Scientific Research Publishing

**Atmospheric and Climate Sciences, 2023, 13, 607-615**
https://www.scirp.org/journal/acs
ISSN Online: 2160-0422
ISSN Print: 2160-0414

# Using Python to Predict Global City Temperatures for 400+ Cities

**Rishi Manjure**

California Institute of Technology, Pasadena, USA
Email: rishi.manjure@gmail.com

**How to cite this paper:** Manjure, R. (2023) Using Python to Predict Global City Temperatures for 400+ Cities. *Atmospheric and Climate Sciences*, **13**, 607-615.
https://doi.org/10.4236/acs.2023.134034

**Received:** July 7, 2023
**Accepted:** October 28, 2023
**Published:** October 31, 2023

## Abstract

The purpose of this investigation was to use Python to model global city temperatures for 400+ cities for many decades. The process used a compilation of secondary data to find my renowned sources and use different regression models to plot temperatures. Climate change is an impending crisis for our Earth, and modeling its changes using Machine Learning will be crucial to understanding the next steps to combat it. With this model, researchers can understand which area is most harshly affected by climate change leading to prioritization and solutions. They can also figure out the next sustainable solutions based on climate needs. By using KNeighbors and other regressors, we can see an increase in temperature worldwide. Although there is some error, which is inevitable, this is mitigated through several measures. This paper provides a simple yet critical understanding of how our global temperatures will increase, based on the last 200+ years.

## Keywords

Machine Learning, Climate Change, Sustainability, Python, Atmospheric Sciences, Modeling

## 1. Introduction

In an era overshadowed by the looming specter of global warming, understanding the intricate dynamics of our planet's changing climate has never been more urgent. The backdrop against which this research unfolds is one of escalating concern and environmental peril, as anthropogenic activities continue to alter the delicate balance of our world's ecosystems. The focus of this research endeavor is to employ Python and Machine Learning Models as a powerful tool for constructing comprehensive temperature models spanning several decades across more than 400 cities globally. This study builds upon the pioneering work of Dr. Jane Smith, whose groundbreaking research in 2017 provided valuable insights into regional temperature variations. Dr. Smith's work, although instrumental, was limited to a smaller dataset and less advanced modeling techniques. Our research aims to expand upon her findings and take a more comprehensive approach to model global temperatures and their evolution over the past 200+ years. This study is of paramount importance due to its potential to shed light on the far-reaching consequences of climate change. As the global temperature steadily rises, comprehending these temperature fluctuations is paramount for charting effective mitigation strategies. By utilizing advanced techniques such as K-Nearest Neighbors (KNeighbors) and other regression methodologies, we can vividly illustrate the escalating global temperature trends. By building on the foundation laid by previous researchers like Dr. Smith, we strive to provide a more comprehensive understanding of how our global temperatures will increase, aiding in the formulation of critical strategies to combat climate change.

