

Predicting NC Air Quality Index

Team 4

Angela Arce
Tammy Geis
Hanita Patel
Spencer Pope

August 25, 2022

Table of Contents - Segment 3 Deliverable

Slide 3: The Project - Predicting NC Air Quality Index (AQI)

Slide 4: The Data Questions

Slide 5: The Dataset

Slide 6: Database Details

Slide 7 - Slide 10: Machine Learning Model

Slide 11: Analysis Process

Slide 12: Data Visualization

Slide 13 - Slide 14: Technologies Used In Project

Predicting NC Air Quality Index (AQI)

The WHY

- Assist people with respiratory illnesses to determine if safe to engage in outside activities
- Information for people/families moving to NC to determine which region may best suit respiratory medical needs



The HOW

- Using various tools and Kaggle dataset predict AQI in regions across NC based on time of year
- App based tool for ease of use in Phase 2

The Data Questions



Does Air Quality vary by time of year?

What AQI is safe/unsafe for the respiratory system?

Does population density have an effect on AQI?

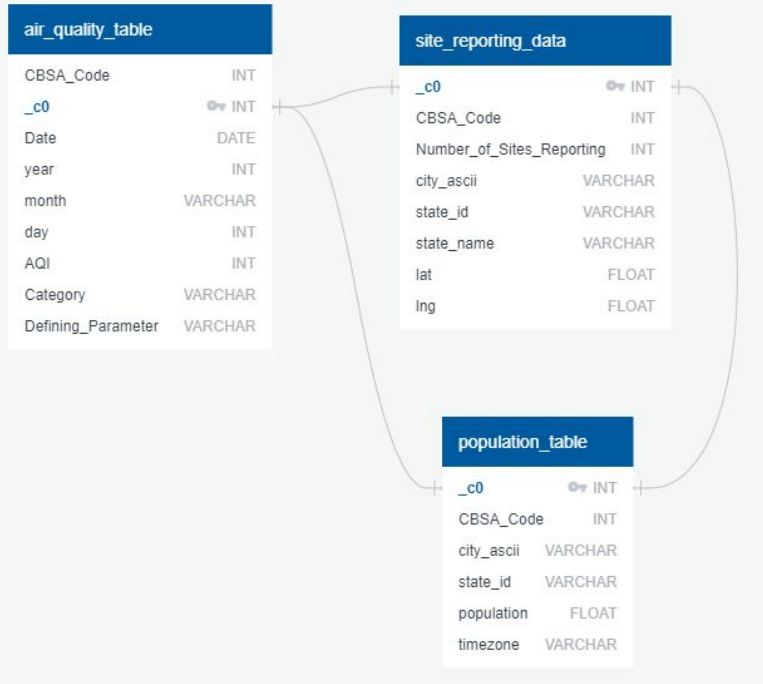
Does location have an effect on AQI?

What region of NC best suits someone with respiratory illness?

The Dataset

“US Air Quality 1980 - Present: Daily AQI Values from stations across the US” Source: Kaggle

www.quickdatabasediagrams.com



Isolated data for NC from dataset
to import to database as shown
in database schema

Database ETL Details

AWS RDB and S3 bucket created to store original csv data file

Google Colab ETL File used Pyspark to create data frame

Data frame schema was updated to align with database ERD

Data frames were created for the tables: AirQuality, Site Reporting and Population

Data frames were written to the database in Postgres Sql

Sample Code

```
from pyspark.sql import SparkSession
spark = SparkSession.builder.appName("Team4-Project").config("spark.driver.extraClassPath", "/conte
```

```
# Read in data from S3 Buckets
from pyspark import SparkFiles
url = "https://geisteam4-project.s3.amazonaws.com/ncaqi.csv"
spark.sparkContext.addFile(url)
user_data_df = spark.read.csv(SparkFiles.get("ncaqi.csv"), sep=",", header=True, inferSchema=True)
```

