|epcc|

Optimising Weather Forecasting Using Citizen Data

Karl Delargy

August 18, 2016

MSc in High Performance Computing with Data Science

University of Edinburgh

Year of Presentation: 2016

Abstract

This MSc project discusses using citizen data to increase forecast granularity, and optimising weather forecasts using data from both the Met Office, and from independent citizen owned weather stations. The problem is first examined, before data is obtained and structured in a database. The data is then explored, identifying any issues, and providing data understanding. Weather forecasts are adjusted using prevalent trends found from official observations. These adjustments were learned from the data, then evaluated on a test set, and achieved an R^2 value of 0.756. These adjustments are discussed, along with any problems generalising these results to forecasts during different times of the year. Finally, weather forecast granularity is increased using citizen data.

Contents

List of Figures

List of Tables

Acknowledgements

Firstly, I would like express my sincere gratitude to my supervisor, Adrian Jackson, for his overwhelming help, guidance and patience throughout the entirety of this project. I would also like to thank the Met Office, who allowed me to scrape their data, and provided insight and advice when requested. And finally, to my friends and family who have been there unconditionally to support me throughout this academic year.

1 Introduction

In today's world, we have access to weather forecasts 24/7. Weather forecasts are printed in our newspapers, reported throughout the day on the news, and smart phones often come with a built-in weather application. There are around 270 weather stations owned and controlled by the Met Office. These weather stations regularly record a number of different observations. The observations from these stations are then used as initial conditions for the equations, which predict the weather at some point into the future at each of the 121 forecast sites in the UK. These forecast sites are no more than 50km apart. However, small scale terrain and small weather phenomena are not resolved by the equations that produce forecasts. To increase the granularity of these weather forecasts, more data must be collected. This shortcoming could be solved using data collected by the British public. Many weather enthusiasts around the UK have purchased small scale weather stations, which makes regular observation observations. These weather stations do not have the same quality measurement apparatus as Met Office weather stations, but may provide insight into localised weather. Any differences in weather from these citizen owned stations and official Met Office weather stations can be used to increase forecast granularity. Additionally, observations from official Met Office weather stations can be compared to forecast weather, and any consistent differences in weather can be used to fine tune weather forecasts at each forecast site.

1.1 Approach

The data must be first identified and obtained. Once it is obtained, it must be structured and piped into a database. The database will be queried to obtain data relevant to analysis needs. The data will then be analysed and inferences will be drawn. Using these inferences, forecasts will be adjusted, and granularity will be increased. These adjustments will be evaluated, before drawing any final conclusions.

1.2 Project goals

Weather forecasting models are very sophisticated, however, these models only make a single prediction for large geographical locations. There may be significant variation across these forecast locations, or *forecast sites*. To resolve this, the forecast granularity must be improved. This will require more data that is currently provided by the Met Office weather stations. Alternative data sources will be explored. After the best among these is identified, it must be obtained and structured into a database so that it is fit for statistical analysis. Pre-existing data sources must also be obtained and similarly structured into the same database so observations can be compared. Additionally, forecasts will be adjusted according the prevalent weather trends observed across all data sources, thereby optimising forecasts. These adjustments will be evaluated, and inferences will be discussed. Or more succinctly:

- Identify and obtain alternative data sources.
- Combine all data sources into a single database, fit for statistical analysis.
- Explore the data so that any issues can be understood and overcome.
- Adjust forecasts according to observed trends.
- Increase forecast granularity.

1.3 Generic Data Science Project Process

Every data science project follows similar guidelines, and the steps outlined below will be referred to throughout this project.

1. Determine project goals.
2. Identify data sources and resources.
3. Obtain relevant data.
4. Clean and prepare the data.
5. Mine the data.
6. Analyse results.
7. Exploit knowledge obtained.

Go back and repeat steps as necessary [1].

2 Background

2.1 Weather Forecasts and Observations

Considering how frequently the British population discuss the weather, these weather reports are greatly taken for granted. Very few truly know how to correctly interpret a weather report, or understand how a weather report is generated. This chapter aims to provide a reasonable understanding of weather forecasts, observations, and explain why weather reports are so crucial to everyday life.

2.1.1 Atmospheric Models

Atmospheric models are designed to simulate atmospheric conditions using mathematical equations. Using these models we can track and observe how atmospheric conditions vary over time. The models are initiated by inputting the initial conditions, which include weather observations collected at the point of model initialisation. The model will then use equations of fluid dynamics and thermodynamics to predict the state of atmospheric conditions at some point in the future [2]. These equations are called *primitive equations*, and are often intractable, and so must be solved using numerical methods which require considerable computational power [3]. Despite advances in modern technology, atmospheric models are limited by a number of factors:

1. Although mathematically brilliant, the current models can only represent a simplified version of the atmosphere. Due to the chaotic nature of the atmosphere, it is difficult to model every atmospheric condition perfectly. More sophisticated, inclusive models could be utilised, but these simulations take much longer to run, rendering the results irrelevant.
2. The primitive equations compute atmospheric conditions at predetermined grid points. Any phenomena smaller that the grid space will not be resolved by the models [4]. Models also struggle to resolve the boundary layer, which opens up scope for increased forecast granularity and optimisation.
3. Often there are initial condition errors and boundary condition errors, and due to the chaotic nature of the atmosphere, small mistakes in initial conditions can have a significant impact on results.
4. Small scale terrain features are unlikely to be handled properly.

These factors can all impact the reliability of weather forecasts, and hence there is significant room for improvement. One method for narrowing the error, and generating the most likely outcome, is *ensemble forecasts*. Ensemble forecasts is where multiple models are run, or the same models are run, but with different initial conditions. The results from these modes are then aggregateed. Not all these models will be given equal

weight. Depending on the current climate, some models may achieve better performance, some may contain biases, and emerging patterns may help an expert to choose one model over another.

