Weather Prediction Modeling

Team Members: Khushi Magiawala, Somil Jain

Manav Ramprasasd, Sarang Pujari

CX 4230: Computer Simulation

Dr. Richard Vuduc

TABLE OF CONTENTS

1. Abstract **2**

2. Introduction **3**

3. Literature Review **4**

4. Conceptual Model **6**

5. Simulation Model **7**

6. Weather Prediction UI **15**

7. Discussion/Conclusion **20**

8. Appendix: Division of Labor **22**

9. References **23**

Abstract

Our goal for this project was to use labeled weather data to predict specific conditions (i.e. Humidity, Temperature, Precipitation, etc.) or some combination of these weather features to provide insight into what type of weather (trends, outliers, correlations, etc.) we experience in Atlanta year-round. We hoped that the conclusions could also give us some insight into storm data. We created a dataset of the weather data collected by Hartsfield-Jackson International Airport Station for the past 3 years to train our models and create our differential equations. We included 3 years worth of data to ensure our predictions are statistically significant and relevant to today's weather trends.

When making our conceptual model, we considered using Markov Chains, but eventually settled on regressions models and ODE's, since they provided a more straightforward path given our data.We used Auto Regressive Integrated Moving Average models (ARIMA) to forecast temperature by creating differential models based on temperature as a function of time.This allows us to forecast weather of future years more accurately. Linear regression models were also employed to find a relationship between humidity and air temperature. What was found was an inverse relationship, which confirmed the research we had done prior. We then used a combination of polynomial regression and differential equation building to make predictions on air temperature, precipitation, and dew point temperature based on each of their relevant factors. These initial predictions were then used with a linear regression model to build a final prediction model, which was significantly more accurate than the initial prediction models.

Finally, we created an interactive weather prediction UI for users to pick a feature for the model to predict and then input amounts for the variable weather features in the predicted feature's differential equation. Here is a link to our repository: https://github.gatech.edu/kmagiawala3/CX4230-Project

Some tools, libraries and frameworks we used in this project are React.js, python, VSCode, Google Colab, Flask, Heroku, Juypter Notebook, Pandas, Sci-kit learn and AIRMA. All work between members was split evenly based on strengths, weaknesses and interests. All members helped with the write-up, formatting, and submission of this project.

Introduction

The most recent Intergovernmental Panel on Climate Change (IPCC) report stressed the importance of reliable predictions of extremes for short and long time scales to reduce potential risks and damages that result from weather and climate extremes. Understanding, modeling and predicting weather is identified as a major research area. Through our modeling of weather data, we hope to understand more about how to predict and forecast.

Our goal for this project is to use labeled weather data to predict specific conditions (i.e. Humidity, Temperature, Precipitation, etc.) or a combination of these conditions that may provide insight into storm data. For our model, we used the Atlanta Weather Dataset provided by the Weather station at the Hartsfield–Jackson Atlanta airport. The data is provided by the website "Reliable Prognosis", rp5.ru. The dataset is a collection of weather data for each day it has been collected in the past 3 years (01.02.2019 till 04.02.2022, all days). There are approximately 63312 data points. The dataset is cleaned to highlight the 14 main columns that we are analyzing. The most relevant data columns include Air Temperature, precipitation, humidity, pressure, and dew point temperature.

Literature Review

Our group was interested in Numerical weather prediction (NWP) to predict weather conditions such as humidity, pressure, precipitation, etc. NWP uses mathematical and statistical models of the atmosphere and oceans to predict future weather patterns. Data inputs are collected from weather satellites and radiosondes and then fed into models to output a realistic simulation. In the simplest form, NWP asks you to sample a weather feature (ex: atmosphere fluid) and then use partial differential equations (ex: fluid dynamics equations) to predict the state of said feature in some future time step. For computation and modeling, scientists use primitive equations along with other factor equations, such as the ideal gas law, density, and pressure (*Numerical weather prediction,* 2022). One big challenge for our project will be scoping our problem well enough to identify all the different factor questions necessary to make weather predictions; this may vary between the domain of our model (global or regional). Some key factors that affect the accuracy of numerical weather predictions are the quality and density of the data points collected. Handling errors and increasing accuracy - model output statistics (multiple linear regression techniques to relate near-surface qualities, such as wind direction and air temperature, to one or more predictors (forecasts from NWP) (*Model output statistics,* 2022).

In his blog, Sebastian Callh details how neural ODE's can be used to model different weather patterns. The article walks us through how climate data was processed for the training models, and then how the models were actually trained. It also shows how as more training data is added to the model, the model fits itself more and more to the data being input. The article shows how temperature, humidity, wind speed and pressure can all be modeled using natural ODE's with initial values (Callh, 2020).

One specific weather case we potentially are interested in modeling and simulating is the occurrence of lightning strikes in various weather/storm conditions. According to the NOAA National Severe Storms Laboratory, predicting the occurrence of individual lightning strikes can be difficult due to the randomness of strikes, but one can study the cloud electrification processes and ingredients of a lightening prone storm. Right now, scientists use current weather features (storm age, size, temperatures, etc.) to predict whether a storm is likely to produce lightning. Currently, "NSSL researchers are using a 3-D cloud model to investigate the full life-cycle of thunderstorms" and running data points through stats and ML models (*Lightning forecasting,* 2021). Additionally, NASA also developed the GOES-R Geostationary Lightning Mapper (GLM) which uses AI and satellite data to identify current and predict future lightning strikes (*NOAA Satellite Data Predict Future Lightning Strikes*, 2021). This concept could be applied to our own version of prediction for weather forecasting to provide more insight into general storm data and metrics.

One application of Numerical Weather Prediction utilizes the Navier Stokes equation to define the fluid motion through various factors: pressure, fluid velocity, density, temperature, and humidity. Eventually, more derivations and mathematical equations lead to the use of the hydrostatic equation (Lynch, 2006). The article delves into various details of how NWP has emerged as a leading forecasting prediction as discussed through topics such as ENIAC integrations, time-stepping schemes, spatial finite differencing, spectral method, and more. These mathematical equations could be applied to our own version of NWP.

Conceptual Model

Since we are dealing with weather, a very observable set of phenomena, we need to make sure whatever model we are intending to build fits the dynamic of the weather variables in question. Since there is an abundance of information on the relationship between different weather phenomena, we realized that we can incorporate some of what we had learned into the model very easily. What we figured we needed was a couple of different approaches to predicting future data.

We initially thought about trying to create a weather Markov Chain that could continuously predict values as days go on. The issue with this is there didn't seem to be enough overlapping relationships between the data, and so constructing a Markov Chain seemed too difficult. The next set of ideas were a combination of ODE's we thought we could create, and regression models that may be appropriate given what we had seen from the data so far.

These ideas proved fruitful in combination with each other, as using polynomial regression models with ODE's exposed a few relationships that we hadn't found yet. We knew about fundamental weather concepts, such as how air pressure and humidity are positively correlated with each other, but negatively correlated with air temperature. Additionally we found strong relationships between humidity, air pressure and precipitation that confirmed research we had done.