## 2. Methodology

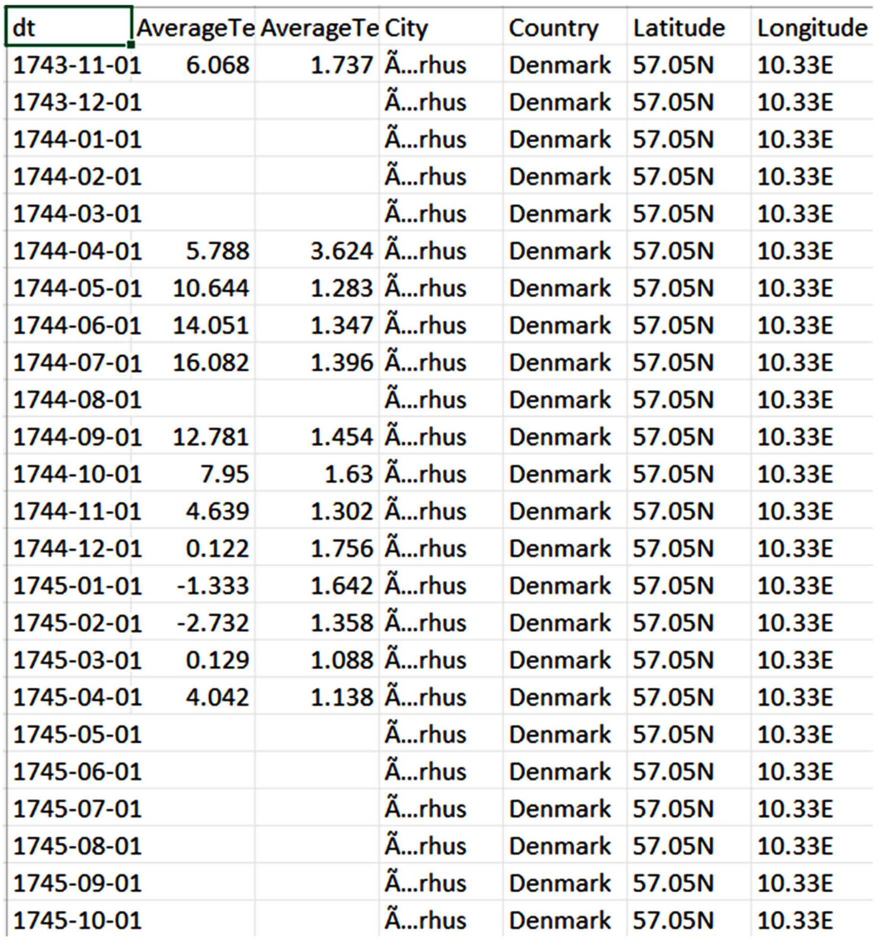
The first step in this process was to be able to visualize the data in a map, preferably, a MatPlotLib colored scatter plot. Initial data was collected from kaggle.com, an online database for various types of data. From there, we downloaded CSV files of Global Land Temperatures by City and Country. These CSV files had columns for dates, temperatures, and locations (**Figure 1**).

These files were stored in a folder named Data. However, to create this visualization, one 8 million row CSV file would take much too long to run. The first function created, named "split_data", and divided temperature data by city into individual years into Json files. This was primarily done to speed up processing time while generating visuals and models. In terms of file directory, artifacts (which held models and data) were the parent file for split_data, which held the rest of the Json files. Each Json file was a default dictionary, consisting of the year, month, temperature, city name, and location. **Figure 2** below shows the file break down of artifacts.

By utilizing split_data, processing data is much more efficient, and easier to access while using models to generate predictions. The JSON files from split_data will be used as inputs for generating models.

Once the function split_data was completed, the map-making process began. The function written, "make map", took the parameter of a year to create a MatPlotLib colored scatter plot of average temperatures during that year in the cities recorded. It would load in the required JSON file of the year from split_data and resize the data onto the scatter plot. **Figure 3** below is a map made of the data points from the year 2000.

By making this function, we can use it to visualize not just past data, but future data as well when generating future models. Those will be presented in various graphics later.