2.1.2 Climatological Models

Climatological forecasts are made by averaging weather observations over time for a specific time of year, and the forecast is based solely on that information [5]. For example, if it rains 1 in every 100 days in Morocco, then we would forecast that it will not rain, and hence, this forecast will be right 99% of the time.

Hybrid methods may also be considered, where the difference between forecast temperature and observed temperature can be used to adjust forecasts accordingly. This type of forecast optimisation will be explored further in Section 5.7.

2.2 The Importance of Weather Forecasting

Modern weather forecasting is considerably expensive, so why are governments and other agencies so motivated to improve weather forecasts?

1. **Warning:** Extreme weather, either off shore or land based, must be predicted to warn those who may be at risk. In the USA alone, there have been 196 instances of severe weather since 1980, with a combined damages cost of $1.1 Trillion [6]. If these extremes of weather can be predicted, countries can better prepare, saving lives [7], and reducing damages to property and livelihoods.
2. **Travel:** Shipping and air routes can be planned in a more economical fashion, thus making both products and travel cheaper. Weather reports also warn of hazardous weather, so ships and boats can take alternate, safer routes.
3. **Planning:** Weather forecasts inform the public, who can then take the weather into consideration when planning trips or events.
4. **Agriculture:** Farmers need weather forecasts so they can plan their schedules. For example, bailing hay can only be done in dry conditions, so farmers need exploit dry weather in order to complete this task.
5. **Estimating energy demands:** For example, a heat spike would result in an increased number of AC units in operation, resulting in a larger drain of energy. If this is not prepared for, there could be an energy shortage.
6. **Military operations:** Different weapons and tactics require knowledge of weather conditions prior to deployment.

2.3 Citizen Data

There are approximately 270 weather stations owned and used by the Met Office across the UK [8]. These sites record the observations which are used as the initial conditions for the primitive equations. However, this isn't *all* of the data which is available. Any citizen can go out and buy a weather station for their garden, for their neighbourhood, or even for their child's school. These privately owned weather stations have the capacity to record a range of weather observations, and their numbers are far in excess of the total number of official weather stations. It has already been noted in Section 2.1.1, that changes in the weather due to local terrain are not resolved by current atmospheric models, and that phenomena smaller than forecast grid points is not incorporated, meaning that models do not resolve the boundary layer. These problems may be solved by incorporating these weather stations owned by everyday citizens, or *citizen data*.

At the beginning of this project, there were approximately 940 citizen data weather sites in mainland UK, however, this number has grown considerably since then. This means there is a wealth of data which remains largely unused. There are instances where local weather conditions significantly vary from the regional forecast due to the specific geographic or micro-climate conditions in that particular area. Using these citizen stations, we could increase the granularity of forecasts by adjusting them according to observed weather. For example, consider a forecast area which has a predicted precipitation probability of 0.4. Imagine this area contains a mountain that impacts the precipitation probability, changing it to 0.2 on one side, and 0.6 on either side of the mountain. Utilising citizen weather stations, these local differences can be recorded, enabling weather forecasts to be more accurate through increased forecast granularity.

Citizen weather data does have its disadvantages. Firstly, official weather stations go through a rigorous placement process to make sure they are not unduly affected by their immediate surroundings. This ensures that their observations represent the wider area, rather than local instances of weather. Citizen weather station placement is up to the owner of the station, and most citizens are unlikely to place their station in a location which is as representative of the wider area. Secondly, official weather stations all record a similar range of observations, and use top quality recording equipment. Citizen weather stations vary significantly in terms of the quality and range of observations recorded. This raises questions as to the credibility of each individual citizen weather station, and citizen weather data as a whole. Recall that small changes in initial conditions of the primitive equations can significantly change the resultant forecast. Thirdly, there is no sanity cap on citizen data. A faulty station could record a wind speed of 2000mph instead of 20.00mph, for example, and upload this to the database. Despite the fact severe errors of this kind are relatively easy to spot, more inconspicuous observations may go unnoticed.

2.4 Web Scraping

Web scraping is a technique used to extract information from diversified web resources onto a local file or database [9]. In other words, the objective of web scraping is to take machine readable data, and transform it into a format which is human readable and/or fit for statistical analysis. There are a variety of methods available, ranging from simple browser plugins, to processing *Hyper Text Markup Language* (HTML) [11] code. However, most commonly it is performed by automated programs, called *bots* [10], which parse HTML code and extract any information of interest. Web scraping is commonly used when data cannot be downloaded, or to interface with legacy systems.

To understand how web scraping works, we must first consider the anatomy of web pages. Web pages are largely composed of HTML code (and some times Java Script), which is a structured hierarchy of *boxes*, surrounded and defined by *tags*. Large boxes will contain many smaller ones, and these smaller boxes may contain even more boxes which are smaller again, and so on. Different tags perform different functions, and may contain different properties [12]. Tags can also belong to groups called classes, which can help with scraping as they allow us to target information in a web page more specifically. In addition to this, a good understanding of HTML code is required to be able to locate and extract the information that is needed.

In order to scrape the data, we require a mechanism for navigating throguh this structured hierarchy, or parse tree. A common device used for this navigation is called *Beautiful Soup*. Beautiful Soup is a Python library used for pulling data out of HTML and XML files. It works with most common parsers to provide idiomatic ways of navigating, searching, and modifying the parse tree [13]. In other words it breaks up the HTML into a kind of map that we can navigate with greater ease to find the data of interest. Alongside this, we need something called *mechanize*. Mechanize is a library which allows bots to disguise the fact they are a bot, and masquerade as a normal browser. Officially, it can be described as stateful programmatic web browsing in Python [14].