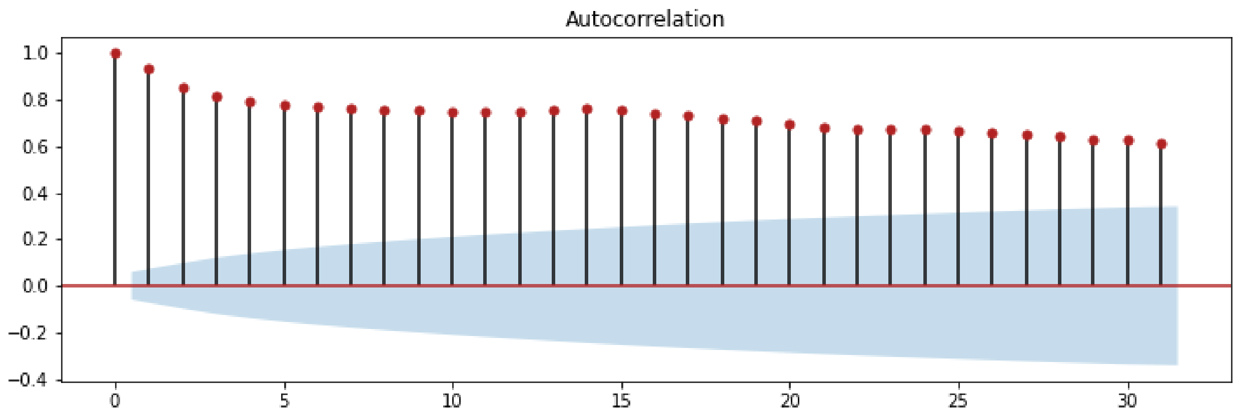
One consideration we did make for this model was the possibility of impossible predictions. For example, there is a strong likelihood that negative precipitation may be predicted due to the number of days with zero precipitation. Issues like this could have been solved with a Markov Chain, but we realized that a negative precipitation prediction can actually just be set to zero, and the correlation between our prediction and the actual values actually improves. While this wasn't the most robust approach, the predictions ended up being significantly more accurate than any of the initial relationships between the data provided.

Simulation Models

Time Series Auto Regressive Integrative Moving Average (ARIMA)

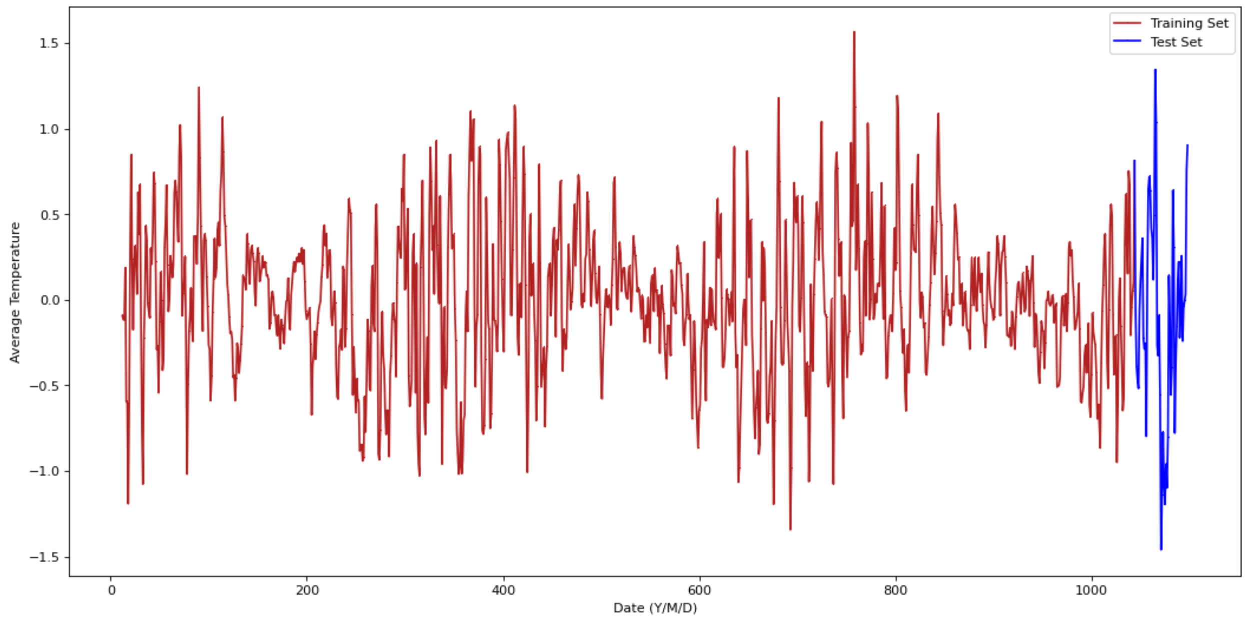
We decided to utilize statsmodels' Auto Regressive Integrated Moving Average models (ARIMA) to forecast the Air Temperature data field. As part of the data cleaning, we averaged the temperature data for each date to make it based on the calendar date and further synthesize the data. To account for seasonality, we employed a Seasonal ARIMA (SARIMA) Model.

To analyze the data, we plotted the average temperature on an autocorrelation plot. The autocorrelation plot below describes the autocorrelation between an observation and another observation at a prior time step that includes direct and indirect dependence information. After analyzing the graph, we are able to see that the data fails the randomness test as most of the autocorrelation graph is far away from 0, meaning the autocorrelation between adjacent and near-adjacent observations is high.