| dt | AverageTe | AverageTe | City | Country | Latitude | Longitude |
|---|---|---|---|---|---|---|
| 1743-11-01 | 6.068 | 1.737 | Ã...rhus | Denmark | 57.05N | 10.33E |
| 1743-12-01 | | | Ã...rhus | Denmark | 57.05N | 10.33E |
| 1744-01-01 | | | Ã...rhus | Denmark | 57.05N | 10.33E |
| 1744-02-01 | | | Ã...rhus | Denmark | 57.05N | 10.33E |
| 1744-03-01 | | | Ã...rhus | Denmark | 57.05N | 10.33E |
| 1744-04-01 | 5.788 | 3.624 | Ã...rhus | Denmark | 57.05N | 10.33E |
| 1744-05-01 | 10.644 | 1.283 | Ã...rhus | Denmark | 57.05N | 10.33E |
| 1744-06-01 | 14.051 | 1.347 | Ã...rhus | Denmark | 57.05N | 10.33E |
| 1744-07-01 | 16.082 | 1.396 | Ã...rhus | Denmark | 57.05N | 10.33E |
| 1744-08-01 | | | Ã...rhus | Denmark | 57.05N | 10.33E |
| 1744-09-01 | 12.781 | 1.454 | Ã...rhus | Denmark | 57.05N | 10.33E |
| 1744-10-01 | 7.95 | 1.63 | Ã...rhus | Denmark | 57.05N | 10.33E |
| 1744-11-01 | 4.639 | 1.302 | Ã...rhus | Denmark | 57.05N | 10.33E |
| 1744-12-01 | 0.122 | 1.756 | Ã...rhus | Denmark | 57.05N | 10.33E |
| 1745-01-01 | -1.333 | 1.642 | Ã...rhus | Denmark | 57.05N | 10.33E |
| 1745-02-01 | -2.732 | 1.358 | Ã...rhus | Denmark | 57.05N | 10.33E |
| 1745-03-01 | 0.129 | 1.088 | Ã...rhus | Denmark | 57.05N | 10.33E |
| 1745-04-01 | 4.042 | 1.138 | Ã...rhus | Denmark | 57.05N | 10.33E |
| 1745-05-01 | | | Ã...rhus | Denmark | 57.05N | 10.33E |
| 1745-06-01 | | | Ã...rhus | Denmark | 57.05N | 10.33E |
| 1745-07-01 | | | Ã...rhus | Denmark | 57.05N | 10.33E |
| 1745-08-01 | | | Ã...rhus | Denmark | 57.05N | 10.33E |
| 1745-09-01 | | | Ã...rhus | Denmark | 57.05N | 10.33E |
| 1745-10-01 | | | Ã...rhus | Denmark | 57.05N | 10.33E |

Figure 1. A screenshot of the raw data of "Global Land Temperatures by City". In column 1 are the dates. Column 2 is the average temperature for that month. Column 3 is the average temperature fluctuation for that month. Column 4 is the city. Column 5 is the country. Columns 6 and 7 are latitude and longitude, respectively [1].

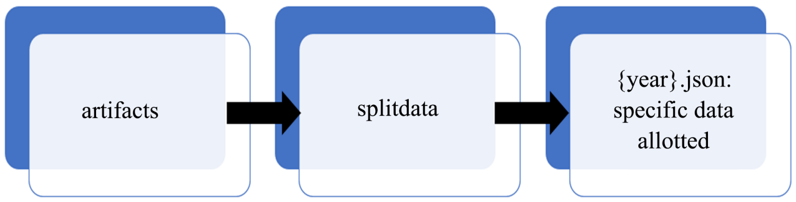
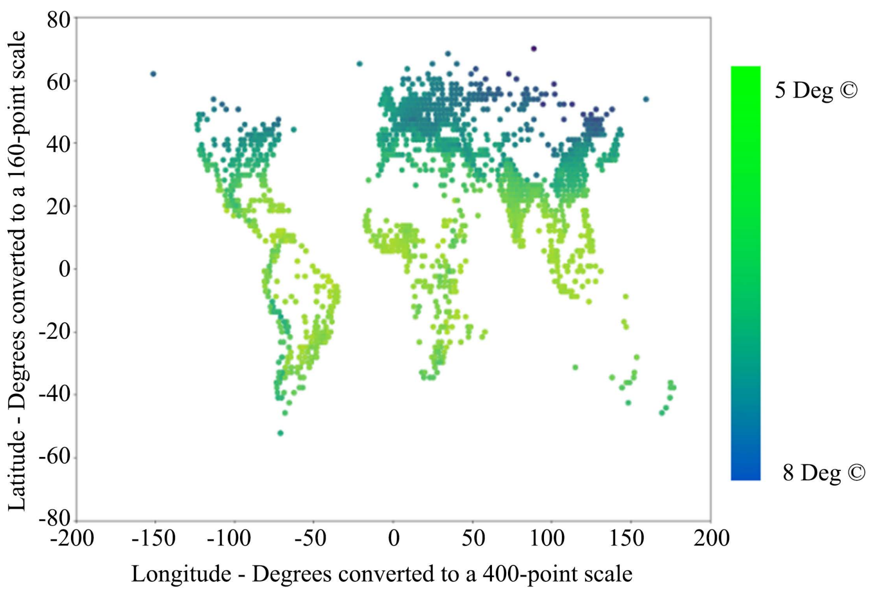


Figure 2. The file system of just the data storage from the online database. Artifacts is a folder which holds tangible, generated information from the code. Split data is a folder which holds each Json file. Each Json file is a default dictionary that holds information from the entire year, the 400 cities, and their temperature and locations.