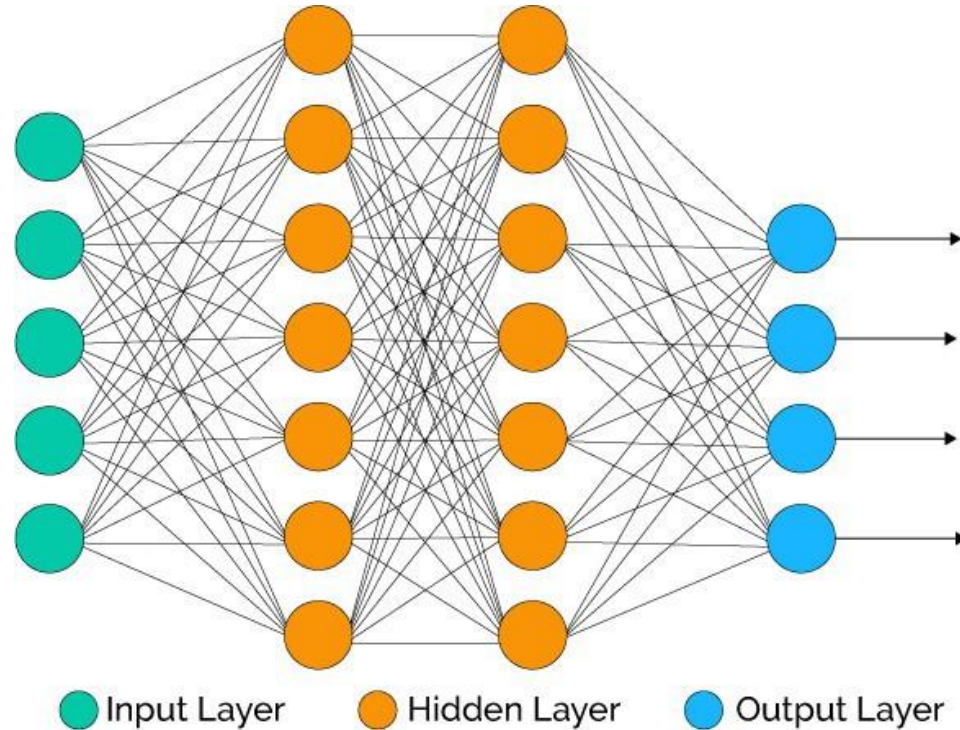
```
# Configure settings for RDS
mode = "append"
jdbc_url="jdbc:postgresql://geisteam4.coe2ggfhl77s.us-east-1.rds.amazonaws.com"
jdbc_url="jdbc:postgresql://geisteam4.coe2ggfhl77s.us-east-1.rds.amazonaws.com:5432/postgres"
config = {"user": "postgres",
          "password": "xx",
          "driver": "org.postgresql.Driver"}
```

```
# Write airquality_df to table in RDS
airquality_df1.write.jdbc(url=jdbc_url, table='air_quality_table', mode=mode, properties=config)
```

```
# Write site_reporting_df to table in RDS
site_reporting_df1.write.jdbc(url=jdbc_url, table='site_reporting_table', mode=mode, properties=config)
```

```
# Write site_reporting_df to table in RDS
population_df1.write.jdbc(url=jdbc_url, table='population_table', mode=mode, properties=config)
```

Machine Learning Model: Neural Network



Why a Neural Network?

- Can detect complex and nonlinear data
- Able to handle messy data
- We are working with a dependent output (AQI Category) that we aim to predict with various input variables

Limitations to Neural Networks:

- Can be prone to overfitting

Machine Learning Details

Connected to the database in PgAdmin to load the data into jupyter notebook

Selected columns we thought would have the greatest impact on predicting the AQI value category

