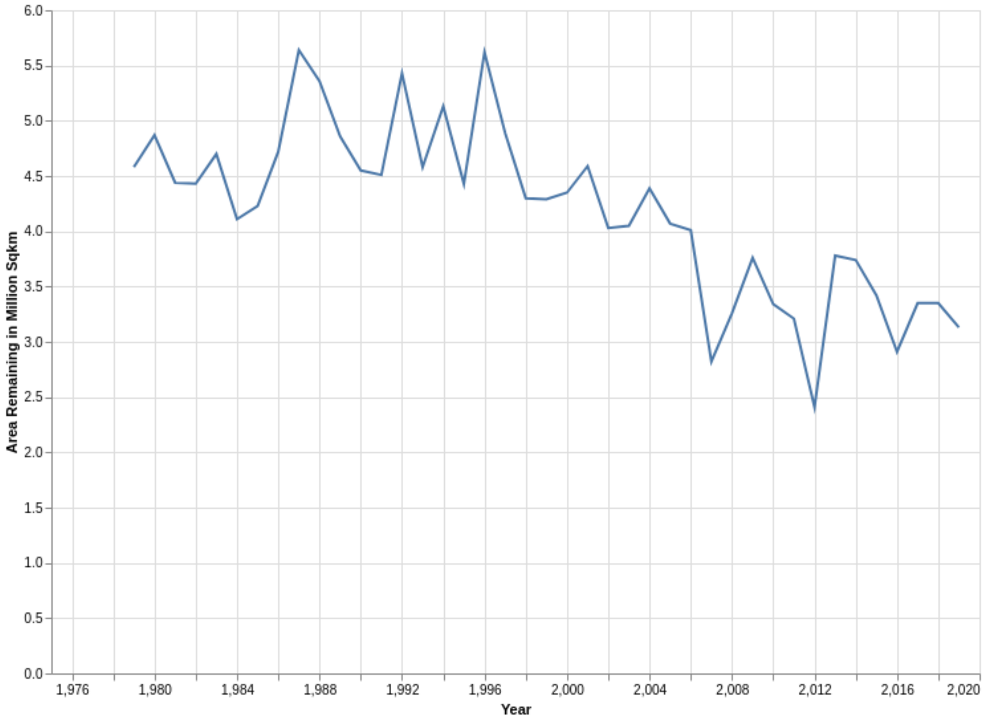


1.4 Arctic Area Remining:

Calculated results in Cassandra:

year	area	extent
1992	5.43	7.47
1999	4.29	6.12
2019	3.13	4.32
1982	4.43	7.3
1981	4.44	7.14
2014	3.74	5.22
1983	4.7	7.39
2003	4.05	6.12
1988	5.36	7.37
2009	3.76	5.26
1995	4.43	6.08
2010	3.34	4.87
2005	4.07	5.5
2006	4.01	5.86
2012	2.41	3.57
1979	4.58	7.05
2017	3.35	4.82
2007	2.82	4.27
1998	4.3	6.54
2015	3.42	4.62
2001	4.59	6.73
2002	4.03	5.83
1990	4.55	6.14
2000	4.35	6.25
1984	4.11	6.81
1985	4.23	6.7
1989	4.86	7.01
2011	3.21	4.56
1991	4.51	6.47
1993	4.58	6.4
1996	5.62	7.58
1994	5.13	7.14
2004	4.39	5.98
2018	3.35	4.79
1987	5.64	7.28
1980	4.87	7.67
1986	4.72	7.41
2016	2.91	4.53
1997	4.89	6.69
2013	3.78	5.21
2008	3.26	4.69

Results in Vegas-Lite:

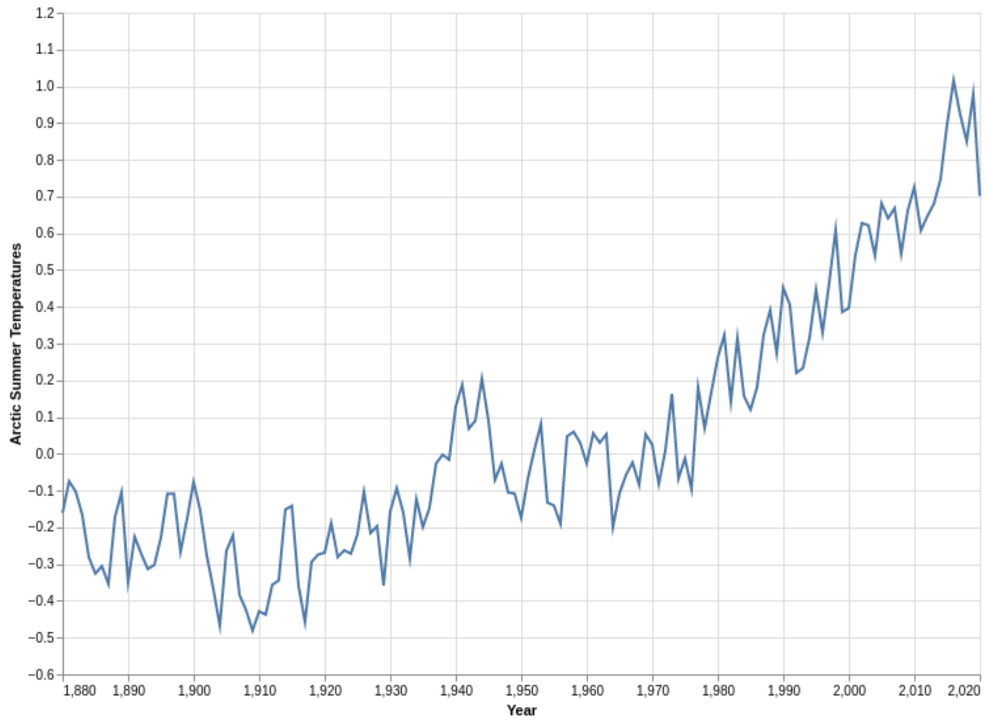


1.5 Arctic Summer Temperatures:

Calculated results in Cassandra:

year	temperature
1940	0.128333
1902	-0.275833
1898	-0.266667
1909	-0.480833
1937	-0.026667
1897	-0.108333
1942	0.068333
1944	0.204167
1899	-0.1775
1992	0.22
1889	-0.105
1933	-0.284167
1999	0.385833
1971	-0.0825
1928	-0.196667
1886	-0.305833
1953	0.080833
1976	-0.095833
2019	0.978333
1982	0.141667
1981	0.323333
1882	-0.1025
1954	-0.1325
2014	0.746667
1965	-0.1075
1983	0.315
2003	0.621667
1952	0.01
1895	-0.228333
1945	0.090833
1972	0.008333
1890	-0.349167
1924	-0.271667
1988	0.390833
1978	0.069167
2009	0.661667
1881	-0.075
1995	0.446667
1903	-0.366667
1912	-0.356667
1934	-0.123333
2010	0.725833
2005	0.681667
2006	0.640833
1883	-0.165833
1935	-0.199167
2012	0.646667
1960	-0.026667
1950	-0.174167
1979	0.165833
1915	-0.141667
1893	-0.313333
1951	-0.069167
1962	0.03
1967	-0.023333

Results in Vegas-Lite:

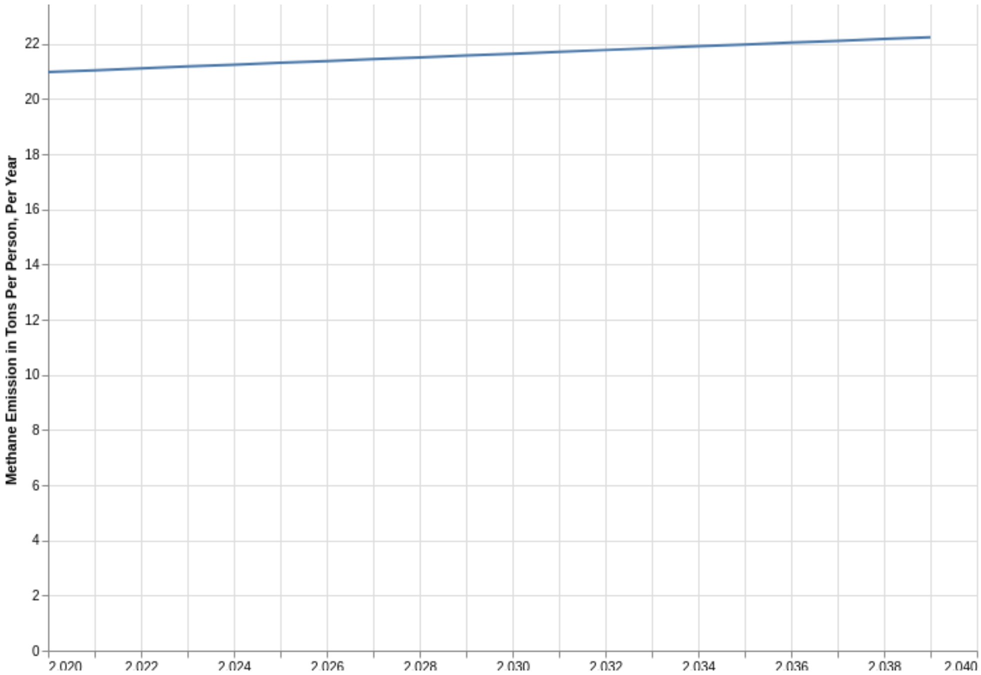


2 Predicted:

year	co2predicted	methanepredicted	nopredicted
2033	4.99212	21.84067	3.8485
2037	5.10371	22.10719	3.88897
2039	5.15506	22.24046	3.9092
2025	4.76883	21.30762	3.76755
2032	4.96871	21.77404	3.83838
2027	4.818	21.44088	3.78779
2022	4.68273	21.10768	3.73722
2030	4.90773	21.64077	3.81814
2023	4.70559	21.17436	3.74732
2028	4.85825	21.50751	3.79791
2038	5.13196	22.17383	3.89909
2034	5.01992	21.9073	3.85861
2036	5.07941	22.04056	3.87885
2035	5.04275	21.97393	3.86873
2020	4.6383	20.97412	3.72234
2024	4.74809	21.24098	3.75744
2026	4.79535	21.37425	3.77767
2031	4.93039	21.70741	3.82826
2021	4.65706	21.04105	3.7269
2029	4.8805	21.57414	3.80803

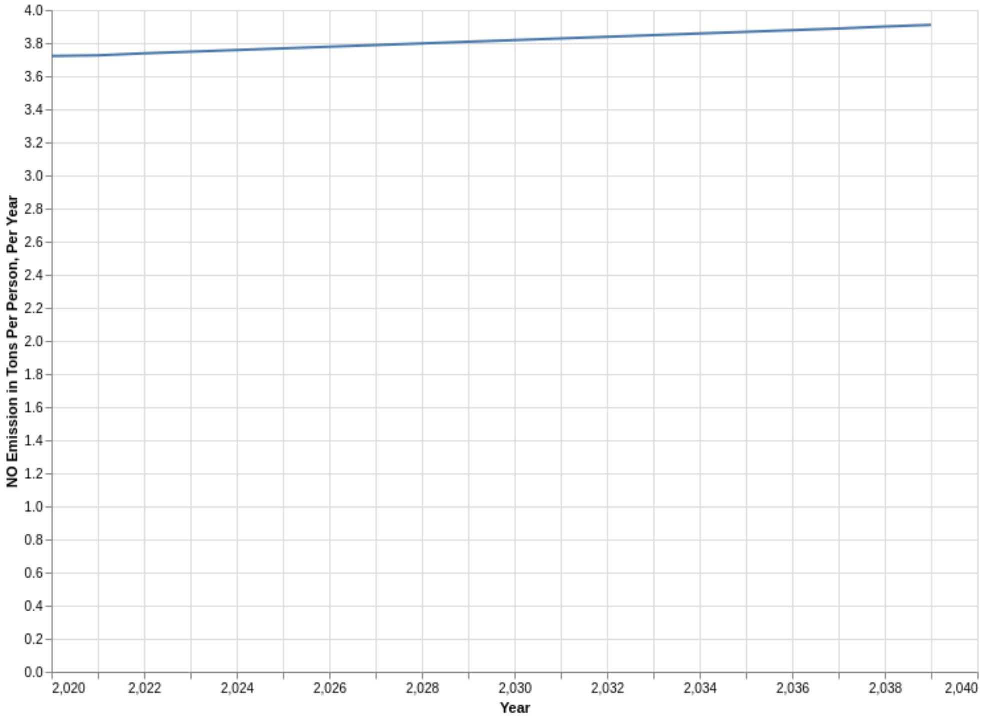
2.1 Methane Prediction Results:

Results in Vegas-Lite:



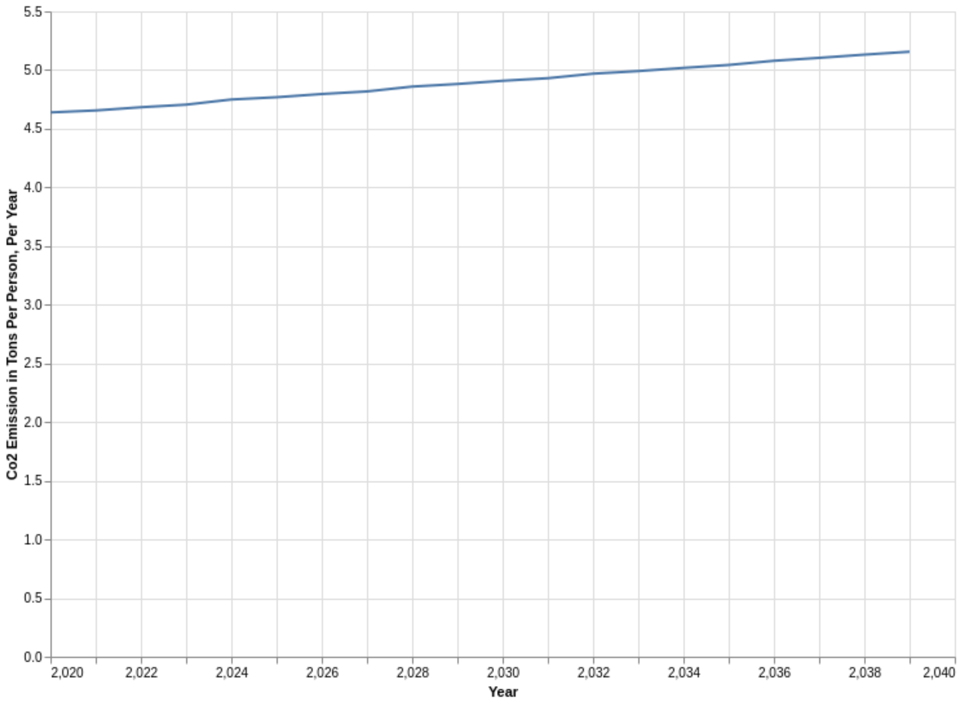
2.2 Nitrous Oxide Prediction Results:

Results in Vegas-Lite:



2.3 Co2 Prediction Results:

Results in Vegas-Lite:



References

1. Datasets as endpoints, available at: https://global-warming.org/
2. "Observed Arctic sea-ice loss directly follows anthropogenic CO2 emission" by Dirk Notz and Julienne Stroeve, published on 11 Nov 2016, available at: https://www.mpg.de/10817029/my-contribution-to-arctic-sea-ice-melt
3. "Scala Programming Language", available at: https://www.scala-lang.org/
4. "Play framework", available at: https://www.playframework.com/
5. Cassandra, available at: https://cassandra.apache.org/
6. Apache Spark, available at: https://spark.apache.org/
7. Apache Spark MLlib, available at: https://spark.apache.org/mllib/
8. ARIMA, available at: https://jar-download.com/artifacts/com.cloudera.sparkts/sparkts/0.4.1/source-code/com/cloudera/sparkts/models/ARIMA.scala
9. Vegas-viz, available at: https://github.com/vegas-viz/Vegas
10. Vega-lite, available at: https://vega.github.io/editor/#/
11. Units conversion, available at: https://www.engineeringtoolbox.com/ppm-d_1039.html