A comparative study of supervised learning methods for long term temperature prediction

# INTRODUCTION

Roberta Boscolo, Climate and energy lead, World Meteorological Organization – why is she relevant to the thesis

She develops [weather, water, and climate services](https://community.wmo.int/en/activity-areas/energy) to support energy resilience and renewable energy.  Since 2022, she is a member of the Expert Advisory Panel of the [Earthshot Prize](https://earthshotprize.org/people-partners/expert-advisory-panel/) and Top Voice of LinkedIn for Green Economy.

Dr Boscolo often highlights the influence of climate change on weather on her personal LinkedIn profile, which now points the attention towards events that cause high mortality rates and high levels of diseases stemmed from climate change, greenhouse gas emissions and unnatural weather events.

For example, Dr Boscolo highlights that “over 70% of the global workforce faced excessive heat in 2020, a 9% increase since 2000. Africa is the hardest hit, with nearly 93% of workers exposed to extreme heat, followed by the Arab states (83.6%) and Asia and the Pacific (74.7%).”, information sourced from the United Nations International Labour Organisation. According to both Dr Boscolo, ILO and the WMO, “human-induced climate change is driving global temperatures higher. Regions not traditionally impacted by extreme heat are now experiencing significant changes: Europe and Central Asia: Heat exposure at work increased by 17.3% over 20 years, affecting 29% of workers. More specifically, according to this Financial Times [article](https://www.ft.com/content/f5deaa2d-3ad1-4087-8b1b-0a16445bbca4), Europe has observed record number of days with extreme heat stress in 2023:

A graph of the global economic growth

Description automatically generated with medium confidence

It seems that the peak of heatwave was in July 2023, where the temperatures reached over 40 degrees Celsius. Previously, in July 2022, Europe has seen a heat stress of a little bit over 30 degrees Celsius. The Financial Times articleshowcased that “about 70 per cent of the global workforce were now at high risk of extreme heat”. Furthermore, Mooney 2024 continues to explain that “the ILO cited a study finding that if the core body temperature rises above 38C, physical and cognitive functions are impaired, and above 40.6C, the risk of organ damage, loss of consciousness and fatality, increases sharply”.

<https://wmo.int/media/news/un-secretary-general-issues-call-action-extreme-heat-0>

Based on the above, it seems that heat extremes are no longer a “once a year” kind of event. They have increased in frequency and intensity, and it is affecting all aspects of the human life as we know it. Which shows just how important weather is to our normal functioning.

There are multiple studies published that highlight the effects of seasonal weather and climate change extreme events and their effects on mental health and thus on our life as we know it and perceive it.

For example, the UK Health Security Agency has published the “Adverse Weather and Health plan” in 2024, highlighting how weather affects the global population, not just the economy.

Both heat and cold events lead to significant numbers of deaths each year and they are expected to increase in future years, according to the UKHSA. According to the HECC report in 2023 cited in the UKHSA report, heat related deaths are expected to increase to over 21000 in the 2070s if not action is taken against them. In 2022, 2985 heat related deaths were observed alone in the summer of 2022 ().

#### 1.3.1 COLD WEATHER

Cold weather has raised its risk in recent years. According to the UKHSA report, “the output of the FluMoMo model, estimated that 5,533 deaths in winter 2022 to 2023 could be attributed to periods of extreme cold (14)”. Cold weather is known to affect the human body through respiratory viruses whilst increasing pressure on health and social care services. In the UK specifically, mortality is higher between December and March, as these months face an increase in respiratory illnesses.

Besides affecting the physical health of humanity, cold weather is also known to cause severe mental health conditions. The UKHSA reports that during the winter months, damp and cold housing have an incredible impact over individuals’ mental health, as “living in these homes can affect people’s ability to go about their daily lives. Some people become socially isolated as they are reluctant to invite friends or family to a cold house, while others seek refuge outside the home” […]According to UKHSA, it is known that there is a direct correlation between mental disorders such as anxiety and depression and cold, damp houses. In the questionnaire “Fuel poverty occurs when a household cannot afford to adequately heat their home or meet basic energy requirements”.

#### 1.3.2 HEAT

According to UKHSA, the second weather extreme is also posing severe impacts on humanity. Extreme heat is increasing at “an unprecedented rate”: “there has been a 6-fold increase of concurrent heatwaves since the 1980s which are compounding the impacts of other natural hazards, such as drought, wildfires and flash flooding”.

In the UK the warmest years on record have occurred since 2002.

The reason why heat events tend to cause such high mortality, is due to thermoregulation. UKHSA reports that “thermoregulation controlled by the hypothalamus and can be impaired in the older adults, those with certain long-term health conditions and potentially in those taking certain medications making them more vulnerable to overheating. Young children produce more metabolic heat, have a decreased ability to sweat and have core temperatures that rise faster during dehydration. Babies and children sweat less than adults, and this reduces their ability to cool down during hot weather, particularly if they are exercising or being active. This puts babies and children at a higher risk of overheating and developing a heat-related illness as well as making any existing illnesses worse. Babies and children need to be carefully watched during hot weather for signs of heat stress”.

As with cold weather, heat events will impact respiratory but also cardiovascular functions of the body, amongst other events such as heat cramps, heat rash, heat exhaustion, heat stroke, etc. Additionally, the air pollution increases the level of cardiovascular related deaths, “as fine particles have been shown to enter the blood stream via the lungs and affect the heart”.

Besides the cardiovascular and respiratory issues that extreme heat events cause, they can also have a big negative impact on the skin and eye health. The UKHSA report highlights that the ultraviolet (UV) radiation, although useful in the production of vitamin D (caused by short term sun exposure) is useful for humans, too much exposure can lead to many health issues, such as skin cancer and premature skin ageing. According to the World Health Organisation (WHO), “the number of cases of malignant melanoma has doubled every 7 to 8 years over the last 40 years, mostly due to a marked increase in the incidence of skin cancers in fair-skinned populations that have been reported since the early 1970sthe cases of malignant melanoma”.

As with cold weather, heat events are also expected to cause serious damage to mental health and wellbeing. The UKHSA reports that the higher the temperatures, the higher the risk of adverse mental health outcomes. The strongest evidence seems to suggest that high temperatures actually lead to an increase in suicide rates and hospital admissions for mental illnesses. The table below has been put together by the UKHSA in order to explain the different types of mental illnesses stemmed from the correlation between temperature and mental health.

Due to greenhouse gas emissions and climate change, the temperature of a place is one of the most significant factor in the health of individuals, therefore it needs close monitoring. The next part of this project is concerned with how is weather forecasted through machine learning.

# LITERATURE REVIEW

<https://journals.ametsoc.org/view/journals/bams/101/7/bamsD190081.xml?tab_body=abstract-display#d4108973e90>

Weather is not a new concept and neither is weather forecasting. However, with an ever increasing threat stemmed from the increase of greenhouse gas emissions (GHG), it is necessary now more than ever to understand how to use all of our tools in order to monitor the events and prepare accordingly.