Once the box containing the pertinent information is found using Beautiful Soup, we can then remove the tags by selecting only the text component of that box. If the data is attached to any noise, we remove it at this stage. The data can then be written to file, or printed to the command line as appropriate.

Web scraping can be an elegant tool when applied correctly, and in the right circumstances. A bot can sift through thousands of URLs rapidly, extracting only the required information, and can be programmed to extract data in the exact format desired. Additionally, browsers may not display all the information which is contained within the HTML code. For example, in the weather data used in this project, wind direction is presented on a browser as a categorical variable (S, NW, NNE etc). However, hidden in the HTML, it is saved in degrees, giving us a more accurate measure of the wind direction. For all of it's benefits, web scraping does bear some unfortunate drawbacks.

1. It heavily relies on web formats remaining unchanged. If a website undergoes a revamp, thereby significantly changing the HTML code, the scraper must be

rewritten to accommodate the new format.

2. A single data type may be saved in several ways on the same page. For example, dates can be saved as `DD/MM/YYYY`, `YYYY-MM-DD` or `DDMMYYYY`. These must be converted to a single format to allow for comparison.
3. Some websites may have defence mechanisms in place to stop bots from accessing their websites, which can be troublesome to work around.
4. Some URLs change regularly, for example, an observation website which changes its URL each time an observation is updated. In order to get the most recent observation, bots must be intelligently designed to determine the most recent URL.
5. HTML codes of older websites can be very disordered, and it is often problematic to find and extract the pertinent information from such websites.

There are a variety of languages and tools for building bots, and throughout this project the bots were written in Python, importing both Beautiful Soup and Mechanize libraries.

2.5 Databases

When data volume is very small, all relevant information can usually be saved on one table without facing any significant problems. However, as data sets grow, it is better practice to store data in databases.

2.5.1 Introduction to Databases

Databases are a means to efficiently store and organise data in such a way that it can be easily accessed, managed, and updated [15]. They are made up of the following:

- **Fields:** Each column in a table is called a field, and this represents a variable.
- **Records:** Each row of a table is called a record, and this represents an observation.
- **Tables:** Columns and rows combined together make up a table, and this can be a single observational unit.
- **Schemas:** A schema is the skeleton structure that represents the logical view of the entire database. It defines how the data is organized and how the relations (tables) among them are related [16].
- **Queries:** A database can be queried to access only the data of interest, and ignore the rest. This is a powerful tool when conducting statistical analysis.

Databases usually come with a DataBase Management System (DBMS), which is system software that allows users to create, retrieve, update and manage data. Often they provide a means to back up data and improve query performance.

2.5.2 Types of Database

The two most common types of database are *relational* and *Not only SQL* databases (NoSQL). In relational databases, tables are known as *relations*, and relations have associated *relationships* with other tables. Relational databases have strict schemas, meaning they are only suitable for structured data.

Each table in a relational database contains a *primary key*, and must be indexed. A primary key is a field which is guaranteed to have a unique entry for each record in a table. If a user attempts to add a new entry with a pre-existing primary key value, the database will not allow it. Indexes help the DBMS to navigate more quickly through a database to find the relevant data. The entries in the field chosen as index should be unique or rare within that field. Often the primary key will be chosen as an index, although this is not a requirement, choosing the index is up to the user.

Relational databases can be normalised to improve performance of queries, reduce data duplication and to make the database more understandable. There are different levels of normalisation, which are outlined below:

1. **1st Normal form:** Repeated sets of related data are removed and put into an individual table, where each set of related data will have its own separate table and is identified with a primary key.
2. **2nd Normal form:** All of first normal form, *and* the non-primary key columns must depend on the whole primary key, as opposed to just part of it.
3. **3rd Normal form:** All of the normalisation from the 1st and 2nd normal forms, *and* non-primary key columns must depend only on the primary key [17].

The database in this project, called *weatherDB*, was designed to fall under the umbrella of 3rd Normal form.

Relational databases are subject to several constraints which are part of the database schema definition. These constraints are in place to maintain data integrity, and protect the database from being filled with nonsensical entries and becoming corrupt. Constraints are rules created at design-time, and the following are some of the most common constraints that data can fall under:

1. **Not null:** Where data is not permitted to have a null value.
2. **Uniqueness:** A new entry will not be permitted if the value of its primary key field already exists as a primary key within the table.
3. **Domain constraint:** The value for the data must fall within the predetermined range of values.
4. **Referential integrity:** When a table refers to a foreign key, a new record may only be added if it refers to a valid foreign key.

Users interact with relational databases via a special-purpose programming language called *Structured Query Language* (SQL). The scope of SQL mechanisms includes data insert, query, update and delete, schema creation and modification, and data access control [18]. The result of an SQL query is another table, except this table only has the information specified within that particular query, and data may be from multiple tables which are *joined* together. Relational databases have something called transactions, where a sequence of SQL are statements executed as a single unit or not at all. Deleting from a table is an example of a transaction, which is useful, because if you delete the wrong information, the transaction can be canceled without the loss of any data.

NoSQL databases are non-relational databases. They have no schema, and this flexibility makes them suitable for structured, semi-structured or unstructured data. They have no transactions or join operations, and can be normalised or de-normalised.

Weather data must maintain data integrity, and the data gathered in this project is perfectly structured. There are important relationships between the data, and joins are crucial for statistical analysis. For these reasons, a relational database will be used to store and manage the data in this project.