The development of the subsequent set of functions centered around the creation of diverse models designed to forecast global average annual temperatures spanning the upcoming decades. At the heart of this functionality was the "make_temp_models" function, which entailed a comprehensive parameterization scheme. Users were afforded the flexibility to specify the type of regressor, the target year for prediction, and the specific city of interest for temperature



Latitude - Degrees converted to a 160-point scale
Longitude - Degrees converted to a 400-point scale
5 Deg ©
8 Deg ©
80
60
40
20
0
-20
-40
-60
-80
-200
-150
-100
-50
0
50
100
150
200

**Figure 3.** Picture of the finished product of the make_map function of the year 2000. The color scale on the right indicates annual average temperatures of certain cities. This is in the format of a PNG file.

forecasting. In cases where no particular city was designated, the model was designed to generate temperature predictions for all 400 cities within the dataset. The core input data for this function consisted of two essential components: firstly, the JSON file corresponding to the year immediately preceding the target prediction year, and secondly, a historical dataset encompassing 119 years of temperature data in JSON format. This historical dataset was instrumental in training the predictive models, enabling them to learn from past temperature trends and patterns. In essence, it served as the foundation upon which accurate temperature forecasts could be constructed. The output of the "make_temp_models" function was a pickle file specifically tailored to the decade being predicted. To illustrate this with an example, consider the scenario where one sought to ascertain the projected temperature outcomes for the decade spanning from 2010 to 2019. In this instance, the "make_temp_models" function would leverage the historical temperature data from JSON files covering the period from 1890 to 2009. This extensive dataset would serve as the basis for training the predictive models, enabling them to comprehend the intricate temperature variations and patterns that had unfolded over the preceding 119 years. Subsequently, armed with the knowledge and insights gleaned from this historical data, the function would proceed to generate a pickle file specifically tailored to the 2010-2019 decade. This pickle file encapsulated the predictive models' collective wisdom and their capacity to generate temperature forecasts with a focus on the specified timeframe. In essence, it provided a condensed yet comprehensive representation of the temperature predictions for the chosen decade. The "make_temp_models" function's versatility extended beyond the example scenario outlined above. Users had the autonomy to select their desired regressor type, target prediction year, and city of interest, tailoring the function's output to suit their specific analytical requirements. Whether one aimed to forecast temperatures for a single city or multiple cities, this function served as a valuable tool for harnessing the power of predictive modeling in the realm of global average annual temperature projections. In summary, the "make_temp_models" function played a pivotal role in the creation of predictive models aimed at forecasting global average annual temperatures. It offered a robust parameterization scheme, enabling users to define crucial aspects of the modeling process. By leveraging historical temperature data spanning over a century, this function facilitated the training of accurate and data-driven predictive models. Its output, in the form of decade-specific pickle files, encapsulated the collective knowledge of these models, enabling users to access temperature forecasts tailored to their specific areas of interest and prediction timelines.

## 3. Results

Now that the basics of the "make_temp_models" have been explained, it is important to highlight the types of regressors and models and their impacts on predictions given.

The first type of regressor is the KNeighborsRegressor, which uses the average data from the closest data points (or neighbors). Figure 4 below depicts how a very simple KNeighborsRegressor would work [2].