Research shows that when it comes to weather, part of society tends to be part of the Dunning-Kruger effect (DKE), which “is a cognitive bias where people are unable to recognize their own incompetence (Kruger and Dunning 1999)”. The authors highlight that “the overestimation of weather knowledge, could have big implications during a natural disaster if their overestimation of weather knowledge creates a false sense of safety. Siems (2016) showed DKE tendencies were evident when analyzing evacuation response patterns during Hurricane Katrina with about three-quarters of nonevacuees showing signs of overestimating their knowledge. An important consideration for a future study is whether one who overestimates his or her weather knowledge may also be overconfident in his or her ability to make important decisions during an extreme weather event”. In simpler terms, in the meteorological context of ever changing extreme weather conditions, research shows that it is necessary to create a much better understanding of weather and temperature, in order to save human life and property.

In order to do that, this project aims to look at the temperature aspect of the weather, which is simply a combination of multiple other factors, such as humidity, UX index, solar radiation, wind gust, wind speed etc. Temperature is the most looked at aspect in weather prediction, because it represents the mix of all the other weather elements that can threaten the population.

* Methods of weather forecasting

Weather prediction has been around since before technology took over the world. It was mostly done through the observation of factors that contribute to temperature. For example, according to “The weather book” (Diana Craig, 2011), historical weather prediction methods included Aristotle’s “Meteorologica” BC, “where he showcased theories based on multiple natural phenomena, from rainbows to snow. “Humans would also rely on a number of methods to observe and predict weather – Dana Craig lists a few examples. Nicholas of Cusa has looked at the moisture absorbed by wool in order to measure humidity (c. 1401 – 64). Leon Battista Alberti has invented the anemometer for measuring windspeed (1404) and much more.

1961 saw the first attempt of a numerical weather prediction model, however only in 1940 I.A. Kibel “developed the first prognostic model by using expansions of the equations of motion for a baroclinic atmosphere in terms of a small parameter”. (<https://www.researchgate.net/profile/Mikhail-Tolstykh-3/publication/282027587_Some_Current_Problems_in_Numerical_Weather_Prediction/links/5601a72d08aecb0ce881acf6/Some-Current-Problems-in-Numerical-Weather-Prediction.pdf>)

At present, almost all operational weather forecasting are based on Numerical Weather Prediction (NWP) [5], which is essentially a set of nonlinear equations, known as the primitive equations [6]. (<https://www.sciencedirect.com/science/article/abs/pii/S2214579620300460>)

According to <https://www.sciencedirect.com/science/article/pii/S2214579620300460?via=ihub>, “weather prediction has been regarded as a physical theory problem, and meteorological scientists have been committed to improving the accuracy of forecasts through the understanding of physical mechanisms, which is a theory-driven approach”.

* How does NWP work

<https://www.sciencedirect.com/science/article/pii/S2214579620300460?via=ihub>

*i*) obtaining the original observation datasets, including [remote sensing](https://www.sciencedirect.com/topics/earth-and-planetary-sciences/remote-sensing) data, in situ observation data, and the simulation results of the NWP models at last time point; *ii*) preprocessing the raw datasets through the [data assimilation](https://www.sciencedirect.com/topics/earth-and-planetary-sciences/data-assimilation) analysis system, including data quality control and quality assurance; *iii*) inputting the preprocessed datasets into the [atmospheric dynamic](https://www.sciencedirect.com/topics/earth-and-planetary-sciences/atmospheric-dynamics) model equations for prediction; and *iv*) post-processing and visualizing the results of prediction.

* Are NWP the best method for weather forecasting?

According to <https://www.sciencedirect.com/science/article/abs/pii/S2214579620300460>, “there are still several challenges in NWP. Firstly, due to the chaotic nature of atmosphere [8], the small differences in initial conditions have a huge impact on model results”. This means that any new information introduced in the model regarding atmospheric physical process will introduce errors and therefore rendering the model unusable This requires more effort and time and it shows a decreased accuracy in the model. The authors continue to highlight that NWP models generate TBs of simulation results each day which are computationally expensive.

With the advancement of technology, machine learning have been invested in as new ways of predicting the weather. Research shows that AI can be both faster and cheaper to execute than regular NWP, and it can also adapt to data much faster when it comes to the chaotic nature of the atmosphere, therefore increasing accuracy.

According to the authors, “The artificial neural network (ANN) is a powerful data modeling tool that can capture and represent complex relationships between inputs and outputs, which is developed by the motivation of implementing artificial systems that can perform intelligent tasks similar to those performed by the human brain […] The deep neural network (DNN) is a kind of ANN composed of multilayers architecture, which can reconstruct the raw data sets from the original feature space into a learned feature space. In other words, they can “learn” features by neural networks (NNs) instead of selecting features manually [11], and achieve higher accuracy and better generalization with the learned features. Deep learning (DL) has achieved encouraging results in many areas, such as computer vision [12], speech recognition [13], [14], [15], natural linguistic programming [16], etc”.

### Machine learning in weather prediction

<https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3350281>

***Weather prediction has seen a variety of approaches in recent years based on Genetic Algorithms and Neural networks but these fail to capture the complex relationships between various factors which affect weather.***

In the neural network approach, Kaur[2] and Maqsood[1] describes a model that predicts the hourly temperature, wind speed and relative humidity 24 hours ahead. The authors have made a comparison of Multilayer Perceptron Networks (MLP), Radial Basis Function Network (RBFN). All these models do a good job in identifying the seasonal variations but fail in trend and random variations. A variety of time series models have also been developed[6] like Autoregression, Autoregressive Moving Average (ARMA), Autoregressive Integrated Moving Average (ARIMA) etc. but combining time series with the neural network[7] is mostly unexplored. We found that RNN with time series performs better than the classical time series models.

According to Singh et al (2019 - <https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3350281> ), time series RNN is better than SVM and ANN. Additionally, the authors highlight that the prediction window is crucial because the larger the prediction window, the higher the error.

A table with black text

Description automatically generated

<https://ieeexplore.ieee.org/abstract/document/8588666>

According to the authors, the more recent improvements have come from applying artificial intelligence (AI) techniques to improve forecasting and to enable large quantities of machine-based forecasts. One of the early successes of the use of AI in weather forecasting was the Dynamical Integrated foreCast (DICast®) System. DICast builds on several concepts that mimic the human forecasting decision process [1}. It leverages the NWP model output as well as historical observations at the site for the forecast.

<https://cs229.stanford.edu/proj2016/report/HolmstromLiuVo-MachineLearningAppliedToWeatherForecasting-report.pdf>

The issue with the system of ordinary differential equations that govern this physical model is unstable under perturbations, and uncertainties in the initial measurements of the atmospheric conditions and an incomplete understanding of complex atmospheric processes restrict the extent of accurate weather forecasting to a 10 day period, beyond which weather forecasts are significantly unreliable. This introduces the need for a reliable way of producing true forecasts, with little data and robust enough to take on multiple perturbations. Machine learning can provide all of that including not need a understanding of physical processes that govern the atmosphere.