2.6 Machine Learning

The standard way to teach a computer to perform new tasks, is to update it's programming so it knows how to complete said task. Machine learning is the science of getting machines to learn without being explicitly programmed [19]. This field of study involves programming algorithms that perform predictive analysis on data, allowing machines to predict which scenarios are most likely to happen. Usually the data is divided into two groups, a training set, and a test set (often 80/20 split, although this can vary depending on the size of the data). The algorithm is "learned" using the training set, and then evaluated using the test set to check for model accuracy and for over-fitting.

There are two main types of machine learning: *supervised learning* and *unsupervised learning*. In supervised learning, we have an input, X, and an output, Y. Our job is to find the mapping of X to Y. When X is a categorical variable, we use a classification algorithm. Many classification algorithms exist, and these range from relatively simple to very sophisticated. If X is a continuous variable, then we use a regression learning algorithm. There are many ways to run a regression, this decision will be based on the data at hand. With unsupervised learning, we only have the input X, and our job is to find regularities/structure within the input space. The most common unsupervised learning technique is *clustering*, and there are different types, using different parameters. Which type of clustering is implemented depends on the data, and the desired result.

3 Obtaining the Data

This chapter describes how data weather data was identified, obtained, cleaned and prepared. This amounts to parts 2 through 4 of the data analytics project process, mentioned in Section 1.3. The data cleansing techniques used in this project will be introduced in this chapter.

3.1 Identifying Potential Data Sources and Resources

This section describes the process of determining what data was required to complete the project goals, along with establishing where such data was available. Not all of the weather data at hand will help us accomplish our project goals. We must carefully consider what may or may not be useful. The following were considered:

1. Citizen observational data.
2. Official Met Office observational data.
3. Official Met Office forecast data.

Hereafter, citizen observational data will be referred to as *citizen data*, official Met Office observational data will be referred to as either *official data* or *observational data*, and official Met Office forecast data will be referred to as *forecast data*.

Initially, it seemed obvious to obtain the observational and forecast data from the Met Office. Unfortunately, it was not possible to simply download their data or obtain it by any other means. The alternative option was to access relevant data through online stores, such as Met Office DataPoint - a service to freely access available Met Office data feeds online [20]. This was useful for viewing day to day forecast data for a specific area, but didn't fulfil the requirements of this project.

The citizen data could be obtained from two possible sources. Firstly, many of the citizen stations have their own website with the data regularly uploaded, and this data can be easily downloaded from each page. However, it's unfeasible to regularly download data from hundreds of these websites because these websites are independent of each other, meaning a different scraper would be required for each station. The data would also be stored with different units, and with different formats. All of this is very time consuming. Secondly, the Met Office recently created a Weather Observation Website (WOW). WOW is a website dedicated to citizen data, where people can upload observations from their citizen station, and anyone can access it through a GUI in the form of a map. Although the data is not downloadable, all relevant information is contained within the one base website, and so this was chosen as the data source.

Due to the data from both these sources not being freely downloadable, the best course of action was to scrape all three kinds of data from their respective sources into a database.

3.2 Getting Started

This section describes how to scrape weather data from the Met Office. We start by describing what tools are available and how they help. This will be followed by a generic process for scraping a website.

3.2.1 Useful tools

The first set of tools to be introduced are *Chrome DevTools*. These come built into Chrome web browsers and can be extremely helpful for web scraping. They allow users to view the HTML code of any page with the click of a button, and also allow users to inspect elements on a particular web page. This allows for quick navigation of a web page's HTML, so relevant data can be easily located.

Once we have located the relevant data with the HTML, the next step is to navigate through the parse tree in order to scrape it. However, often the data we seek is accompanied by other unwanted data. One approach is to scrape the data along with the accompanying data, or *noise*. Assuming that the noisy data doesn't vary from page to page, we could then simply use a series of `.replace` functions. If the noisy data does vary, we need a more advanced, dynamic tool called *Regular Expression*, or *regex* for short. Regex allows users to input sequences of characters which define search patterns. IT uses flexible wild cards and a range of string identifiers to pinpoint the exact data required. Regex has many different functions, and comes with excellent documentation and "cheat sheets".

Weather stations update their forecasts and observations regularly, meaning that the bots must be executed in a similar fashion. Cron is a unix, solaris utility that allows tasks to be automatically run in the background at regular intervals by the cron daemon. These tasks are often termed as cron jobs in unix, solaris. Crontab (CRON TABle) is a file which contains the schedule of cron entries to be run and at specified times [21]. In other words, programs can be scheduled to run at regular intervals without having to press a button, or for the device running the bots to be logged in. This is extremely useful for scraping data where updates are frequent, however there are some drawbacks. For this project, the bots are running on a personal laptop, which means it must be kept within range of Internet signal and be switched on. For various reasons, the laptop in question did not fulfil both these requirements at times, and as a result, there is some missing data. Crontab also requires the path to the file to remain unchanged, if files are rearranged then any existing cron jobs must be edited appropriately.

When scraping data from many URLs that belong to the same base website, they will often be scraped with the same bot. This means the same scraper can be used for each URL, meaning the URLs can be bundled together into a vector, and the data can be scraped using a `for` loop of the URLs. A problem arises in that websites are often removed, and hence that element of the `for` loop will fail, and so the rest of the bot will fail. To avoid this failure, Python provides `try:` and `except`, which allow for

failure of individual elements, meaning the loop can finish its cycle. This is vital to web scraping among other applications.

3.2.2 Generic Web Scraping Process

Each web scraping project will vary, but the following guidelines should be followed where possible.

1. Find the URL(s) which contain the data of interest.
2. Navigate through the HTML parse tree to find where the data is located.
3. Select that part of the HTML.
4. Strip away any non relevant data.
5. Print data to screen, or write to file as appropriate.