In the context of our experimental work, the KNeighborsRegressor emerged as one of the standout models, primarily due to its remarkable performance characterized by exceptionally low prediction errors. Intuitively, one might assume that incorporating a greater number of data points or neighbors into the

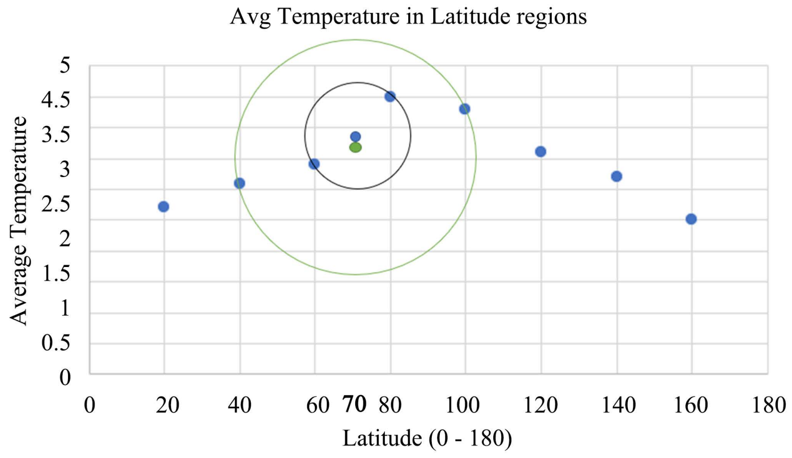


Figure 4. This figure is how a model KNeighborsRegressor would work. The goal of this simple model is to predict the average temperature at 70 degrees Latitude, with other data points given. If the KNeightborsRegressor takes the parameter of 2 neighbors, it takes the average of the 2 closest data points. As shown in the figure, the black circle indicates the closest 2 data points (60 and 80 latitude) which will be used to predict 70 degrees latitude. The black point indicates the average of the two points. Additionally, the green point is the prediction with 4 closest neighbors, as indicated by the green circle on the graph [2].

modeling process would inherently lead to more accurate predictions. However, this assumption does not hold universally, as our findings have revealed a nuanced relationship between the number of neighbors and prediction accuracy. In coastal regions, a pattern emerges, where employing fewer data points for predictions can yield more advantageous results because coastal regions act differently that inland cities. By restricting predictions to rely solely on neighboring coastal cities, the model can better capture the specific climatic conditions and idiosyncrasies prevalent in these regions. This localized approach often outperforms the broader inclusion of distant neighbors. Nevertheless, in a more general context, increasing the number of neighbors utilized in the modeling process tends to lead to superior prediction outcomes. The fundamental principle behind this trend is the wisdom of crowds, where aggregating information from a larger and more diverse set of neighbors allows the model to gain a more comprehensive understanding of the underlying patterns and trends in temperature data. To illustrate this relationship, we present a graph depicting the mean errors in annual temperature predictions for the city of London, specifically for the years following 2013. The graph serves as a visual representation of the impact of varying the number of nearest neighbors on prediction accuracy. In particular, we examine three scenarios: one utilizing 3 nearest neighbors, another employing 7 nearest neighbors, and a third incorporating 12 nearest neighbors [3]. This graphical representation enables us to observe how the mean prediction errors evolve as we modify the number of neighbors considered in the modeling process. It provides valuable insights into the trade-offs and nuances associated with selecting the optimal number of neighbors, thereby informing our approach to fine-tuning predictive models for temperature forecasting in specific regions (**Figure 5**).