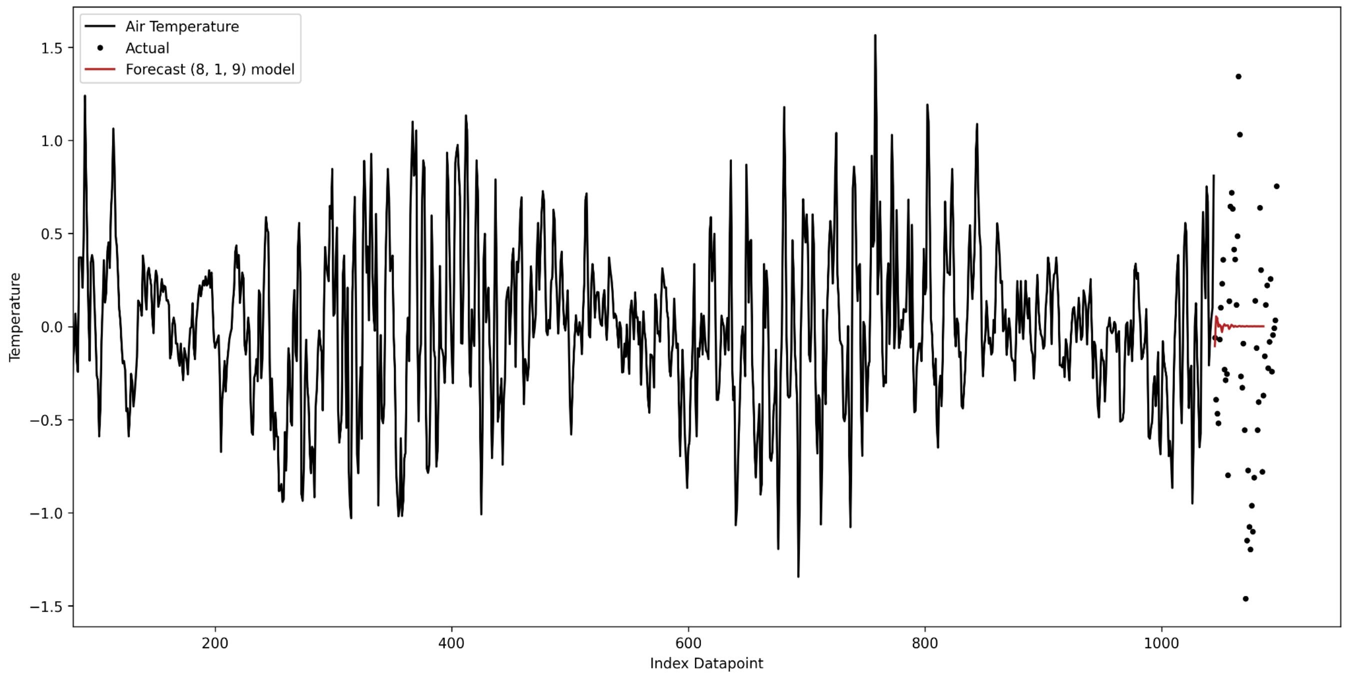


Next, the AD-Fuller test was performed to understand if the data was stationary. The p-value needs to be less than 0.05 to be stationary, however the calculated p-value is 0.4042806713817787. Because of this, we will employ a differentiated model (d = 1 within the ARIMA model). Also, a rolling mean differencing was implemented to reduce the p-value to the desired range.

Now, the dataset is ready for our model. 95% of the data (1044 observations) is set up as the training data, and the other 5% (54 observations) is for the test data.



The next step in the model is to test different combinations of the Autogressive Moving Average (ARMA) model of order p and q as well as the differentiating factor, d. The top two combinations were pinpointed by finding the lowest Akaike Information Criterion (AIC that determines which model is best fit). Finally, the p, d, q values for both as well as the training set is inputted into the SARIMAX model, and the forecasted graph is created to compare against the test data as shown below.

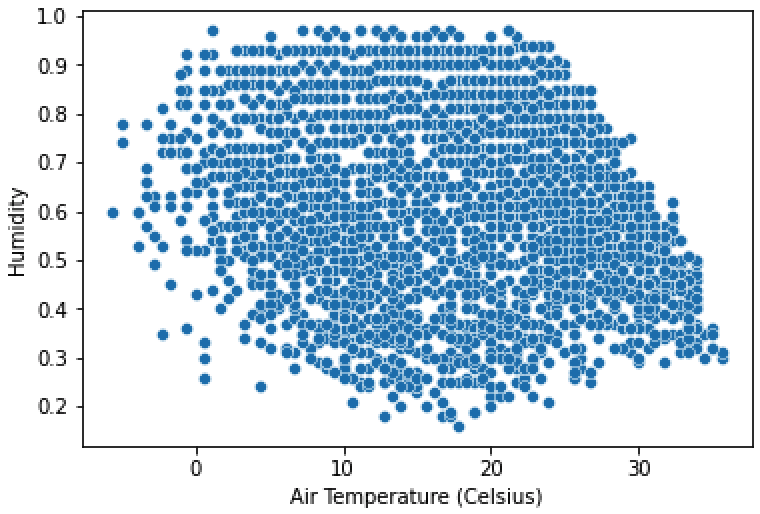


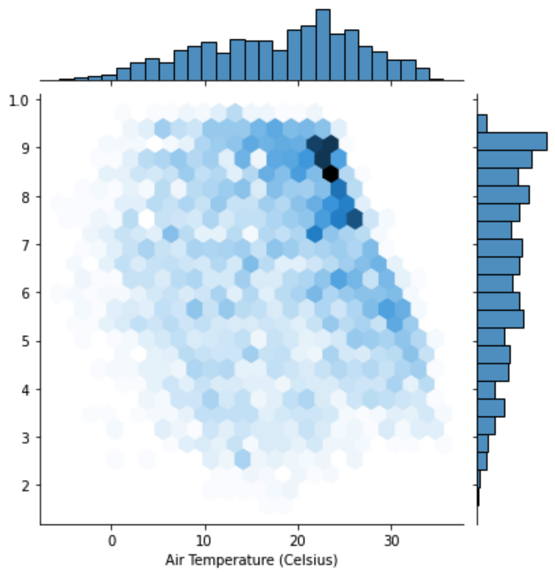
Temperature vs Humidity Linear Regression Modeling

For our model, we used the Atlanta Weather Dataset provided by the Weather station at the Hartsfield–Jackson Atlanta airport. The data is provided by the website "Reliable Prognosis", rp5.ru. The dataset is a collection of weather data for each day it has been collected in the past 3 years (01.02.2019 till 04.02.2022, all days). There are approximately 63312 data points.

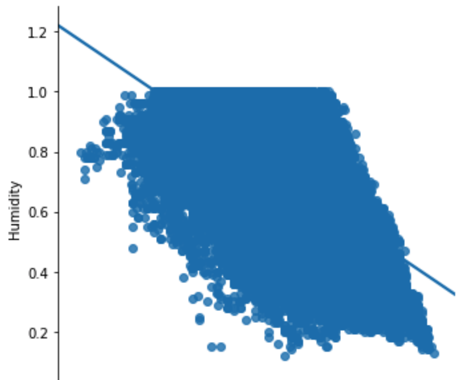
Given various parameters such as precipitation level, temperature, dew point, atmospheric pressure, and humidity, we aimed to find mathematical trends that could be extrapolated from these variables. For one approach, we focused on finding the relationship between temperature and humidity using a linear regression model. Linear regression is a mechanism in which we build a linear model which uses one or more than one input parameters(x1, x2, x3..) and uses them to determine an output(y). The form of the model mathematically is: y = w0 + w1x1 + w2x2…

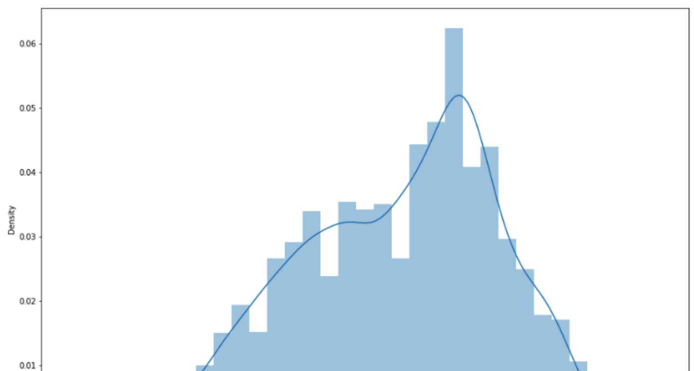
First, we attempted to gauge the relationship between humidity and temperature through various plots.