3.3 Weather Data

This section will describe how the citizen data, forecast data and observational data were scraped. As expected with any real-world data set, a number of issues presented themselves, and these will be reviewed. Throughout this section we will gather initial insights into the data.

3.3.1 Citizen Data

There are hundreds of citizen weather stations in the UK, but unfortunately there is no list available which contains the site IDs. These site IDs are essential to the scraping process, as they must be loaded onto the end of the base address of the Met Office website to obtain the citizen observations. The only option was to collect these IDs manually by clicking on each station on the GUI map of Great Britain, presented on the browser. Once collected, it is a simple process of combining the IDs with the base address, and putting them into a vector. As mentioned in section 2.3, citizen weather stations vary a great deal in their design and sophistication, meaning that some weather stations record more observation types than others (where type of observation means temperature, rainfall, etc). The initial plan was to write all of these observations in a single file, and if a citizen station did not record a given type of observation, then `"Unknown"` was written in its place. This design would have been functional for a relatively small data set, for example, data collected over the course of this project. However, it is bad practice and would not work for databases with larger data sets. The database redesign strategy was already described in Section 2.5.2, but it meant the data set was still littered with `Unknowns`. The crontab for the citizen data was set on a schedule to run every hour on the hour.

A variety of web scraping problems were encountered throughout this project. The most time consuming of these was the website revamps. Not long after the scraper was complete there was an entire website revamp. This meant almost all of the previous scraper was rendered unusable. After rebuilding the scraper again, the websites underwent more adjustments, but to a lesser extent. However, a troubling new feature was introduced, whereby the URL changes every time the observation is updated. This was solved using the search page of the Met Office website, and the process is outlined below:

Methodology 1:

1. Make a vector containing all of the station names.
2. Add each element of this vector to the end of the search URL and cycle through the search pages.
3. This page will provide links to the closest matching stations.
4. The closest matching string will be the most recent URL for that station.

After this had been constructed, it became apparent that the most recent observation could be found using a fixed URL, which could be hard coded into the bot, however this method would be used again for both the observational and forecast data. The bot was originally designed to write `Unknown` for the values that were unreadable, but since the database design changed, these were surplus to requirement. This bot was then redesigned to only write values which existed, and the `Unknowns` in the data which were previously collected, needed to be removed. Additionally, dates were saved in different formats both across and within the three data categories. This meant the data underwent a thorough cleaning process to make these consistent. At the time, the data were saved in separate CSV files, and the following cleaning process took place in Microsoft Excel. A separate column performed a test to check if the word "Unknown" was present in a row. If it was, the cell in this new column was "x", if it was not, the cell in this column was assigned a "-". The rows were filtered to only show rows with an "x", and then these rows were removed from the file.

3.3.2 Observational Data

The URLs for the observational data were found using a variation of Methodology 1. Once found, a similar process was applied to the observational data, in order to scrape it. Data observations from the last 24 hours are presented in the browser, and hence are available from scraping. Methodology 1 has been extended slightly to cover two days worth of observations (24 hours spans over two calender days). The code for the first day was essentially replicated for the second day, and edited to include both these dates. The crontab was set up to scrape these observations every 12 hours, on the hour.

Similar problems were prevalent in observational data that were encountered in citizen data, such as: website revamps, changing the design of the data, and cleaning. For observational data (and forecast data) another problem was encountered. The Met Office

attempts to block bots from crawling their website using the *Robots exclusion standard*, commonly referred to as *robots.txt*. The standard specifies how to inform the web robot about which areas of the website should not be processed or scanned [22]. Robots.txt is completely advisory, and can be bypassed by adding in a few lines of code to the bot shown in Listing 1.

Listing 1: Bypassing robots.txt

```
ua = 'Mozilla/5.0 (X11; Linux x86_64; rv:18.0)
Gecko/20100101 Firefox/18.0 (compatible;)'
br = Browser()
br.addheaders = [('User-Agent', ua), ('Accept', '*/*')]
br.set_handle_robots(False)
```

This code is inserted into the bot just before mechanize is used, and allows the bot to scrape as normal. To ensure that this was not breaking any rules, a confirmation email was sent to the Met Office. They replied with assurance that this was allowed within the bounds of this project.

3.3.3 Forecast Data

The methodology used here was similar to that of the observational data, except it collected data over 7 days. Again, there were problems with website revamps, robots.txt, changing the design of the data, and cleaning. The forecast bot also records three dates: issue date, today's date, and the forecast date. These were stored in three different formats, and required the use of regex `split` and `search` functions to convert these to a single format, ready for processing. The crontab for this scraper was set up to run every hour, on the hour.

3.4 Station Profiles

As previously mentioned in section 2.3, citizen stations greatly vary in terms of design and sophistication. This means that some stations may record observations to a greater degree of accuracy than other stations, or may record different types of observation. Fortunately, each station is graded for each type of observation it makes, and these grades must be scraped into their own separate table as they will become useful during the analysis stage. The grades are stored on the same web page as the citizen data, and were relatively simple to scrape.

Along with the grades, the name of the station, latitude and longitude were also recorded. Latitude and longitude will be useful for analysing specific areas, but they can also be used to locate the nearest official weather station or forecast location, and record the distance to it. This process is outlined generally by Methodology 2.

Methodology 2:

1. The latitude, longitude and ID of each citizen and official Met Office forecast location is scraped into a vector and copied into a `.c` file.
2. Using `structs`, the distance from each citizen station to every forecast location is calculated using a standard latitude-longitude distance calculation formula.
3. This distance and the ID of the forecast location are then written to a file. This ID is saved under the field "forecastID".