Neural networks seem to be the popular machine learning model choice for weather forecasting because of the ability to capture the non-linear dependencies of past weather trends and future weather conditions, unlike the linear regression and functional regression models that we used. This provides the advantage of not assuming simple linear dependencies of all features over our models. Of the two neural network approaches, one [3] used a hybrid model that used neural networks to model the physics behind weather forecasting while the other [4] applied learning more directly to predicting weather conditions. Other approaches for weather forecasting included using Bayesian networks. One interesting model [2] used Bayesian networks to model and make weather predictions but used a machine learning algorithm to find the most optimal Bayesian networks and parameters which was quite computationally expensive because of the large amount of different dependencies but performed very well. Both linear regression and functional regression were outperformed by professional weather forecasting services, although the discrepancy in their performance decreased significantly for later days, indicating that over longer periods of time, our models may outperform professional ones.

<https://www.mdpi.com/2073-4433/13/2/180>

In this paper, we performed an analysis of the 500 most relevant scientific articles published since 2018, concerning machine learning methods in the field of climate and numerical weather prediction using the Google Scholar search engine. The most common topics of interest in the abstracts were identified, and some of them examined in detail: in numerical weather prediction research—photovoltaic and wind energy, atmospheric physics and processes; in climate research— parametrizations, extreme events, and climate change. With the created database, it was also possible to extract the most commonly examined meteorological fields (wind, precipitation, temperature, pressure, and radiation), methods (Deep Learning, Random Forest, Artificial Neural Networks, Support Vector Machine, and XGBoost), and countries (China, USA, Australia, India, and Germany) in these topics. Performing critical reviews of the literature, authors are trying to predict the future research direction of these fields, with the main conclusion being that machine learning methods will be a key feature in future weather forecasting. For atmospheric scientists, the most interesting group of techniques was found to be ***supervised learning.*** In the case that some labelled data are available, one can use it as a training dataset from which to build a function that maps given inputs to outputs. ***That function can be used in a different dataset, named testing one, to evaluate the model, and if the results are satisfactory, it can be used in the classification or regression of any kind of application needed. In that group we find methods, such as Decision Trees, e.g., Random Forest (RF) [5] or XGBoost (XGB) [6], Artificial Neural Networks (ANN) [7], Deep Learning (DL) [8], and Support Vector Machine (SVM) [9].*** The second group in machine learning is unsupervised learning (Figure 1), in which algorithms do not have labelled data to train from, and must decide upon other ways to divide a given dataset, or reduce the dimensions of it, for further analysis. In this group, the popular methods among atmospheric scientists are K-means Clustering (K-means) [10] and Principal Component Analysis (PCA) [11]. The main goal of this study is to present a review of the machine learning methods and applications within the main topics of meteorology, as well as in climate analyses. We show examples of the use of machine learning techniques as a new method that helps to solve important and complex issues in weather forecasting and in the study of climate change over different temporal- and spatial scales. wable energy is very common among scientists, the phrase “Wind Forecasting” had the highest count (Figure 2). The phrase with the second highest count turned out to be “Ensemble Forecasting”, due to the growing interest in improving probabilistic forecasts and in the methods required to interpret them correctly. ***Slightly fewer counts were recorded for phrases such as “Data Assimilation”, “Extreme Events”, “Remote Sensing”, and “Land Cover”. Less than 10 counts were found for the phrases “Tropical Cyclones”, “Coupled Models”, “Cloud Physics”, and “Boundary Layer”.***

Some of the most interesting results derived using this method are presented in the following section. Figure 4 presents the most commonly used meteorological fields in NWP studies. Scientists mentioned the term ‘wind’ more than 200 times, and this term is related to an important group of renewable energy and wind forecasting studies, as presented in Figure 2. The term ‘precipitation’ was used almost 150 times, usually with regards to applications for short-range prediction, and downscaling or post-processing. Several papers on bias correction of temperature and air pressure were present, as well as studies on radiation, both using photovoltaic application and emulating this scheme in NWP:

***A graph with blue bars

Description automatically generated***

<https://iopscience.iop.org/article/10.1088/1755-1315/427/1/012013/pdf>

In this article, Ma et al (2020), use artificial intelligence and machine learning to analyze the outdoor temperature and humidity that is used in HVAC control, for potential identification of the urban heat island effect.

### Weather prediction and machine learning algorithms

Perhaps talk a little bit about the increase in natural disasters due to unstable weather from climate change

Give examples of how useful weather prediction and DS has been in natural disasters for both economical and humanitarian purposes

* Mexican Seismic Alert System (SAMEX)
* <https://www.researchgate.net/publication/322999028_A_Dedicated_Seismic_Early_Warning_Network_The_Mexican_Seismic_Alert_System_SASMEX>
* <https://hess.copernicus.org/articles/27/1865/2023/>
* Mention LSTM in the article so stress how important and useful it can be

### WEATHER PREDICTION IN LONDON, UNITED KINGDOM

Weather forecasting is based on data such as humidity, temperature, UV index, wind gust, etc. However each of these features depends massively on the part of the world where the data is sourced from. Depending on it, each model performs differently and can produce different results. In a world of ever increasing natural disasters and temperatures, the world needs more research into models that can be used with multiple types of data from multiple parts of the globe without requiring extra data engineering. In certain situations, the extra time saved from easy-to-apply algorithms can save life and property.

The next part of this project is concerned with the methodology applied to multiple machine learning and deep learning algorithms that facilitated this comparative study of weather prediction.

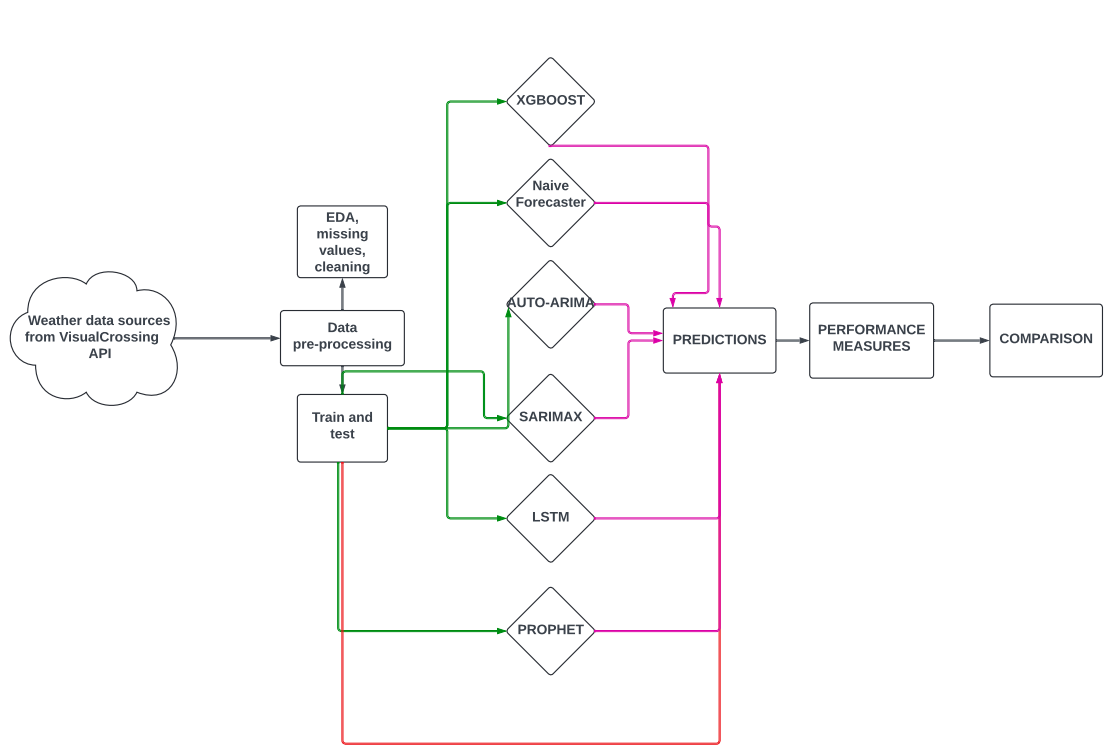
FOR SARIMA <https://www.sciencedirect.com/science/article/pii/S1877050924007403/pdf?md5=d9b6bc98c11f52d765e5ed6d9807b1a6&pid=1-s2.0-S1877050924007403-main.pdf>

FOR AUTO ARIMA

<https://www.sciencedirect.com/science/article/pii/S1877050921000053>

# METHODOLOGY

This next part will follow the metholodgy applied to the weather data sources from the VisualCrossing API. Below there is a representation of the machine learning model applied.