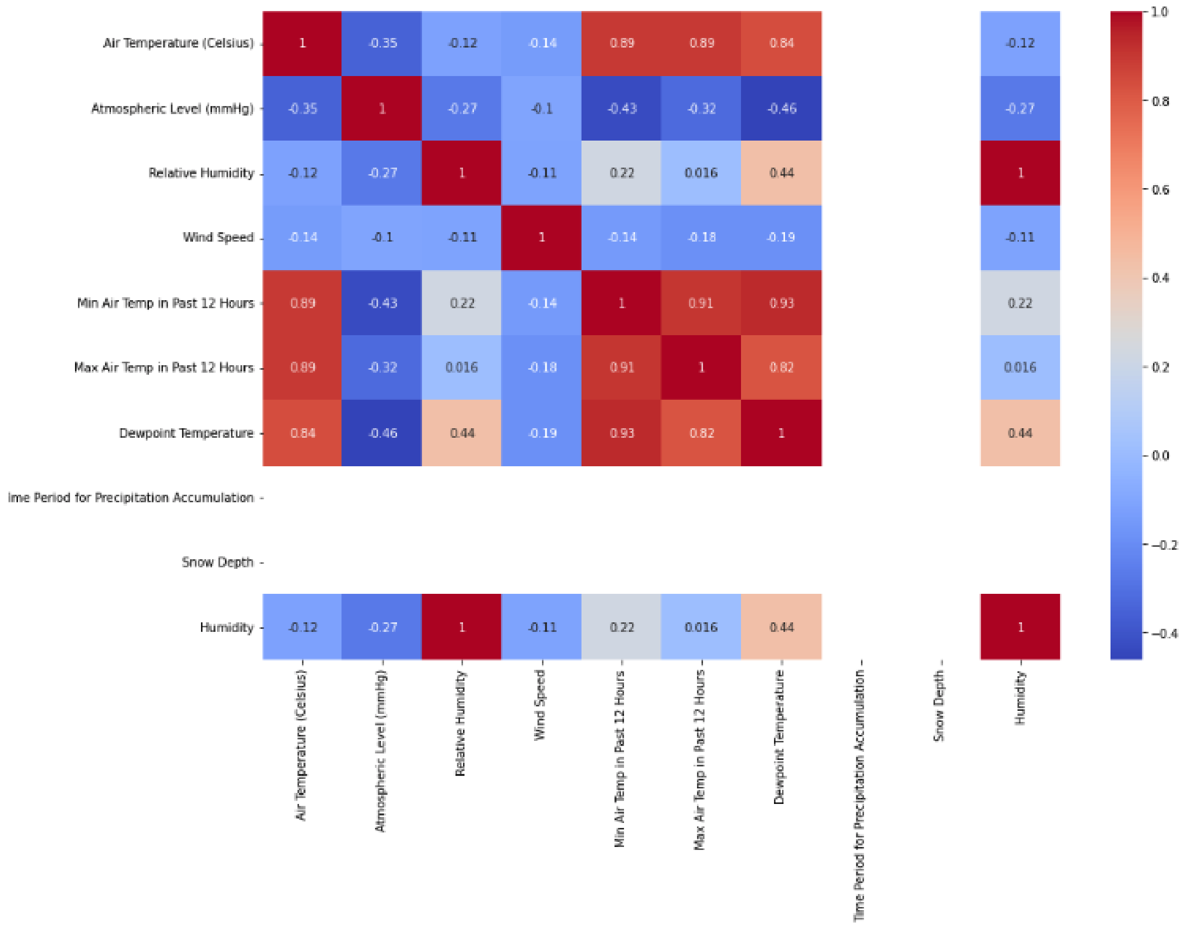


From these plots alone it was a little difficult to ascertain a relationship, but there does appear to be a somewhat negative correlation between humidity and temperature at first glance. This is confirmed by the linearity check between the two variables:



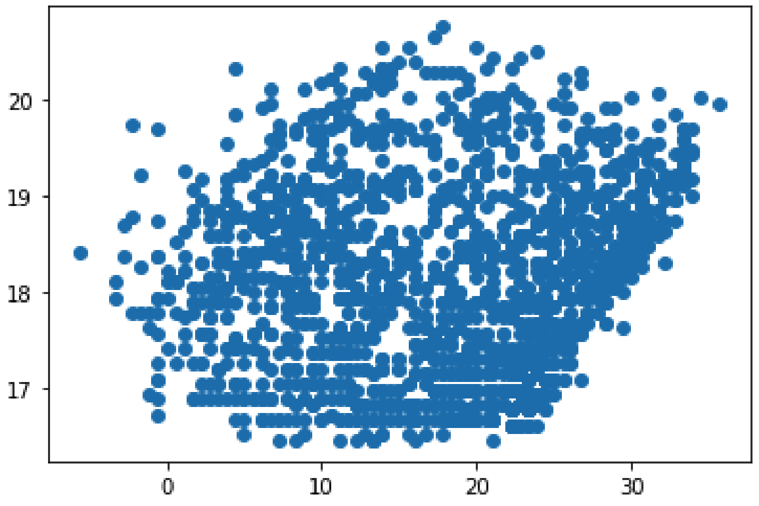


As a general check, we also conducted a correlation analysis between all the variables from our dataset.

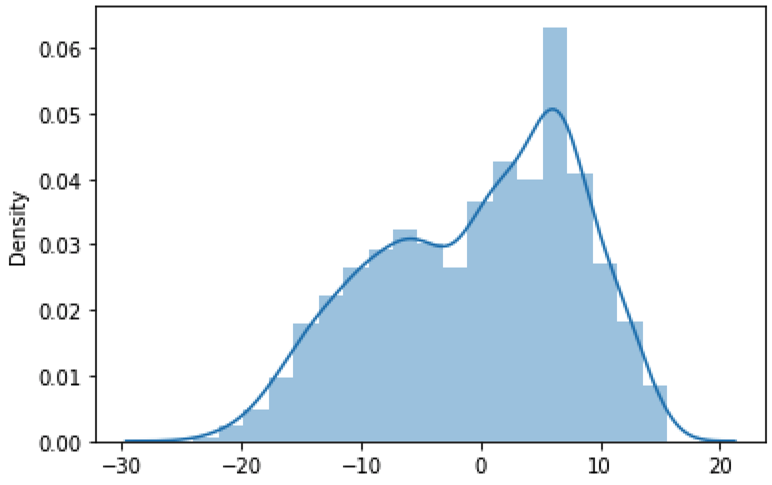


We notice that temperature and humidity do have a correlation, but some of the other variables such as atmospheric level and dew point, and atmospheric level and minimum air temperature have even stronger correlations. We explore some of these correlations in other models.

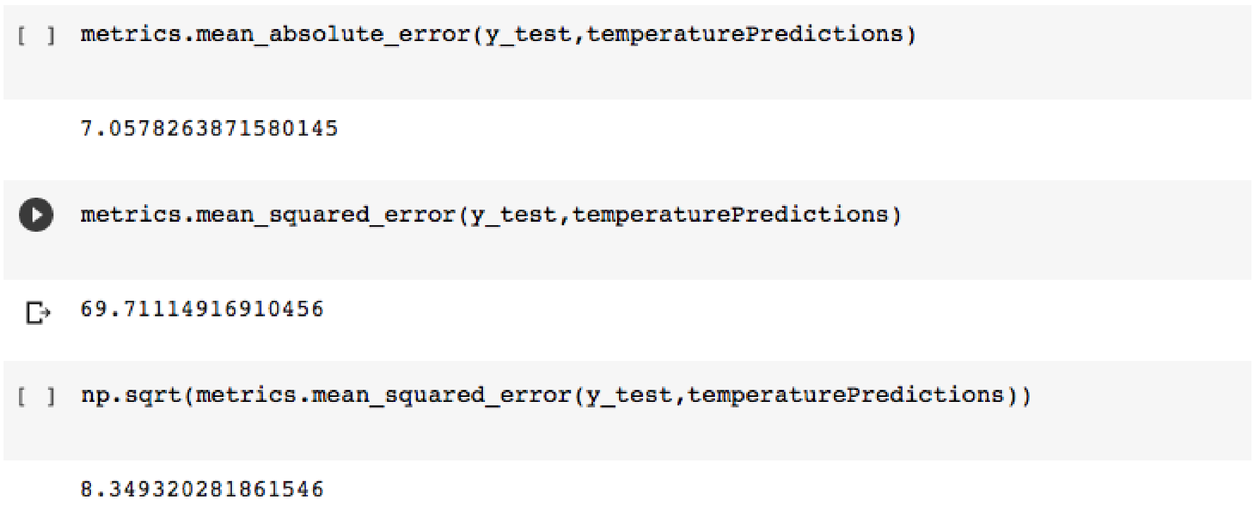
From here, we went into our data training and testing phase for the linear regression model. We assigned half the data to be the training dataset, and the other half to be testing. We used sklearn's Linear Regression framework to train and test our model. In comparing our actual and predicted values, we got this scatter plot. Evidently, there is some discrepancy but also significant overlap.