Knowing the closest forecast location is very useful as it allows us to compare weather predictions and observational data to look for any interesting trends, or unusual values. Most of the forecast locations are official weather stations, and so the forecasts can be easily compared to official observations. This makes the analysis more efficient and intuitive.

4 Database design

This chapter will examine the DBMS used in this project, along with its accompanying unified visual tool. Next, we will describe the data, and examine relationships between tables. Finally, we will review the DBMS, and discuss some of the problems encountered with the DBMS during this project, and further afield.

4.1 MySQL

MySQL is a popular, high performance, opensource, relational DBMS. For most applications, it's very difficult to go wrong with MySQL due to its simplicity. It's a very scaleable, robust, and full-featured DBMS, used by many top websites, such as YouTube, Twitter, and Facebook, among many others [10].

MySQL can be used in conjunction with MySQL Workbench, a unified visual tool which provides data modeling, SQL development, and comprehensive administration tools for server configuration, user administration, backup, and much more. It is platform agnostic, and makes interactions with relational databases much easier. Due to these features, MySQL is perfectly suited for our weather data, and so will be used for all interactions throughout this project.

4.2 Design Features

From relational database theory, we know that 3rd normal form means data in databases is broken up for organizational purposes, and to improve the performance of queries, as previously explained in Section 2.5.2. Every table should focus on describing one characteristic, and the associated data which is necessary for describing that characteristic. However, we don't want to break up the data too much, where lengthy and complicated queries are required to access pertinent data.

Previously, relational databases have been discussed, and how the tables in relational databases are "related". But what is actually meant by this? To link tables, thereby creating a relationship, we use something called a *foreign key* [23]. A foreign key is a field in one table that refers to the primary key of another table [24]. The table which contains the foreign key is called the child table, and the table it refers to is called the parent table [25]. The purpose of the foreign key is to identify a particular row of the referenced table, and these relationships should try to emulate real world relationships.

Relationships between tables can take three different forms: one-to-many, many-to-many, and one-to-one. One-to-many relationships are the most common. This means that a parent record can refer to several child records in another table. "Many" does not necessarily mean more than one, in fact, it can mean 0 or 1. But it should be noted that each child record can reference only one parent record [26].

With many-to-many relationships, a row in one table can refer to many rows in another, and vice versa. This is possible through the use of a *junction table*. A junction table is a third table whose primary key consists of the foreign keys from both the first and the second table [27].

One-to-one relationships are relatively rare, because all the data in these two tables should be contained within one. These relationships usually occur when data has been broken up into too many tables. With one-to-one relationships, each row can refer to only one row in another table and vice versa.

4.3 Creating Database Schema

Database schemas can be created using SQL commands from the terminal, but editing a table is relatively troublesome, and SQL requires strict syntax. This makes database schema creation slow and unnecessarily problematic. To make this easier, MySQL provides a unified visual tool, as mentioned in Section 4.1, called MySQL Workbench. MySQL Workbench provides a means of easily creating and editing database schemas, with support for creating foreign keys and adding attributes to fields. Once the schema is created, it is easily *forward engineered*, i.e. the SQL commands are generated and run by MySQL Workbench. The database can then be updated as normal through the command line.

4.4 Integrating MySQL and Python

Initially, our data was being scraped directly into CSV files. These CSV files can then be imported into MySQL using a python connector called PyMySQL. PyMySQL is an open source Python-MySQL driver. Fortunately, PyMySQL is hosted on GitHub, meaning it is easily accessible and trivial to download. A very simple PyMySQL example is shown in Listing 2.

Listing 2: PyMySQL example.

```
import pymysql

conn = pymysql.connect(host='127.0.0.1',
            unix_socket='/tmp/mysql.sock',
            user='root', passwd='Password',
            db='mysql', charset='utf8')

cur = conn.cursor()

siteID = 1
UV7 = 2
```

```
cur.execute("USE weatherDB")
cur.execute("INSERT INTO forecastUV(siteID,UVRating)
    VALUES (%s,%s)", (stationID,UV7))

cur.connection.commit()

cur.close()
conn.close()
```

This program logs in MySQL, chooses a database, and enters two values into table `forecastUV`. There are two types of objects we are concerned with: firstly, the *Connection object* (`conn`), and secondly, the *Cursor object* (`cur`). The connection object is responsible for creating and maintaining a connection with the database, handling rollbacks, keeping the database informed and creating new cursor objects [10].

Cursors are essentially information trackers. For example, it keeps track of which database is currently in use, recent queries, and much more. Connections can have many cursors, for example, if you have several databases, and data must be written across all of them.

When utilising a cursor connection approach, we must be wary of closing connections. If we forget to close many of these, the database may fail [10], and this is what `cur.close()` and `conn.close()` accomplish at the end of Listing 2.

4.5 SQL Data Types

SQL requires each field to have a data type. There is a rather lengthy list of these data types, and those which have been used throughout this project will be outlined below.

INT: Integer values that fall within the (signed) range: [-2147483648 , 2147483647].

BIGINT: Integer values that fall within the (signed) range: [-9223372036854775807 , 9223372036854775807].

DECIMAL(p,s): Exact numerical, precision p, scale s. Fall within [$-10^{38}+1$, $10^{38}-1$].

VARCHAR(x): String of variable length, with a maximum of x characters.

DATE: Dates saved in the following format: %Y-%m-%d.

TIME: Time of day saved in the following format: %H:%M or %H:%M%S.

ENUM: Enumeration type. Only a predetermined set of values will be accepted. These are defined by the user.

4.6 Data Description

As mentioned in section 3.1, there are three kinds of data from three sources: observational data, citizen data and forecast data. Although all three take similar measurements, there are some distinctions which must be made clear. This section will focus on listing the column headers of each table, with a brief explanation where necessary, and highlighting any distinctions between similar types of observation across the different tables.