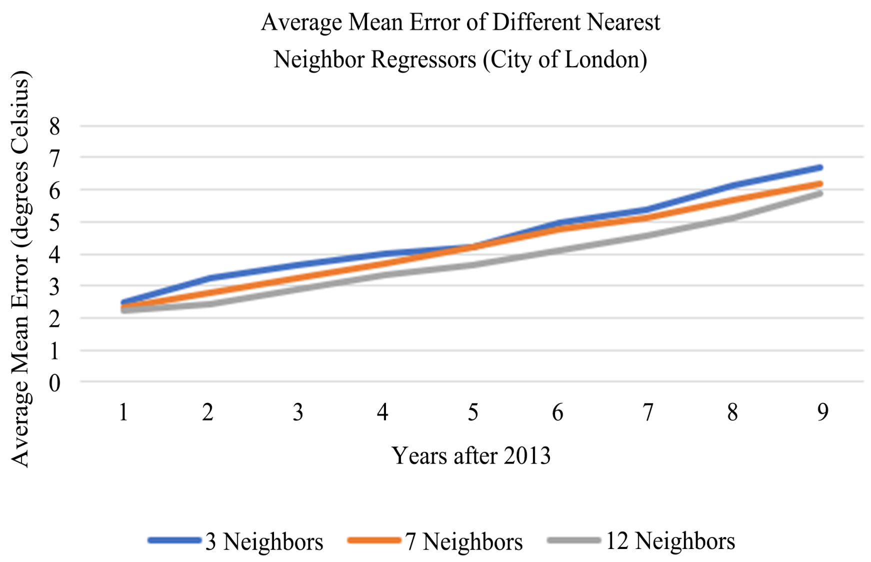


**Figure 5.** This figure is a graph of the mean errors (in degrees Celsius) of nearest neighbor models in the City of London. The blue line, which uses the models with 3 neighbors, consistently has the highest mean error. Meanwhile, the errors for 7 neighbors and 12 neighbors are slightly more accurate.

While the KNeighborsRegressor showcased admirable accuracy by employing 12 neighbors, it still exhibited a significant limitation: an average error of nearly 6 degrees persisted over the course of a decade's worth of predictions. This level of inaccuracy was deemed unsatisfactory, prompting the exploration of an alternative model—the Multi-Layer Perceptron (MLP), which subsequently yielded vastly superior results when compared to the KNeighborsRegressor. Diverging from the simplicity of the KNeighborsRegressor, the MLP stands out as a significantly more intricate and versatile model. It boasts an array of parameter options and leverages a diverse and extensive dataset to formulate its predictions. In our case, this dataset spans an impressive 120 years, encompassing a staggering 48,000 data points. This extensive historical record provides the MLP with a robust foundation for learning and comprehending the complex patterns inherent in temperature data. The essence of the MLP's predictive prowess lies in its ability to learn non-linear functions from the provided data observations and corresponding targets. Unlike many traditional regressors, the MLP introduces the concept of hidden layers into the modeling process. These hidden layers, positioned between the input and output layers, play a pivotal role in shaping the model's transformation of input data into output predictions. To grasp the significance of these hidden layers, consider **Figure 6** as a visual representation of how a 1-hidden layer MLP model operates. The illustration captures the essence of how this intermediate layer, or layers, strategically "weights" the input data, orchestrating a transformation that ultimately yields the desired output data. This unique architecture endows the MLP with the capacity to capture intricate relationships and dependencies within the data that might elude simpler models like the KNeighborsRegressor.

The MLP regressor can take up to 100 by 100 hidden layer sizes, though in terms of this experiment, a 32 by 32 model had the best results. In fact, the model had errors as low as 1.82 degrees Celsius. **Figure 7** below depicts a 3 × 3 plot of the decade of 2013. In each plot are the mean average errors and a map of the annual temperatures [4].

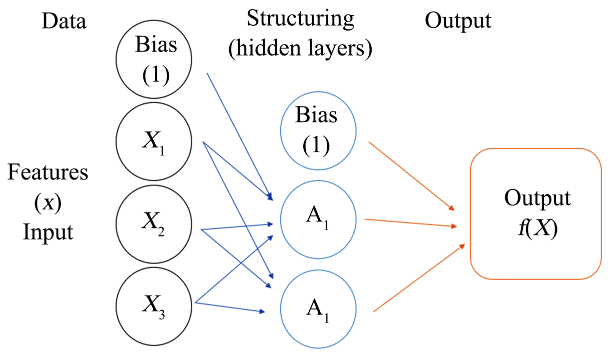


**Figure 6.** A simple MLP regression model with one hidden layer and no other parameters given. The first row of "circles" in the diagram represents the raw data which is passed into the regressor. These can be labeled as the observations or "x" datapoints. Then, the datapoints are transformed by the hidden layer. This layer weights the data into a non-linear function, which then can be used to determine an output: f(X) [4].