We then check the distribution of our residuals, by subtracting our temperature predictions from our temperature test data. We see from the distribution graph below that this gives us a fairly normal distribution.



With some additional error analysis we find that our mean squared error is a bit high, and perhaps other models would be more accurate for predictive purposes.

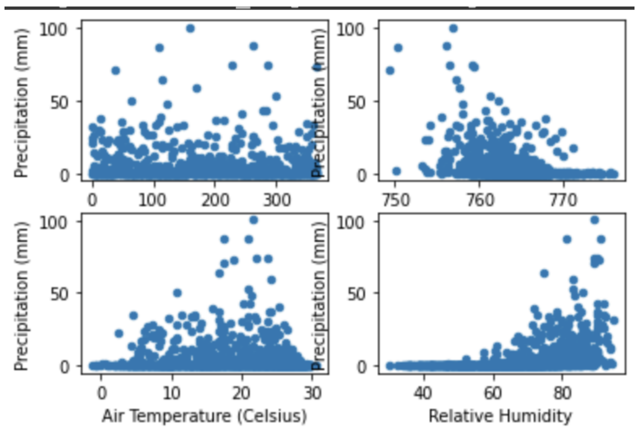


```
[ ] metrics.mean_absolute_error(y_test,temperaturePredictions)

    7.0578263871580145

    metrics.mean_squared_error(y_test,temperaturePredictions)

    69.71114916910456

[ ] np.sqrt(metrics.mean_squared_error(y_test,temperaturePredictions))

    8.349320281861546
```

We also calculate the coefficients of our linear regression equation using linearRegressionInstance.coef_, and get a value of -5.3081. This means that if humidity is increased by one unit, the temperature will reduce by 5.3081 units per our linear regression model.

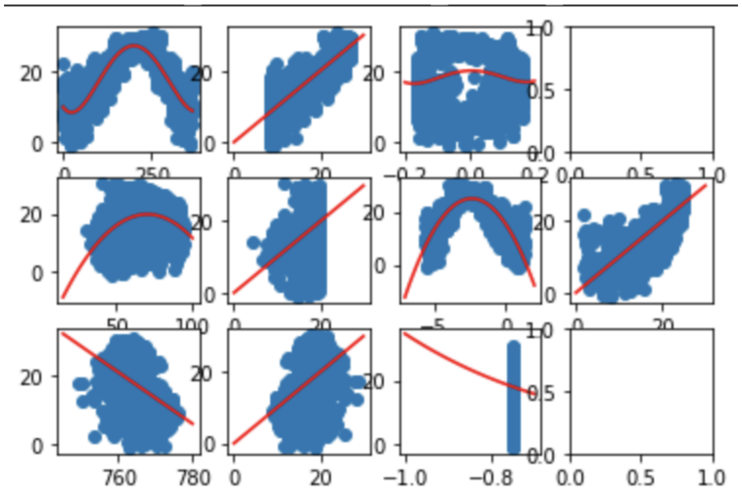
Differential Equations Regression Modeling

With an abundance of data and many weather related variables to consider, the first step was to clean the date and identify variables of interest. From the original data, we decided to remove and consolidate potential variables. For example, we averaged measurements from the same day so that the only time variable is the day of the year. Additionally, variables that were difficult to quantify were removed, and variables with ambiguous data were handled. For example, lowest cloud height was a range originally, but was changed to be the average of the range listed.

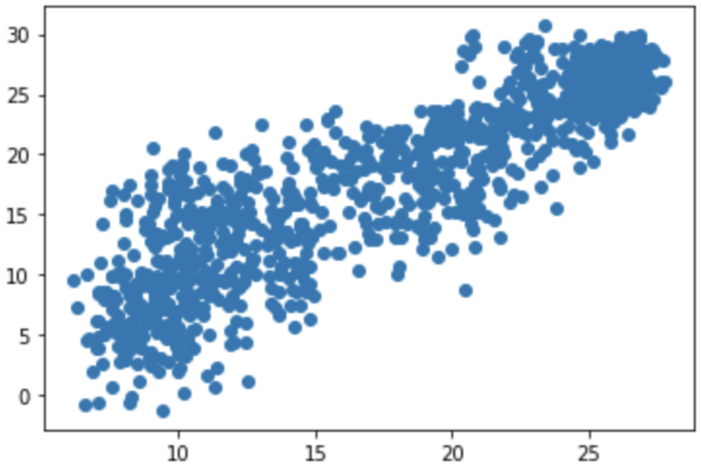


Next, we began to work on finding relationships between phenomena. We first graphed each variable against each other variable to determine some underlying relationships that may exist. Above is one such example for precipitation, where clearly some variables have more of a correlation than others. By doing this, we were able to clearly show that the temperature has a dependency on the date and the air pressure. After doing research online, though, we found that there is supposed to be a link between relative humidity and air temperature.

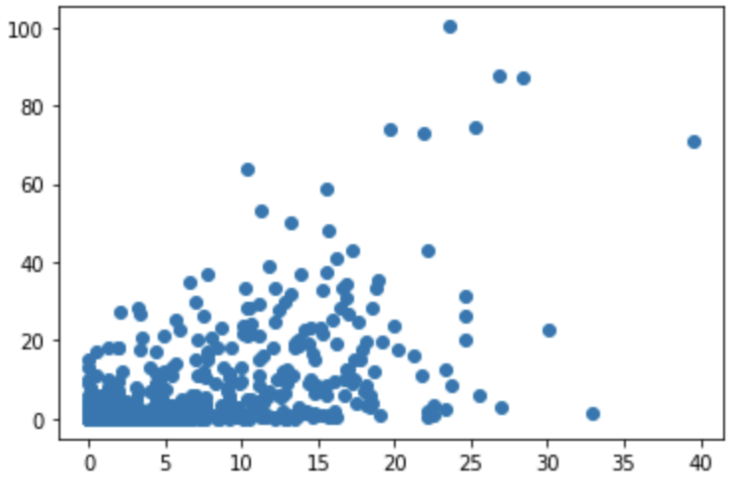
What we did next was to find the derivative of each of the variables that could potentially affect the temperature to see if there was any merit in those relationships. What we found was that while relative humidity doesn't have a relationship with air temperature when plotted with a scatter plot, its derivative with respect to air temperature has a clear relationship with air temperature itself. The graphs below show how date, relative humidity and air pressure, in that order, correlate with air temperature. The first column shows a simple scatter plot of the data in question. The second column attempts to fit that data properly. The third column is the derivative of each of the variables, where relative humidity was found to have a correlation, and was also subsequently fit to the air temperature data using polyfit.