The most natural starting point is the "profile" table for official weather stations. The reason for beginning here is that this table contains no foreign keys, meaning it is a parent table only, and never a child table. The forecast location/official station profile table is called `stationProfile`, and contains the following fields:

- **ID**: This is the primary key in `stationProfile`. It must be not null, unsigned and automatically increments for new entries. This has data type INT and was designated as an index.
- **siteID**: This is the weather station/forecast ID, this must be not null, unsigned and unique so that foreign keys can reference it and has data type INT, and was designated as an index.
- **latitude** Latitude for the weather station/forecast location. This has data type DECIMAL(20,10).
- **longitude** Longitude for the weather station/forecast location. This has data type DECIMAL(20,10).
- **heightAboveSea** This is the weather stations/forecast locations height the sea level. This has data type DECIMAL(20,10).

Although `siteID` is a unique field, it is easier to add a new column `ID` to use as primary key, especially when adding new entries [10]. So long as `siteID` has a uniqueness attribute, foreign keys can still reference it. Observational data, forecast data, and the citizen station profiles will all refer to `stationProfile`. With that in mind, it is logical to describe the fields within the citizen station profile table, called `citizenProfile`.

- **ID**: This is the primary key in table `citizenProfile`. It must be not null, unsigned and automatically increments for new entries. This has data type BIGINT and was designated as an index.
- **siteID**: This is the citizen weather station ID, this must be not null, unsigned and must be unique so that foreign keys can reference it. This has data type BIGINT and was designated as an index.
- **siteName**: This is the name of each citizen weather station and has data type VARCHAR(60).

- **exposure**: Exposure ratings relate to the site of the temperature and rainfall instruments only, which should ideally be at ground level. Sensors for sunshine, wind speed etc are best exposed as freely as possible, and rooftop or mast mountings are usually preferable. Exposure guidelines are based on a multiple of the height, h, of the obstruction above the sensor height; the standard is a minimum distance of twice the height (2h). Thus for a rain gauge at 30 cm above ground, a building 5 m high should be at least 9.4 m distant (5 m less 0.3 m, x 2), and a 10 m building should be at least 17 m from a thermometer screen (10 m less 1.5 m, x2) [28].
- **temperature**: Rates the measurement apparatus for collecting temperature observations. Ideally, calibrated mercury-in-glass thermometers or calibrated electronic temperature sensors. [28]. This has data type ENUM.
- **rain**: Rates the measurement apparatus for collecting rainfall observations. Ideally, a "five-inch" copper rain gauge, with deep funnel, the rim of the gauge level and mounted at 30 cm above ground level, meeting the minimum exposure requirement of being at least 'twice the height' of the obstacle, away from the obstacle [28]. This has data type VARCHAR(5).
- **wind**: Rates the measurement apparatus for collecting wind observations. This has data type VARCHAR(5).
- **urbanClimateZone**: Used to rate the physical nature of cities for urban climatologists in terms of surface cover and surface structure [29], and has data type VARCHAR(5).
- **reportingHours**: Indicates how regularly weather observations are taken and uploaded. This has data type VARCHAR(5).
- **latitude**: Latitude for the weather station. This has data type DECIMAL(20,10).
- **longitude**: Longitude for the weather station. This has data type DECIMAL(20,10).
- **forecastID**: This is the ID of the closet official weather station providing a weather forecast. It is not null, unsigned and is a foreign key which refers to `siteID` in the `stationProfile` table. It has datatype DECIMAL(20,10).
- **nearestWeatherForecastLocation**: The distance to that station, and has data type INT.

An overall site rating can then be calculated from the following:

5* = E5, T=A, R=A
4* = E >= 3, T=A, R=A
3* = E >= 3, T[=A,B or C], R[=A,B or C]
2* = E >= 1, T[=Any], R[=Any]

1* = E =0,1,R or U, T[=Any], R[=Any]
(Where E = Exposure, T = Temperature, and R = Rainfall) [28].

Next, we are going to describe the blueprint for each of the citizen station observation tables.

- **ID**: This is the primary key in citizen observation tables. It must be not null, unsigned and automatically increments for new entries. This has data type BIGINT and was designated as an index.
- **siteID**: This is the citizen weather station ID, and must be not null and unsigned. Additionally, this is the foreign key referring to `siteID` in table `citizenProfile`. This has data type BIGINT and was designated as an index.
- **<OBSERVATION[i]>**: There are 8 different observations made from citizen weather stations, `<OBSERVATION[i]>` can be one of: `dewTemp`, `humidity`, `pressureSea`, `pressureStation`, `rainfall`, `temp`, `windDirection`, or `windSpeed`. Each of these are described further in section 4.7, and hold a range of data types.
- **date**: Date observations are recorded with data type DATE.
- **time**: Time observations are recorded with data type TIME.

The official weather observations follow similarly, except with additional time and date fields. The blueprint for these is described below.

- **ID**: This is the primary key in official observation tables. It must be not null, unsigned, and automatically increments for new entries. This has data type BIGINT and was designated as an index.
- **siteID**: This is the official weather station/forecase ID, and must be not null and unsigned. Additionally, this is the foreign key referring to `siteID` in table `stationProfile`. This has data type INT and was designated as an index.
- **obsDate**: Date observations are recorded with data type DATE.
- **obsTime**: Time observations are recorded with data type TIME.
- **issueDate**: Date the observations are issued, and has data type DATE.
- **issueTime**: Time the observations are issued, and has data type TIME.
- **<OBSERVATION[j]>**: There are 8 different observations made from official weather stations, `<OBSERVATION[j]>` can be one of: `pressureArrow`, `humidity`, `pressureStation`, `temp`, `visibility`, `windDirection`, `windSpeed` or `WX`. Each of these are described further in section 4.7, and hold a range of data types.