## Dataset and pre-processing

According to their website, <https://www.visualcrossing.com/weather-data>, “the Visual Crossing global weather database provides easy access to decades of national and global weather data […] This allows our weather engine to quickly and easily provide the weather data that you need for any project worldwide. Our engine processes billions of hourly and [sub-hourly](https://www.visualcrossing.com/resources/documentation/weather-api/requesting-sub-hourly-weather-data-using-the-weather-api/) weather observations from more than one hundred thousand worldwide observation stations, including satellite and maritime sources to create our global weather observation database”. Visual Crossing also provide model-based 15 day forecasts, long range weather patterns based on statistical climate modelling and reports concerning weather, however this thesis has not made use of any of these.

An API is an Application Programming Interface, which allows two components to communicate and exchange data. Every API contains one client and one server. According to the AWS website, <https://aws.amazon.com/what-is/api/>, “the application sending the request is called the client, and the application sending the response is called the server.

Visual Crossing is a paid API that provides access to the historical data through their web-based data explorer page.

According to their website, Visual Crossing is sourcing data from “observation stations across the planet, which is then fed back to services like NOAA and DWD”.

There are plenty of sources to obtain historical weather data from, however it has been difficult to find reliable, independent sources with complete and unskewed data. Visual Crossing was chosen due to complete data and easy to use API.

<https://www.visualcrossing.com/resources/documentation/weather-api/timeline-weather-api/>

Requests for the historical weather data have been done through the API requests provided by Visual Crossing, that look like this:

*https://weather.visualcrossing.com/VisualCrossingWebServices/rest/services/timeline/[location]/[date1]/[date2]?key=YOUR\_API\_KEY*

Each link requires the user to include a location, either one or more dates and their unique API key generated after the purchase of the membership. The response is a JSON (JavaScript Object Notation) format, which is for a web API. The response include a set of properties depending on the user’s requirements. In this case, the response was the location of London, United Kingdom, with dates from 1st January 2014 to 1st January 2024, which represent a total of 10 years of daily historical weather data.

The API response contains multiple features commonly used in weather forecasting:

|  |  |
| --- | --- |
| ***Feature*** | ***Description*** |
| **Description** | longer text descriptions suitable for displaying in weather displays |
| **Cloudcover** | how much of the sky is covered in cloud ranging from 0-100% |
| **Conditions** | textual representation of the weather conditions |
| **Datetime** | ISO 8601 formatted date, time or datetime value indicating the date and time of the weather data in the local time zone of the requested location |
| **Dew** | dew point temperature |
| **Feelslike** | what the temperature feels like accounting for heat index or wind chill |
| **Feelslikemax** | maximum feels like temperature at the location. |
| **Feelslikemin** | minimum feels like temperature at the location |
| **Humidity** | relative humidity in % |
| **Icon** | a fixed, machine readable summary that can be used to display an icon |
| **Moonphase** | represents the fractional portion through the current moon lunation cycle ranging from 0 (the new moon) to 0.5 (the full moon) and back to 1 (the next new moon) |
| **Precip** | the amount of liquid precipitation that fell or is predicted to fall in the period |
| **Precipcover** | the proportion of hours where there was non-zero precipitation |
| **Precipprob** | the likelihood of measurable precipitation ranging from 0% to 100% |
| **Preciptype** | an array indicating the type(s) of precipitation expected or that occurred. |
| **Pressure** | the sea level atmospheric or barometric pressure in millibars (or hectopascals) |
| **Snow** | the amount of snow that fell or is predicted to fall |
| **Snowdepth** | the depth of snow on the ground |
| **Source** | the type of weather data used for this weather object |
| **Stations** | the weather stations used when collecting an historical observation record |
| **Sunrise** | The formatted time of the sunrise (For example “2022-05-23T05:50:40”). |
| **Sunset** | The formatted time of the sunset (For example “2022-05-23T20:22:29”). |
| **Temp** | temperature at the location. Daily values are average values (mean) for the day. |
| **Tempmax** | maximum temperature at the location. |
| **Tempmin** | minimum temperature at the location. |
| **Uvindex** | a value between 0 and 10 indicating the level of ultra violet (UV) exposure for that hour or day |
| **Uvindex2** | an alternative UV index element that uses the algorithms and models used by the[US National Weather Service](https://www.cpc.ncep.noaa.gov/products/stratosphere/uv_index/uv_global.shtml) |
| **Visibility** | distance at which distant objects are visible |
| **Winddir** | direction from which the wind is blowing |
| **Windgust** | instantaneous wind speed at a location |
| **Windspeed** | the sustained wind speed measured as the average windspeed that occurs during the preceding one to two minutes. |
| **Windspeedmax** | maximum wind speed over the day. |
| **Solarradiation** | (W/m2) the solar radiation power at the instantaneous moment of the observation |
| **Solarenergy** | (MJ /m2 ) indicates the total energy from the sun that builds up over an hour or day. |

As this project is a comparative study of different machine learning models that are used for weather forecasting, it was necessary to source complete and specific information about the climate. The target feature in all models was the temperature, and the majority of the features were used in several models as lagged features.

#### ETHICAL CONSIDERATIONS

Visual Crossing is a business that provides meteorological data for business intelligence and policy making. The historical data that was used for this thesis has been purchased through a monthly individual membership of £35, therefore it is safe to assume that at least some of the funding comes from memberships. Although it is unclear if this is the only funding that the business accepts. They have two headquarters in the Virginia, United States and in Germany.

At the moment of writing this, there are no known limitations when it comes to sourcing meteorological data from this provider or in general.

#### DATA PRE-PROCESSING

As with any data, the given dataframes contained a number of missing values from three columns: preciptype, windgust and severerisk. There are multiple reasons why a station can register missing values, however in this case it is most likely that there were no cases of precipitation, wind gusts or severe risks during those days.