The next step was to optimize these relationships so that we can effectively describe the air temperature as a result of the conditions of that particular day. To do this, we used polyfit to find polynomial relationships between these variables. What we can now do is take an input of date, air pressure, and humidity, and a temperature prediction for date, air pressure, and humidity can be determined. With these predictions, we can build a linear model that relates these three predictions to the actual air temperature values from our data, and using this model, we can make an even better prediction. The residuals for our final model for predicting temperature is shown below. The R value is 0.88, which shows a positive correlation is clearly present between our predictions and the actual value.



This process was replicated for precipitation predictions (shown below), which takes in relative humidity and air pressure as its parameters. These models were then exported, and along with the equations that calculate the inputs for the models based on the inputs from the user, the weather related predictions can be determined.



Weather Prediction UI

Overview

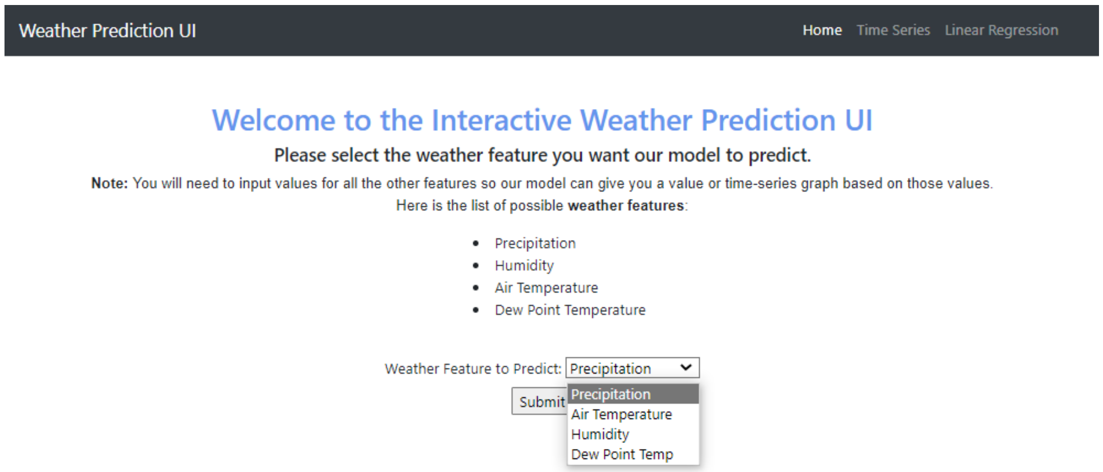
In order to showcase our modeling in an interactive way, we decided to create a weather prediction application that allows users to pick a feature for the model to predict and then input amounts for the variable weather features in the predicted feature's differential equation. To do this, we essentially figured out which features affect temperature or humidity the most, and then based on which feature the user selects for the model to predict, they can input values for those affecting weather features. Then, these values are sent back to our backend (models) and a predicted value is returned to the user. The goal is for the user to see if it would rain or what the temperature would be like based on other weather conditions, such as humidity and dew point temperature. The application has three main features: interactive weather prediction form, time series modeling results, and linear regression modeling results. The tools used for our code are React (frontend) and Flask-python (server-side) and Heroku API endpoints to communicate with the models.

Backend: Flask and Heroku

The interactive weather prediction user interface uses the Differential Equation Regression Modeling talked about in the previous section. In order to host the model, we first exported it as a pickle file and then put it in a Flask app. To have our backend running on the web instead of locally, we then put it on Heroku which gives us endpoints the frontend and backend can use to exchange information.

User Flow

The user starts the prediction-ui application using “npm start” and the Heroku service starts the Flask app which hosts our Differential Equation Regression Models. The user sees the Home Screen with directions on how to pick the appropriate weather feature they want our model to predict.



Welcome to the Interactive Weather Prediction UI

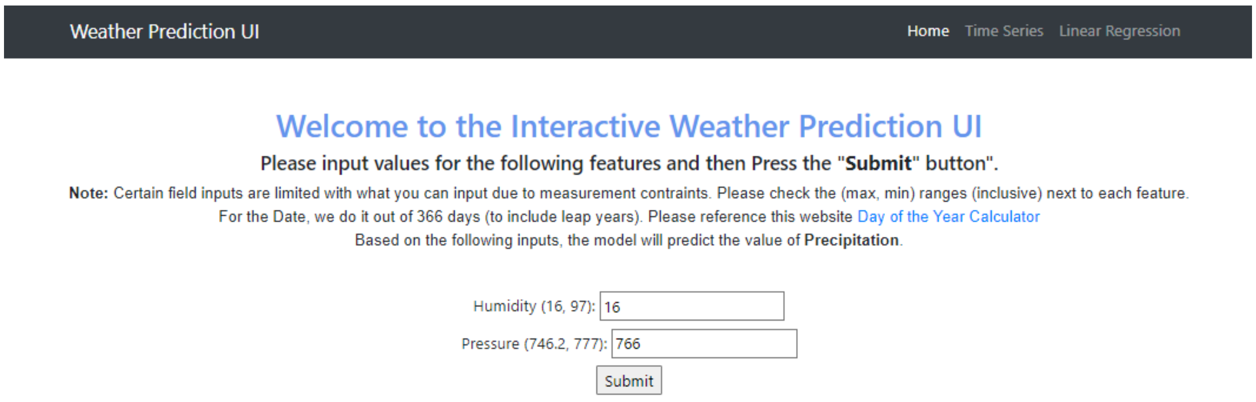
Please select the weather feature you want our model to predict.

Note: You will need to input values for all the other features so our model can give you a value or time-series graph based on those values.
Here is the list of possible **weather features**:

- Precipitation
- Humidity
- Air Temperature
- Dew Point Temperature

Weather Feature to Predict: Precipitation

Based on their selection, another form will appear for the user to input values for the other various weather conditions we predetermined through correlation analysis to affect their selected prediction weather feature the most (with input constraints for reasonable measurements).



Welcome to the Interactive Weather Prediction UI

Please input values for the following features and then Press the "**Submit**" button".

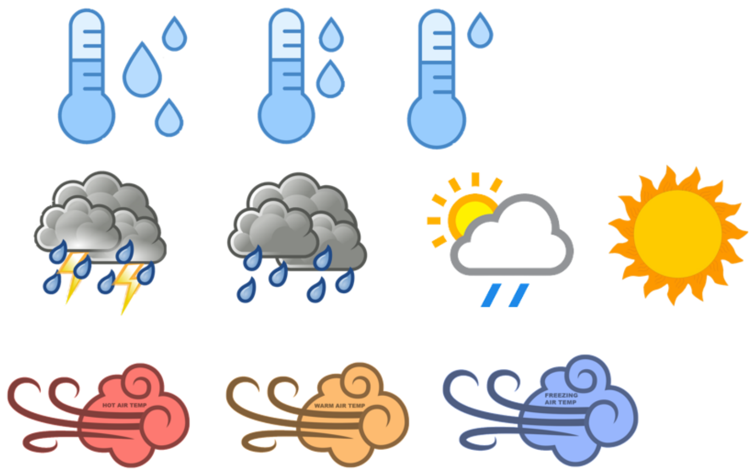
Note: Certain field inputs are limited with what you can input due to measurement contraints. Please check the (max, min) ranges (inclusive) next to each feature.
For the Date, we do it out of 366 days (to include leap years). Please reference this website Day of the Year Calculator
Based on the following inputs, the model will predict the value of **Precipitation**.