Categorized the AQI value column into 1-6 values

1 = Good, AQI: 0-50

2= Moderate, AQI: 51-100

3= Unhealthy for Sensitive Groups,
AQI:101-150

4= Unhealthy, AQI:151-200

5= Very Unhealthy, AQI:201-300

6= Hazardous, AQI:301+

```
#Creating connection to postgres
```

```
engine = pg.connect(  
    database="postgres",  
    user="postgres",  
    password="geisadmin01",  
    host="geisteam4.coe2ggfhl77s.us-east-1.rds.amazonaws.com",  
    port='5432'  
)
```

```
#Loading in data
```

```
air_qual_df = psql.read_sql('SELECT * FROM air_quality_table', engine)  
site_df = psql.read_sql('SELECT * FROM site_reporting_table', engine)  
population_df = psql.read_sql('SELECT * FROM population_table', engine)
```

```
#pulling data from the tables to pass into machine learning model
```

```
ml_df = [air_qual_df[['year', 'month', 'aqi']],  
         site_df[['lat', 'lng']],  
         population_df[['population', 'density']]]  
ml_df = pd.concat(ml_df, axis = 1)  
ml_df.head(50)
```

```
def aqi(x):  
    if x <= 50:  
        return 1  
    elif x >=51 and x <= 100:  
        return 2  
    elif x >=101 and x <= 150:  
        return 3  
    elif x >=151 and x <= 200:  
        return 4  
    elif x >=201 and x <= 300:  
        return 5  
    elif x >=301:  
        return 6  
    else:  
        return 1
```

```
ml_df["AQI"] = ml_df["aqi"].apply(lambda x: aqi(x))  
ml_df.head(50)
```


Machine Learning: Neural Network

X and y variables determined:

X: all columns of the dataframe except for the AQI Category column

Y: AQI category value the model is aiming to predict

Scale the data:

Utilized MinMaxScaler

The data was then split the data into train and test sets

```
❏ y= ml_df.AQI
   x= ml_df.drop(columns=["AQI"]).values

   x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_state = 42)

❏ # Create a Scaler instances
   scaler = MinMaxScaler(feature_range=(0,1))

   # Fit the minmaxScaler
   x_scaler = scaler.fit(x_train)

   # Scale the data
   x_train_scaled = x_scaler.transform(x_train)
   x_test_scaled = x_scaler.transform(x_test)

   x_train_scaled = np.reshape(x_train_scaled, (x_train_scaled.shape[0],1,x_train_scaled.shape[1]))
   x_test_scaled = np.reshape(x_test_scaled, (x_test_scaled.shape[0],1,x_test_scaled.shape[1]))
```

Neural Network Model

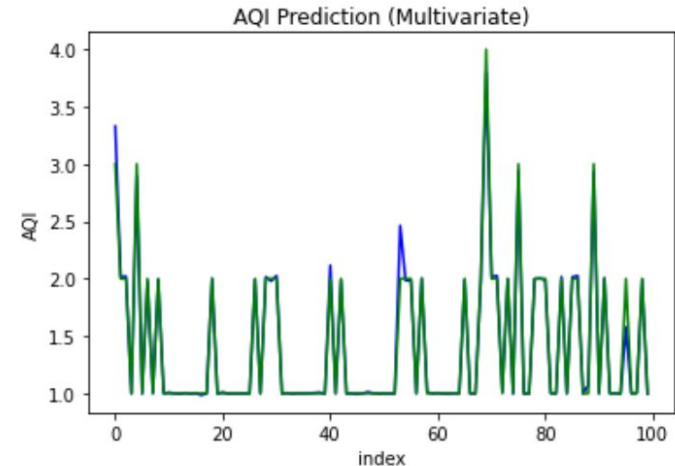
- Utilized a Keras Model
- Two hidden layers were used
 - First hidden layer: 70 nodes
 - Second hidden layer: 30 nodes
- Dropout layer of 0.2

After running the model for 25 epochs it returned a loss of 0.02 and had an accuracy of 64%

Model: "sequential_2"

Layer (type)	Output Shape	Param #
lstm_4 (LSTM)	(None, 1, 70)	21840
dropout_4 (Dropout)	(None, 1, 70)	0
lstm_5 (LSTM)	(None, 1, 30)	12120
dropout_5 (Dropout)	(None, 1, 30)	0
dense_2 (Dense)	(None, 1, 1)	31

=====
Total params: 33,991
Trainable params: 33,991
Non-trainable params: 0



Analysis Process

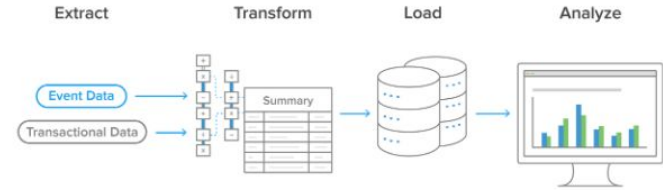
Step 1 Requirements

- BrainStorming - team used this technique to discuss various ideas to determine project idea
- Requirements - identified information and data source needed to set up database, generate machine learning model and data visualization