In order to be able to still use the observations from those days, this thesis made use of the k-nearest neighbor method of filling in missing values and chose not to drop those details. According to <https://www.sciencedirect.com/science/article/pii/S0164121212001586#:~:text=kNN%20imputation%20is%20designed%20to,or%20attribute)%20is%20categorical%2C%20referred>, this way of dealing with missing data “are designed according to Minkowski distance or its variants, and have been shown to be generally efficient for numerical variables (features, or attributes)”. The authors present this method of imputation because it is “designed to find *k* nearest neighbors for a missing datum (*incomplete instance*) from all *complete instances* (without missing values) in a given dataset, and then fill in the missing datum with the most frequent one occurring in the neighbors if the target feature (or attribute) is categorical, referred to as majority rule, or with the mean of the neighbors if the target feature is numerical, referred to mean rule […] Due to its simplicity, easy-understanding and relatively high accuracy, the *k*-nearest neighbor (*k*NN) approach has successfully been used in real data processing applications, such as surveys conducted at Statistics Canada, the U.S. Bureau of Labor Statistics, and the U.S. Census Bureau ([Chen and Shao, 2000](https://www.sciencedirect.com/science/article/pii/S0164121212001586" \l "bib0225))”.

In shorter terms, the kNN searches through all the given instances, finds the most relevant ones for the missing values and creates new inputs replacing the NaN. This method is based on the Minkowsky distance and its variants and it is typically used for numerical features, as it does not perform well with categorical ones.

#### TOKENIZATION

The data sourced from Visual Crossing contained one string column titled “description”, which included a series of short sentenced descripting the weather and temperature on the day. As this was useful information in the context of weather prediction, this project has made use of this column through the tokenization process.

This process converted string data into a numerical representation and made it possible to include this information in the models done afterwards. The tokenization process breaks the strings into features and then categorizes them into clusters with the use of the TfidfTransformer (<https://scikit-learn.org/0.15/modules/generated/sklearn.feature_extraction.text.TfidfTransformer.html#:~:text=Tf%20means%20term%2Dfrequency%20while,good%20use%20in%20document%20classification>.) . According to the sklearn documentation, “ goal of using tf-idf instead of the raw frequencies of occurrence of a token in a given document is to scale down the impact of tokens that occur very frequently in a given corpus and that are hence empirically less informative than features that occur in a small fraction of the training corpus”. Each cluster captured sequential patterns which were then represented through word clouds for ease of understanding.

The tokenized data has been summarized into three word clouds, each representing a separate weather state.

Words on a white background

Description automatically generated

The first cluster (cluster number 0) illustrates a mostly cloudy weather state, however quite rainy.

A close-up of words

Description automatically generated

The second cluster, cluster number 1, represents a mostly cloudy day.

A close-up of words

Description automatically generated

The third cluster, cluster number 2, illustrates a mostly clear day.

Finally, this information inserted into a column in the main dataframe, titled “cluster\_description”, where the data is a mixture 0, 1, or 2, each representing a weather state.

#### DATA ENGINEERING

#### DATA SPLITTING

The first model of the project will be XGBoost and the data that is used is time series. Since XGBoost is not a time series predictive model, this project has split the datetime column (day, month and year of the daily observations) into a column for the month and a column for the year in order for the model to be able to differentiate between the two.

The next step was to use the get\_dummies function from pandas, in order to create twelve columns representing each of the months in one year. According to its documentation (<https://pandas.pydata.org/docs/reference/api/pandas.get_dummies.html> ) , “each variable is converted in as many 0/1 variables as there are different values. Columns in the output are each named after a value; if the input is a DataFrame, the name of the original variable is prepended to the value”.

The project then performed a typical data split for a time series, using the iloc function, where it selects all the data from the start up to, but not including, the last 368 rows (days). Then the test set was created by selecting rows starting from the 368th last row up to the last 3 rows. The final step is to create a *df\_future dataframe* by selecting the last 3 rows of *df\_london\_final*.

The purpose of df\_future is to hold the features for these 3 days, where predictions will later be generated. At this point, it contains only the features, not the predicted values.

# Classification

Firstly the data from four locations has been imported into Jupyter Notebook. The Southern hemisphere includes Brisbane, Australia and the Northern Hemisphere include London, United Kingdom, Manila, Philippines and Bujanovac, Serbia. All locations have been chosen due to their increased number of natural disasters that have taken place in the last ten years.

This part of the project then filtered all four dataframes to include all the meteorological observations from 1st January 2023.

The dataframe used for the training of the model has been initialized as a dataframe with four rows and 1 column. Each location has its own observation row with all the temperatures include in a pandas series, while the target values are initialized as y\_train. The same process is performed again to initialize the X\_test and y\_test series. X\_validation is created as an empty 2-dimensional labeled data structure with a shape of 2 rows and 1 column. It acts as a pre-allocated dataframe before it is populated with the model classification.

The model used for this hemisphere classification is the TimeSeriesForestClassifier from sktime. According to its documentation, it is “an ensemble of decision trees built on random intervals”. Each tree has its own sample sqrt(m) intervals. Each finds the mean, standard deviation and slope for each interval and concatenates to form a new dataset. The model ensembles the trees with averaged probability estimates. The model has low configurability, in order to help performance. The classifier has a small list of parameters that can be tuned, and in this case, the project only used the basic number of estimators (200).

The reason why the TimeSerisForestClassifier (TSFC) is useful is that it uses a new measure called “Entrance” (entropy and distance) gain which identifies high-quality splits. In this article, the authors showcase that the TSF with Entrance gain outperforms TSF using just entropy gain and also two nearest neighbour algorithms. They carry on explaining that “a time series tree is the base component of a time series forest, and the splitting criterion is used to determine the best way to split a node in a tree. “ As the new measure Entrance, is created from both the entropy and distance gain, the authors explain that the entropy of a split is “then the difference between the weighted sum of entropy at the child nodes and the entropy at the parent node […] in time series classification, the number of candidate splits can be large […] therefore we consider an additional measure called Margin, which calculates the distance between a candidate threshold and its nearest feature value”.

In shorter terms, TSF uses a standard decision tree algorithm but with the Entrance gain as the splitting criterion that combines the entropy gain and a distance measure to identify the best splits. As both interpretability and accuracy are required for a classification task, the authors show that TSF can provide both requirements in comparison to other tree based classifiers.

In the project at hand, the predictions y\_pred have been done on the X\_test series, which produced perfect results, meaning that each hemisphere has been predicted successfully based on the temperature of each location given.

The performance metrics used were accuracy, precision, recall and the F1 score, all with a result of 1. Below the confusion matrix confirms the results:

A green and purple squares

Description automatically generated

The model was also tested on new data which are two new locations both in the Southern hemisphere: Lagos in Nigeria and Lima in Peru. Both locations have been successfully predicted as being part of the Southern hemisphere, which highlights the usability of the TSF classifier and the benefit of the Entrance gain measure.

This experiment came from a place of wanting to understand how a simple observation like the temperature registered in multiple places can help in the identification of the hemisphere. It showed how easy it can be to predict the correct labels with this TSF, but it also shows how it can be implemented on never before seen data. This can contribute massively to situations where one might need to predict locations as part of climate observation.

## Forecasting

### LAGGED FEATURES

### TRAINING AND TESTING DATASET

As this project is a time series comparative study, the datetime column that combined the day, month and year of an observation, needed to be split, so the models can learn which temperature belongs in which month. Therefore, the column has been divided into month and year. In order to make it easier for the model to correlate each temperature with the correct month, the *pandas.get\_dummies(*) function was applied. This created twelve columns corresponding to each month of the year through binary representation. The dummy variables have been then concatenated back into the original dataframe, ready to use for modelling.