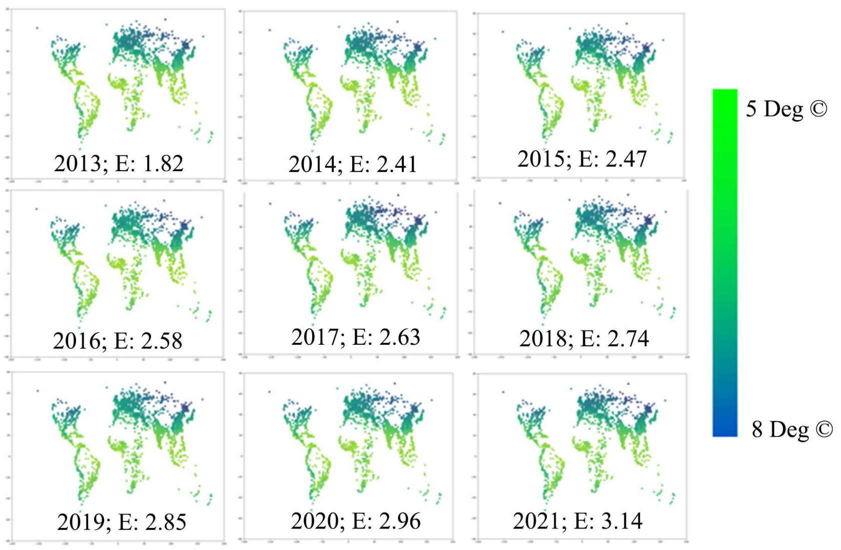


**Figure 7.** A 3 × 3 plot of the decade of 2013; these are predicted temperature maps by the MLP Regressor The "E:" represents the mean error for all the data points plotted. As the years progress, the error rises nearly linearly from 1.82 to 3.14.

Although the error does increase steadily over the decade, the MLP Regressor with 32 by 32 hidden layers proved to be the most accurate model.

## 4. Conclusion and Implications

The scope of this investigation encompassed the modeling of annual climate temperatures in a vast array of over 400 cities spanning the globe. This ambitious endeavor was achieved through the strategic utilization of diverse modeling techniques and an extensive dataset. The implications of these outcomes extend far beyond the realm of temperature forecasting, offering a multitude of tangible benefits and applications. One of the foremost advantages lies in the capacity to employ these temperature predictions for the proactive management of critical environmental concerns. For instance, the ability to accurately anticipate temperature fluctuations can serve as a vital tool in predicting and mitigating the occurrence of wildfires. Temperature is a key factor influencing the ignition and spread of wildfires and precise temperature forecasts can facilitate timely response and resource allocation to combat these destructive events. Moreover, the impact of temperature changes on ecological systems is profound, making these predictions invaluable in the modeling of animal species populations. As temperatures fluctuate, it directly influences the habitats and behavior of various species. By leveraging temperature change data, scientists and conservationists can enhance their understanding of how these shifts impact wildlife, aiding in the development of conservation strategies and the preservation of vulnerable species. While the achievements of this investigation are commendable, there remains room for improvement in the pursuit of even more accurate climate models. The refinement of models and the acquisition of larger initial datasets represent promising avenues for enhancement. With additional resources and data at our disposal, the potential to create increasingly precise climate change models becomes tangible. This pursuit is vital in our ongoing efforts to understand and address the complex challenges posed by climate change. One of the notable outcomes of this investigation is the ability to predict and visualize annual temperature maps based on historical data spanning previous centuries. These maps offer valuable insights into the long-term temperature trends and variations that have shaped our planet's climate. They serve as critical tools for researchers, policymakers, and the public alike, fostering a deeper understanding of the dynamic nature of our global climate system.

## Conflicts of Interest

The author declares no conflicts of interest regarding the publication of this paper.

## References

[1] Climate Change: Earth Surface Temperature Data.
https://www.kaggle.com/datasets/berkeleyearth/climate-change-earth-surface-temperature-data

[2] Scikit "sklearn.neighbors.KNeighborsRegressor".
https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsRegressor.html

[3] Scikit "sklearn.neural_network.MLPRegressor".
https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPRegressor.html

[4] Choudhury, K. (2020) Deep Neural Multilayer Perceptron (MLP) with Scikit-Learn.
https://towardsdatascience.com/deep-neural-multilayer-perceptron-mlp-with-scikit-learn-2698e77155e