Humidity (16, 97): 16
Pressure (746.2, 777): 766

Once they press “Submit”, on the front-end, a POST call with these input values is made to the flask app which then feeds these values into the model. Once the model runs, that same POST call will return the predicted value to the front-end, which we then use to scale the visual representation of the predicted feature. Finally, they are taken to the Prediction Screen which simulates a day in Atlanta. The users will only see visual representations of the weather conditions they inputted values for and the predicted weather condition. Alongside the visual representations, they will see the value the model predicted and their input values.

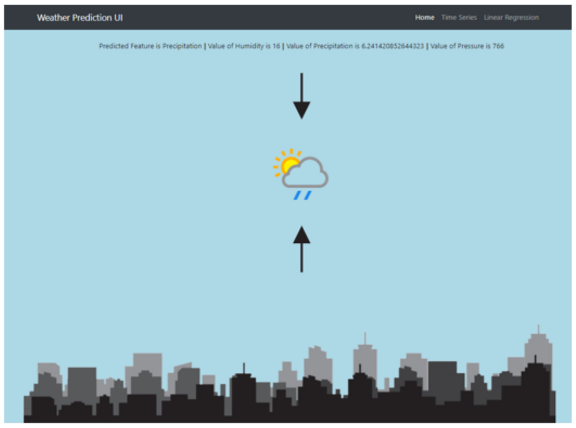
Visual Representations of Weather Features

Each feature has its own visual representation for the simulation. Our UI has visual representations of all 5 weather features even though not all of them will show up for each prediction. The UI shows the predicted value and the corresponding feature’s visual will change based on that value. As shown in the image below, the precipitation, dew point temperature, and air temperature features change based on how much there is. In the following section’s images, you will see these changes in the simulation.

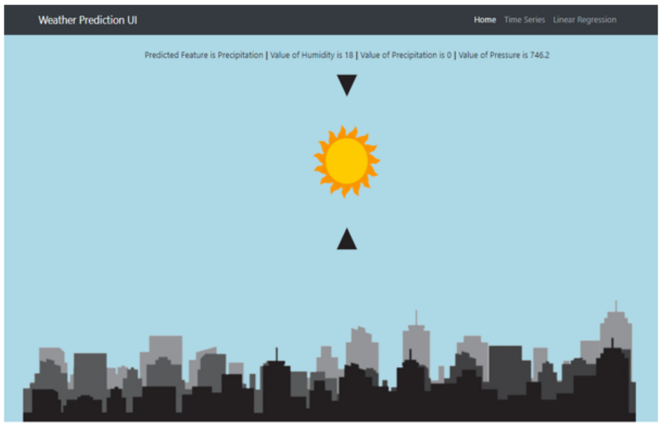


Trials and Validation

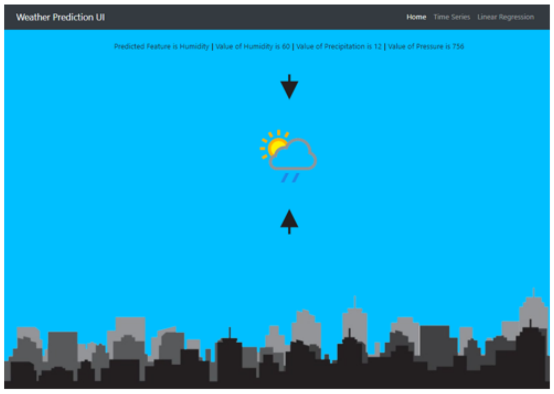
We played around with our user interface to see the different simulation variations we could get and if the prediction values made numerical sense based on how the visual representations looked on the screen. Here are two predictions for precipitation and then humidity side by side to show how the difference in input values and prediction values changes the feature's visual.



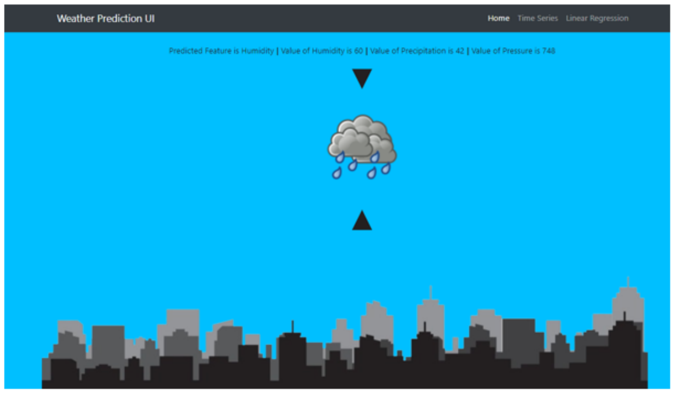
Predicted Feature is Precipitation | Value of Humidity is 16 | Value of Precipitation is 6.241420852644323 | Value of Pressure is 766



Predicted Feature is Precipitation | Value of Humidity is 18 | Value of Precipitation is 0 | Value of Pressure is 746.2

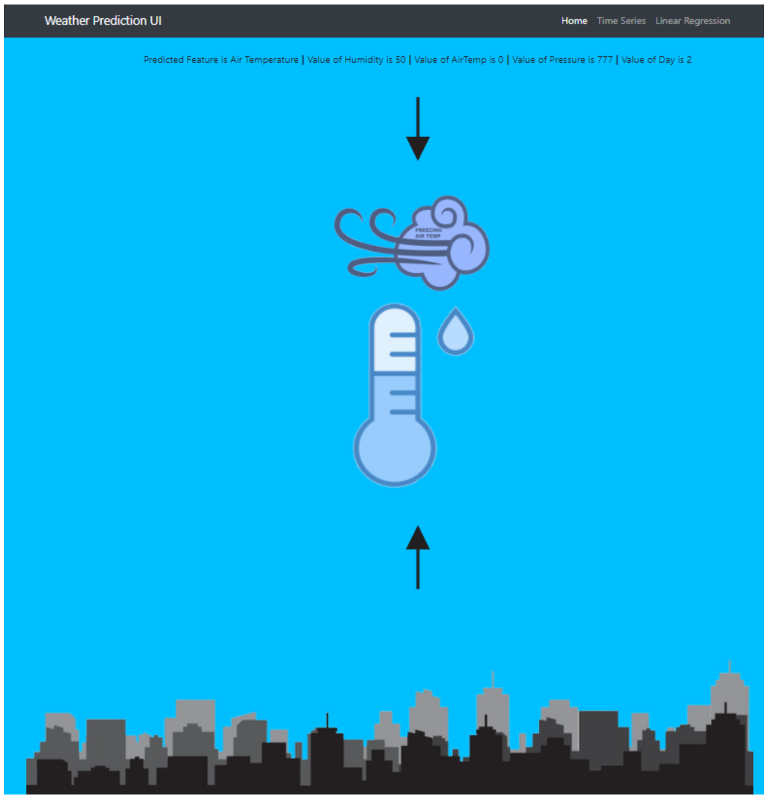


Predicted Feature is Humidity | Value of Humidity is 60 | Value of Precipitation is 12 | Value of Pressure is 756



Predicted Feature is Humidity | Value of Humidity is 60 | Value of Precipitation is 42 | Value of Pressure is 748

One interesting observation was when our model predicted the air temperature as 0 degrees celsius. At first, we thought it may be an error in our model, but then we realized that we chose the first day of the year for the prediction which is January. Thus, it very well could've been freezing as it was winter and Atlanta has had freezing temperatures at that time of year.