Afterwards, the initial dataframe has been split into *df\_train, df\_test and df\_future*. The training set took the last 368 days from the initial dataset, the test set contains the data from 368 days before the end up to the last 3 days, and lastly, *df\_future* contains the last 3 rows of the dataframe, which will be used for generating predictions.

The next section of this project is concerned with the application of multiple machine learning algorithms on the prepared data, generating forecasts, and comparing performance metrics in order to understand which model performed best. Each model has a baseline model and a tuned model. Both types undergo performance metrics that are then compared.

### XGBoost

The first model tried in this project was XGBoost, which is an optimized distributed gradient boosting library […] which implements machine learning algorithms under the Gradient Boosting framework. XGBoost, or Extreme Gradient Boosting, is used widely across multiple industries due to its ease of implementation and range of problems it can adapt to. It also facilitates easy computation and does not require highly specific data processing and it is mostly used for regression predictive modelling (however it can be used as a classifier as well).

Gradient boosting is formed out of multiple ensemble algorithms which are constructed from decision tree models. “Each tree is added one a time to the ensemble and fit to correct the prediction errors made by prior models”, highlights <https://machinelearningmastery.com/xgboost-for-regression/>. The authors continues to explain that models in the gradient boosting area, use an “arbitrary differentiable loss function and gradient descent optimization algorithm. This gives the technique its name, “*gradient boosting*,” as the loss gradient is minimized as the model is fit, much like a neural network”.

XGBoost was designed be computationally efficient and effective, and therefore became a go to algorithm for data scientists in multiple types of fields.

Another reason why XGBoost tends to be popular in multiple fields is that due to its boosted trees, it is easy to look at importance scores for each feature that it used. Each score shows how necessary that feature is in calculating the most accurate prediction. Naturally, not everything is needed in forecasting. In a decision tree, the more valuable a feature is, the higher it is on the tree. According to <https://machinelearningmastery.com/feature-importance-and-feature-selection-with-xgboost-in-python/>, “importance is calculated for a single decision tree by the amount that each attribute split point improves the performance measure, weighted by the number of observations the node is responsible for. The performance measure may be the purity (Gini index) used to select the split points or another more specific error function. The feature importances are then averaged across all of the decision trees within the model.”.

In this project, the baseline model is initialized with the *XGBRegressors()* instance, which will implement the gradient boosting targeted for regression tasks. The model is then fit or “trained”, on a dataset that does not contain the target feature (the temperature), but does contain all other features, such as *windspeed, humidity, dew, UV index*, etc.

The model then makes a prediction based on the test dataset (unseen data), where the target feature is dropped again. The final step of this first model is to look at the predictions created by it, by calling *the y\_pred\_xgboost* variable - an array that provides the sequence of numerical values.

In order to try to improve the model, this project looked at implementing the XGBoost grid search.

### Naïve Forecaster

The second model used in this project was the Naïve Forecaster from sktime. It is defined as being one of the simplest form of forecasting, typically used in sales and finance departments. In the context of this project, it utilizes the formed window of feature and target observations and creates the predictions. As opposed to the former model, XGBoost, the Naïve Forecaster does not use features – it only uses the former target variable and predicts it.

The Naïve Forecaster model does not require complex calculations and can be used as an entry-level model in forecasting. It also does not requite too much data as it mostly uses historical data, so it can be used in instances where there is not enough historical information generated. Overall, it can be implemented easily, with little data and used in basic forecasting tasks.

Unfortunately, with such an easy model, there come a number of disadvantages. Even though it is easy to apply and understand, this forecaster cannot be used for more complicated tasks, such as climate modelling. The model can hold a major amount of error, that can affect the overall analysis. It lacks precise prediction, as it does take seasonal variation into consideration, or any other patterns nor external factors that can influence the analysis. As it cannot take seasonal variation into consideration, the model can easily underfit or overfit.

In this project, the Naïve Forecaster was used as a benchmark of all of the models, in order to see how poorly can a model perform with the given data, in the context of climate modelling. The project contains both a baseline model and a grid search for this section.

According to its documentation , the model “obtains the so-called “last window”, a 1D array that denotes the most recent time window that the forecaster is allowed to use”. Its parameters are made out of “*strategy*” , “*sp*” and “*window length*”. This project has made use of the “strategy” parameters that comes as “*last”, “mean” and “drift*”.

* The “*last*” strategy parameter “forecast the last value in the training series when sp is 1. When *sp* is not 1, last value of each season in the last window will be forecasted for each season”.
* The “*mean*” strategy parameter “forecast the mean of last window of training series when *sp* is 1. When *sp* is not 1, mean of all values in a season from last window will be forecasted for each season”.
* Finally, the “*drift*” strategy parameter “forecast by fitting a line between the first and last point of the window and extrapolating it into the future”.

### Auto-ARIMA

The third model used in this project is the Auto-Regressive Integrated Moving Averages. ARIMA is typically used in instances where a time series analysis is required. It works when the data is stationary (the mean and variance should not vary with time) and when the data used is a univariate series. The model has three core elements:

* AR (auto-regressive term) is the past values used to forecast the next value and it is represented by the p value in the PACF plot.
* I (differencing term)
* MA (moving average term) is the “ number of past forecast errors used to predict the future values” and is typically represented by the q parameters in the ACF plot.
* Auto-ARIMA also includes seasonal parameters P, D, and Q, however those are calculated automatically by the model, and not needing anymore user input

Auto-ARIMA in itself is a grid search of the best combination of parameters given the data and the p and q values. Therefore there was no need to create the ACF and PACF plots at this stage.

The model can only be applied to the data once the data is confirmed to be stationary, which was confirmed in the EDA phase of the project through an ADF test. The Augmented Dickey-Fuller test (ADF) is a statistical test that checks the presence of the unit root test in the data, which helps to understand if the data is stationary or not. It has two hypotheses:

* Null Hypothesis: The series has a unit root
* Alternate Hypothesis: The series has no unit root.

According to its documentation, iIf the p-value is above a critical size, then we cannot reject that there is a unit root”. In the context of the project, the data was already stationary, therefore the Auto-ARIMA model was applied with no further processing.

A black background with white text

Description automatically generated

There are multiple other advantages to Auto-ARIMA besides the fact that it is easier to implement than its sibling, Arima. According to Medium, Auto-ARIMA can capture both short term trending and long term trending in a time series analysis. Additionally, because it goes through all the possible combinations of parameters to find the best one, it is easier to implement. It becomes more efficient and user friendly as it eliminates the manual model tuning.

Unfortunately, the model also struggles in understanding complex data patterns as it assumes that the relationship between future values and past values is almost linear in all scenarios. Additionally, it also can be computationally slow and expensive. In this project, the computation of the Auto-ARIMA model took TIME, as opposed to other faster models implemented here. Additionally, the model can be sensitive to outliers and the given data has multiple a number of them given by the analysis in the first part of the project:

A graph with different colored squares

Description automatically generated

The most outliers were part of the *windgust* and *solarradiation* features, which according to the feature importance done by XGBoost, are quite necessary in correctly predicting the temperature. Therefore, the model could be quite influenced by the outliers given the in the two columns.