Step 2 Conceptual Design

- Technologies - outlined technologies to be used in project. Included AWS, Google Colab, Postgres SQL, Tableau, Neural Network
- Database ERD - using data source set up the database ERD and schema
- Machine Learning - reviewed various models supervised, unsupervised and neural network with final decision to pursue neural network
- Data Dashboard - preliminary data visualizations decided to guide data needs

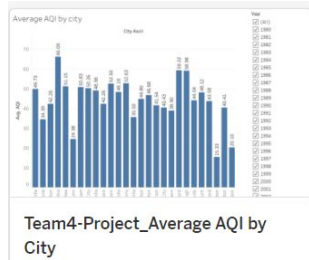
Step 3 Programming, Testing and Analysis



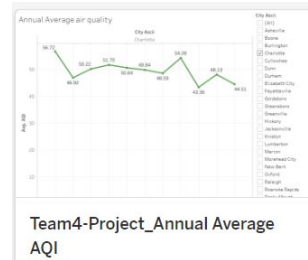
- Data was extracted from Kaggle and stored using AWS
- PySpark within Google Colab notebook was used to transform the data. Data elements modified to match data schema including data types and column names.
- Dataframes set up for each table
- Dataframes loaded to SQL database
- Storyboard created and Tableau used for dashboard
- Machine learning model to predict AQI categories based on Keras model

Data Visualization

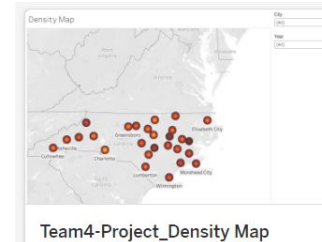
Average AQI by City
Purpose to show AQI based on population of city



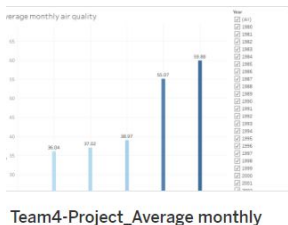
Annual Average AQI
Purpose to show AQI over time by city or compare cities



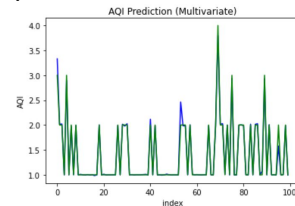
Density Map
Purpose to show visually AQI based on density by city



Average Monthly AQI
Purpose to determine if AQI varies based on time of year



Connect Machine Learning Model to Tableau for Prediction
Purpose to show accuracy of model



<https://public.tableau.com/authoring/Team4NCAQIPrediction/Team4NCAQIPrediction#1>

Link to Tableau Dashboard

Tool for Visualization/Dashboard: Tableau

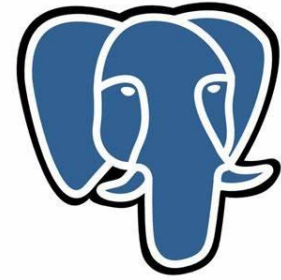
Interactive element: Filtering by city or date on various tables in dashboard

Technologies Utilized

Analyzing/Cleaning Data



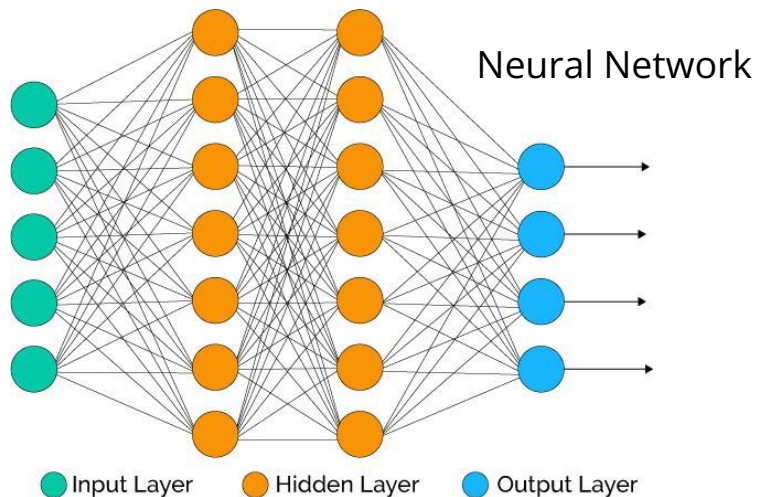
Database



PostgreSQL

Technologies Utilized

Machine Learning Model



Dashboard



Other