Predicted Feature is Air Temperature | Value of Humidity is 50 | Value of AirTemp is 0 | Value of Pressure is 777 | Value of Day is 2

Discussion/Conclusion

In this project, we aimed to simulate a realistic weather predictive model based on Atlanta Weather Data collected over three years. We utilized three different mathematical models to conduct our research: ARIMA Time Series Model, a Linear Regression Model measuring Temperature vs Humidity, and a Differential Equations Regression Model. Through these models, we aimed to not only predict variables such as Temperature, Humidity, Precipitation, and Dew Point, but also better understand the relationships between these weather indicators.

Each of our models demonstrated a different quality that furthered our understanding of the weather climate system. The Time Series ARIMA Model was able to forecast the temperature data to a certain extent. One aspect that this model highlighted is the importance of addressing seasonality and stationary data. At the same, it also was representative of how temperature is significantly influenced by other factors as well that are explored in other models. Our linear regression model implied a negative correlation between air temperature and humidity, with an increase in humidity causing a decrease in temperature by 5 units. While our residuals for this model showed a normal distribution, our mean squared error was a bit high suggesting the model can not exclusively be used for temperature prediction.

Perhaps the biggest takeaway from our simulation and its results is that there seems to be a pretty strong correlation between the changes in relative humidity to the air temperature. This could certainly be investigated further, but it does provide some interesting insight that we didn't find in our initial research. As to why this phenomena is occurring, since we found nothing relating air temperature and humidity other than that they are generally inversely proportional, we have no explanation.

Future Steps and Development

In terms of the Time Series ARIMA Model, one improvement that could be made with more time and research is implementing exogenous variables such as precipitation and humidity to improve the forecasting and prediction accuracy. Also, a formal testing validation would be insightful such as mean squared error or residual plots.

The linear regression could be made better by adding normalization, trying out different learning rates, and maybe training for a longer duration. We would also aim to attempt a multivariate linear regression model in the future, as the correlations would potentially improve with better feature engineering.

For the ODE model, regression models beyond polynomial regression weren't explored much during this project. To make the models better, better feature creation and more model exploration could be done. Additionally, if the data was cleaner, stronger correlations would have likely shown, but since so many assumptions and edits needed to be made during data curation, those correlations may not have come through.

The UX design and overall structure could be improved to show the users how accurate the model's prediction is so they can view the simulation with that in mind. For the future, we said we wanted to incorporate storm data, so it would be cool to implement "lightning strikes" if all the weather condition values point towards a storm. Our visual representations for some of the weather conditions could be improved. For instance, precipitation is represented by three pictures (each based on how much rain is to be showed on the UI). There was a react-rain-animation library that actually shows the droplets falling on the screen. The number of droplets would increase based on precipitation's value. However, this library and most libraries similar to it are outdated and do not work with the React version our application is built with. If they get updated, then we can use those libraries to make our visuals look more realistic. One final improvement to the UI would be to include a page for "Differenital Equation Regression Modeling" so the user can learn how their input data is being used to predict the weather condition value of their choice.

Appendix: Division of Labor

Manav finished the data curation and decided to further remove and consolidate potential variables. For example, averaged measurements from the same day so that the only time variable is the day of the year. Additionally, worked on the ODE creation aspect. After trying to build time-dependent ODEs for each variable by using a series of functions as ODE solvers, I realized that it made more sense to create specific functions for each variable, and then make models around those functions. For example, the temperature is heavily dependent on the day of the year, and can easily be modeled using polynomial regression to predict the temperature on a particular day of any year. Therefore, ODEs should be able to be created for each variable that we want to model using this method.

Somil focused on the modeling and regression analysis portion of the project, and also did a bit of the data cleaning. He also conducted a feature correlation analysis based on the results of the linear regression model. His linear regression model predicted temperature based on humidity, and found a negative relationship between the two variables. He also conducted an error analysis using mean square errors and residuals which suggested that the prediction analysis was not completely reliable, and further investigation is needed.

Sarang was able to work on the Time Series Analysis of Air Temperature based on the given data. A traditional machine learning model combined with differential equations was utilized: Auto-Regressive Integrated Moving Average models (ARIMA). This model splits the given temperature data into a training and test set to run the ARIMA mode on, which eventually will be tested by zero-differentiated and first-differentiated models. The most optimal and accurate model will be chosen between the two to eventually forecast the temperature data and predict future data. As part of future steps, Sarang needs to revise the current data to include a Date column that will make plotting more refined and clear. Also, he will be working on finishing up the rest of the model, which includes implementing the first-differentiated model, analyzing and deciding between the two, and finally delving into forecasting and eventual creation for our UI Webpage.

Khushi oversaw the development of the UI. She and Sarang wireframed out the different screens to accomplish the goal of the application: users can interact with our various ML prediction models and then see visual representations of the imputed and predicted values. We are focusing on 5 weather factors (air temperature, relative humidity, precipitation, dewpoint temperature, and atmospheric level or pressure). She created the different visual representations of each feature and scale based on predicted and inputted user values. She also formatted the "Prediction Screen" so the different weather features were placed like they would be in the real world to simulate a random Atlanta day and weather over the skyline. Manav hosted the Differential Equation Regression Models on Heroku using a Flask app and set up GET and POST calls with Khushi to feed user input data into the models and output predicted values for Khushi to show on the UI.

References

Callh, S. (2020). Sebastian Callh personal blog. Forecasting the weather with neural ODEs | Sebastian Callh personal blog. Retrieved March 13, 2022, from https://sebastiancallh.github.io/post/neural-ode-weather-forecast/

GEOSTATIONAROPERATIONAL ENVIRONMENTAL SATELLITES—R SERIES. (2021, July 28). N*OAA Satellite Data Predict Future Lightening Strikes*. NOAA satellite data predict future lightning strikes. Retrieved March 13, 2022, from https://www.goes-r.gov/featureStories/noaaSatelliteDataPredictFutureLightningStrikes.html

Liangtao Xu, Shuang Chen, Wen Yao, "Evaluation of Lightning Prediction by an Electrification and Discharge Model in Long-Term Forecasting Experiments", Advances in Meteorology, vol. 2022, Article ID 4583030, 13 pages, 2022. https://doi.org/10.1155/2022/4583030

https://maths.ucd.ie/~plynch/Publications/pcam0159_proof_2.pdf

Lynch, P. 2006. The Emergence of Numerical Weather Prediction: Richardson's Dream. Cambridge: Cambridge University Press.

NOAA National Severe Storms Laboratory. (n.d.). *Lightning forecasting*. NOAA National Severe Storms Laboratory. Retrieved March 13, 2022, from https://www.nssl.noaa.gov/education/svrwx101/lightning/forecasting/

Wikimedia Foundation. (2022, February 13). *Model output statistics*. Wikipedia. Retrieved March 13, 2022, from https://en.wikipedia.org/wiki/Model_output_statistics

Wikimedia Foundation. (2022, February 23). *Numerical weather prediction*. Wikipedia. Retrieved March 13, 2022, from https://en.wikipedia.org/wiki/Numerical_weather_prediction