A graph with text on it

Description automatically generated

As with the rest of the models applied on this data, the project started by initializing the Auto-ARIMA instance with the parameters of *sp=12*, which refers to the seasonal periodicity – in this case, sp refers to the 12 months of the year. The forecaster is then fit to the data and it starts looking for the best combination of its parameters. The predictions are then done on the test set and added to the dataframe df\_test\_w\_preds which holds all predictions from all models.

### SARIMAX

The Seasonal Auto Regressive Integrated Moving Average with eXogenous regressors (SARIMAX) is widely used for time series analysis, due to its understanding of the seasonal component and therefore making it the most advanced version of the ARIMA model. It is based on the ARIMA model with the addition of the seasonal element. Besides *the p, d ,q*, SARIMAX also has the P, D, and Q, which according to its documentation are “order of the seasonal component of the model for the AR parameters, differences, MA parameters, and periodicity. *D* must be an integer indicating the integration order of the process, while *P* and *Q* may either be an integers indicating the AR and MA orders (so that all lags up to those orders are included) or else iterables giving specific AR and / or MA lags to include. *s* is an integer giving the periodicity (number of periods in season), often it is 4 for quarterly data or 12 for monthly data”. This project chose to use SARIMAX instead of SARIMA due to the allowance of use of the exogenous variables, which refers to the most important features that contribute to the prediction of temperature.

As SARIMAX requires the seasonal elements as well, this project made use of the ACF and PACF plots. The seasonal order used for this project was the same as Auto-ARIMA – 12 months.

A screenshot of a graph

Description automatically generated

According to Alharbi and Csala, “the SARIMAX model assumes linearity, although the actual temporal connection and covariance are generally non-linear. The regression approaches assume that both the input and output variables follow a Gaussian distribution and that the high level of uncertainty included in the time series data may significantly affect the performance of certain forecasting models. The SARIMAX model has the ability to minimize the error values and enhance the overall accuracy even when the lengths of the input and output dataset are very close to each other and are in similar directions”.

Unfortunately, SARIMAX also comes with some disadvantages, such as complex parameters selection (p, d, q, P, D, Q). This can prove difficult to implement depending on the data, needing more time to implement and thus making it more complex than other models, that perform better in a shorter amount of time.

In this project, SARIMAX was implement by initialised with the parameters of endog which is the observed time series process y and exog, which represents the array of exogenous regressors. They were followed by the order of the p, d, and q parameters and then the seasonal order of P, D, Q and S. The model then was fit to the data and the obtained predictions were added to *df\_test\_w\_preds*.

This project also incorporated a SARIMAX grid search, due to the high number of parameters needed. The grid search was designed to loop through all p*, d, q, P, D, Q and S* elements and provide all the combinations possible. Then the predictions stemmed from all parameters combinations were added to df\_test\_w\_preds in order to be able to find the best combination possible via the average results of each combination. The best result came from the *ae\_0\_1\_1\_0\_1\_0\_12* combination, which was the only one left in the *df\_test\_w\_preds* dataframe, in order to compare the results with the other models.

### LSTM

The Long Short Term Memory (LSTM) model is one of the recurrent neural networks (RNN) most popular for large datasets for time series analysis. A RNN is a deep learning model that makes predictions based on sequential inputs : “They are distinguished by their “memory” as they take information from prior inputs to influence the current input and output […] the output of recurrent neural networks depend on the prior elements within the sequence”.

Some of the advantages of LSTM are incredibly beneficial for time series analysis, such as handling long sequences of information for a longer period of time. LSTMs also include a memory cell “that can store and retrieve information over long sequences. This memory cell allows LSTMs to maintain important information while discarding irrelevant information, making them suitable for tasks that involve remembering past context”. Finally, they can control the flow of the gradient during backpropagation – the “forget gate, for example, can prevent gradients from vanishing when they need to be propagated back in time. This enables LSTMs to capture information from earlier time steps effectively”.

The main drawbacks of LSTM models include being computationally expensive and therefore can make training slower whilst requiring more resources. This model an also overfit when there is not enough data to train on, since the model learns from the previous sequence. If the training data is too short, the model can become confused. The LSTM model is also not the easier to tune compared to other models in this project – it includes a multitudes of parameters that can give different results, whilst using the same data. It can be difficult to implement as it takes a longer period of time to understand which combination of parameters is the best given the analysis at hand.

This part of the project started with a baseline LSTM model with basic parameters, followed by a model with two Dense layers, a model with updated batch size and a final model that includes all the features.

The baseline model was conducted using the *split\_sequence()* function which “will split a given univariate sequence into multiple samples where each sample has a specified number of time steps and the output is a single time step”. The number of steps used was 30 and both train and test datasets have been ran through the same function.

The baseline model includes one input layer, one hidden layer and one output layer. The model uses the *adam* optimizer and the *mean\_squared\_error* loss function. The *adam* optimizer “ is an optimization algorithm that can be used instead of the classical stochastic gradient descent procedure to update network weights iterative based in training data”. The *mean\_squared\_error* loss function is typically used in regression tasks, because it measures the average squared difference between the predicted and actual temperatures. The higher the value, the lower the performance of the model. The baseline model was then ran through the *predict\_future* function, which takes the model, the data, the number of steps the predicted number of days as input. The predicted number of days for this project was 365, representing the days in one year.

The next step was to try to improve the baseline model with additional layers. This time the architecture included as follows: a first input layer, a dropout layer, second layer, second dropout layer, a dense layer, and the output layer. This model differs from the baseline by adding in dense and dropout layers. Dropout is a regularization method of model that are prone to overfitting. It is used when “input and recurrent connections to LSTM units are probabilistically excluded from activation and weight updates while training a network”. This dropout was added to the input layer of the LSTM model, for a probability of 20% which means that “the data on the input connection to each LSTM block will be excluded from node activation and weight updates”. The dense layer includes a neuron that obtains results from all the other neurons from the layer used previously, which means that it performs a matrix-vector multiplication, where the “where the row vector of the output from the preceding layers is equal to the column vector of the dense layer”.

The following action was to try to improve the LSTM model with dense and dropout layers by tuning the batch size. According to , “the batch size limits the number of samples to be shown to the network before a weight update can be performed. This same limitation is then imposed when making predictions with the fit model”. Furthermore, the batch size is used to control how many predictions are done at one time.

In the case of this project, the model is using a batch size of 1 and 3 epochs. A batch size of 1 means that the model always updates the weights after it processes one sample. It leads to frequent updates but each update is using less data than larger batches.

The model was tried with multiple batch sizes, such as 1, 15, 20, 36, 64 etc. The larger the batch size, the more overfitting the model was doing, therefore the batch size was left at 1.

The final stage of the LSTM section was to add the features needed to predict the target. The data went through the *split\_sequence()* function, which was defined earlier – it splits the time series data set into sequences needed for sequential data, such as LSTM. The function takes *features, target and n\_steps* as parameters.

The first part required two lists to be initialised that hold space for the sequences of features. The loop goes through the *features* array and extracts the sequences. It then appends to the lists and converts them to *numpy* arrays.

The model is then initialised as *Sequential,* with the architecture being the following:

* First LSTM layer with 100 units which are returned as full sequence for outputs
* A dropout layer of 20%
* Second LSTM layer with 100 units which only returns the output
* Second dropout layer of 20%
* A dense layer with 50 units and a ReLu activation function (Rectified Linear Unit) – “linear function that will output the input directly if it is positive, otherwise, it will output zero. It has become the default activation function for many types of neural networks because a model that uses it is easier to train and often achieves better performance”.
* The output layer with a single unit added for the prediction of the temperature

As the model outputs a 2D array with a batch size of 1, the next step was to flatten the predictions and add them to the *df\_test\_w\_preds* dataframe for comparison.

### Prophet

The final model applied to predict the temperatures in London, United Kingdom was prophet sourced by Meta, which follows the sklearn model API. The model was designed to handle missing values and outliers, so the user can focus solely on implementation and forecast.

The prophet model splits the time series into trend, seasonality and vacations (i.e. public holidays that are automatically integrated), which makes it quite powerful compared to models like ARIMA. It has high implementation in industries like finance, travel and e-commerce. For example, in the financial industry, prophet “has changed everything by making it possible to analyze historical revenue and profit data to identify trends and seasons”. It is also much easier to implement than the regular time series models, which could be a major gain if implemented in an area such as natural disaster forecasting, where time is of the essence.

As with most models, its major limitation is that it can be confused by outliers and therefore can crate skewed results.

The first stage of the model is to pre-process the data into only two columns: ds (date stamp) and y (feature target). Following that, the model is then initialised and on the train data and fit. The predictions are done on a period of 365 calendar days.

The output of the model is a dataframe that contains the date stamp column and three other columns, *yhat*, *yhat\_lower and yhat\_upper.* The prediction of the model assigned each row in the future a predicted value, which is located in the *yhat* column. The model then makes it easy to see the forecast components of trend, weekly and yearly. The predictions are added to the dataframe that contains all predictions and the performance measured were applied.

### Performance measures

Each model have had their performance measured calculated:

* MAE (Mean Average Error) – “characterizes the alteration among the original and predictable values and is mined as the dataset’s total alteration mean” . Probably the easier performance measure to use in the context of predicting temperature. MAE tells the user how many units away the prediction is from the actual value.
* MSE (Mean Squared Error) – “measures the **average squared difference between predicted values and the actual values** in the dataset”
* RMSE (Root Mean Squared Error) – “the square root of the mean of the square of all of the error”. It is usually used to compare most machine learning models, “but only to compare prediction errors of different models or model configurations for a particular variable and not between variables, as it is scale-dependent”.
* R2 (R squared) – “indicates the percentage of the variance in the dependent variable that the independent variables explain collectively”. It is measured on a scale from 0 to 100%.
* MAPE (Mean Absolute Percentage Error) – “relative error measure that uses absolute values to keep the positive and negative errors from cancelling one another out and uses relative errors to enable you to compare forecast accuracy between time-series models”. This measure was used in order to understand in which direction the forecast goes, i.e. if the predicted temperature is 7, MAPE will be able to say if it is -7 or +7. It is used to address the limitations of the other performance measures.

Each of the models also includes a code that is able to calculate how long it takes for the fit to compute. It is done through the *time* module. At the end of the notebook, there is a dataframe called *time\_results* that has the name of the model and the corresponding time in seconds.

The details about the requirements of the packages used in this project, alongside the code, can be found at the GitHub link: <https://github.com/nicolaelisabetarosca/weather-prediction>

The next part of the project is concerned with the results stemmed from all models, the comparation of performance measures, and fit time.

# RESULTS

#### CORRELATION (HEATMAP)

A colorful chart with numbers

Description automatically generated with medium confidence

What is a correlation heatmap?

A correlation heatmap is a “data visualization technique that uses color to represent different levels of data magnitude” (<https://www.datacamp.com/tutorial/seaborn-heatmaps>).

The diagonal red line that contains the number 1 means that each of the features correlates with each other, therefore it is not of use. There is also a block of red like boxes that seem to have higher correlation with each other, however those features are similar by definition (tempmax, tempmin, temp, feelslikemax, feelslikemin, feelslike, dew) therefore we will not be considering them as part of the feature importance.

The ones that are important to consider are the boxes that represent a deep shade of blue, such as humidity. The heatmap illustrate a correlation of -0.43 between humidity and the target feature, temperature. This means that whenever one decreases so does the other one and it shows a strong connection between the two. According to the United States National Weather Service (<https://www.weather.gov/lmk/humidity#:~:text=Warm%20air%20can%20possess%20more,if%20the%20air%20is%20warmer>.), “warm air can possess more water vapor (moisture) than cold air, so with the same amount of absolute/specific humidity, air will have a HIGHER relative humidity if the air is cooler, and a LOWER relative humidity if the air is warmer”.

Additionally, there is a group of pink-orange like boxes that show another strong correlation between them and the target feature, temp. Solar energy, solar radiation and the UV index seem to have a positive correlation with the temperature, as all are represented by numbers between 0.57 and 0.59. This attests that every time one features increases, so does the temperature. This makes sense in the context of cloud cover as well – the less cloud cover, the more sun, the higher the temperature.

This heatmap has demonstrated the important features in correlation to the temperature, based on scientific weather information.

## Classification

The first part of this thesis is concerned with a hemisphere classification of all four locations chosen: London, Bujanovac, Manila and Brisbane. The relevance of this classification stems from the hemispheres’ relevance to the seasonal patterns but also the climate and the direct impact they have over weather. An example of that is the [Coriolis effect](https://www.noaa.gov/education/resource-collections/weather-atmosphere/weather-systems-patterns#:~:text=Coriolis%20effect&text=In%20the%20Northern%20Hemisphere%20air,examples%20of%20these%20cyclonic%20systems.), where “in the Northern Hemisphere air veers to the right and in the Southern Hemisphere to the left. This motion can result in large circulating weather systems, as air blows away from or into a high or low pressure area.” In shorter terms, when the Northern Hemisphere is inclined towards the sun, they it experiences summer. In the same time, the Southern hemisphere is experiencing winter. According to [CES](https://www.ces.fau.edu/nasa/module-3/why-does-temperature-vary/seasons.php#:~:text=When%20the%20Northern%20Hemisphere%20is%20tilted%20toward%20the%20sun%2C%20latitudes,the%20sun%20and%20experiencing%20winter.), “the summer season begins in the Northern Hemisphere on June 20 or 21, known as the summer solstice, when the axis of rotation is tilted a full 23.5° toward the sun. The incoming solar radiation strikes Earth directly at a perpendicular or 90° angle to the 23.5°N parallel of latitude […]Fall or autumn in the Northern Hemisphere begins September 22 or 23 […] Winter in the Northern Hemisphere begins on December 21 or 22 […] Spring in the Northern Hemisphere begins on March 20 or 21”.

Looking at the heatmap correlation for all locations, it seems that the impact on weather is done by several factors such as dew, humidity, solar radiation, solar energy and UV index, with metrics ranging from -0.43 up to 0.59. Knowing that the position of a country in a hemisphere dictates factors such as those mentioned above, this thesis chose to look at the impact that hemisphere classification has on temperature (our target).

## Forecasting