The final table type are the forecast tables. These contain two additional fields, forDate and forTime, and contain a larger number of tables than both citizen observation and official observations.

- **ID**: This is the primary key in official observation tables. It must be not null, unsigned, and automatically increments for new entries. This has data type BIGINT and was designated as an index.
- **siteID**: This is the official weather station ID, and must be not null and unsigned. Additionally, this is the foreign key referring to `siteID` in table `stationProfile`. This has data type INT and was designated as an index.
- **obsDate**: Date observations are recorded with data type DATE.
- **obsTime**: Time observations are recorded with data type TIME.
- **issueDate**: Date the observations are issued, and has data type DATE.
- **issueTime**:Time the observations are issued, and has data type TIME.
- **forDate**: Forecast date, and has data type DATE.
- **forTime**: Forecast time, and has data type TIME.
- **<OBSERVATION[k]>**: There are 11 different forecasts made from official weather stations over 7 days, `<OBSERVATION[k]>` can be one of: `feelTemp`, `humidity`, `precipitationProb`, `temp`, `UVRating`, `visibilityDistance`, `visibilityRating`, `windDirection`, `windGust`, `windSpeed` or `WX`. Each of these are described further in section 4.7, and hold a range of data types.

4.7 Observation Definitions

As mentioned in Section 4.6, there are many types of observations which may have different definitions across different tables. This section will describe the meaning of these fields, with special focus on those fields which have the same name, but have different meanings across the database. We begin by examining the different citizen observations.

- **dewTemp** is the temperature below which water droplets begin to condense, forming dew thanks to the air being saturated with water [30, 31]. This has data type DECIMAL(20,10), and was measured in degrees Celsius.
- **humidity**, or relative humidity, is the relative amount of gaseous water in the air, and can be an indicator of precipitation, fog or dew. It measures how far the air is from water saturation, which varies for different temperatures. For example, at higher temperatures the air can hold more water, if there

was the same amount of gaseous water in the air, but at a lower temperature, the humidity would be lower for this higher temperature. Humidity has data type INT, and is expressed as a percentage [32].

- **pressureSea** is the mean sea level pressure. It is common to convert air pressure to its mean sea level counterpart [33]. Pressure is measured in hectopascals, and has data type DECIMAL(20,10).
- **pressureStation** is the atmospheric pressure at station height. Pressure is measured in hectopascals, and has data type decimal(20,10).
- **rainfall** is the amount of rainfall in terms of mm/h. This has data type DECIMAL(20,10).
- **soilTemp** is the temperature of the soil beneath the weather station. Temperature is measured in degrees Celsius and has data type DECIMAL(20,10).
- **temp** is the temperature of the air surrounding each weather station. Temperature is measured in degrees Celsius and has data type DECIMAL(20,10).
- **windDir** is the measurement of a plane angle for wind direction, measured in arc degrees. This is stored as data type INT.
- **windSpeed** is the speed of the wind, measured in knots. This is stored as data type DECIMAL(20,10). Please note: for queries this was converted to mph for comparative analysis.

We are now going to examine official weather observations.

- **humidity** or relative humidity, is the relative amount of gaseous water in the air, and can be an indicator of precipitation, fog or dew. It measures how far the air is from water saturation, which varies for different temperatures. For example, at higher temperatures the air can hold more water, if there was the same amount of gaseous water in the air as a lower temperature, the humidity would be lower for higher temperatures. Humidity has data type INT, and is expressed as a percentage [32].
- **pressureArrow** states whether the pressure at station height is rising, falling or remaining the same, and has type VARCHAR(5).
- **pressureStation** is the atmospheric pressure at station height. Pressure is measured in hectopascals, and has data type **decimal(20,10)**.
- **temp** is the temperature of the air surrounding each weather station. Temperature is measured in degrees Celsius and has data type DECIMAL(20,10).
- **visibility** is a categorical variable and can take the following values: E = Excellent, VG = Very Good, G = Good, M = Moderate, P = Poor, VP = Very Poor.
- **windDirection** is the measurement of a plane angle for wind direction, and

gives the cardinal directions, for example S (south), NNE (north, north-east), etc. Its data type is VARCHAR(5).

- **windSpeed** is the average speed of the wind, measured in mph. This is stored as data type DECIMAL(20,10).
- **WX** is a general outlook on weather, and has data type VARCHAR(45).

And finally, we examine the forecast data. There are 11 forecasts tables, and their fields of interest are explained below.

- **feelTemp** describes the temperature that the human body may perceive the air temperature to be. This temperature is different from regular air temperature because humidity and gusts may make the air feel warmer or cooler than the actual air temperature [34].
- **humidity** or relative humidity, is the relative amount of gaseous water in the air, and can be an indicator of precipitation, fog or dew. It measures how far the air is from water saturation, which varies for different temperatures. For example, at higher temperatures the air can hold more water, if there was the same amount of gaseous water in the air as a lower temperature, the humidity would be lower for higher temperatures. Humidity has data type INT, and is expressed as a percentage [32].
- **precipitationProb** is the probability that there will be at least 1mm of rainfall in the given window. It is expressed as a percentage, and has data type INT.
- **UVRating**: the strength of UV light depends on the angle the sunlight hits the surface of the earth, the condition of the ozone layer, and cloud cover among other factors. UVRating gives us an idea of how strong UV radiation is in that particular area. It can take values ranging from 1-11, and is stored as data type INT [35, 36].
- **temp** is the temperature of the air surrounding each weather station. Temperature is measured in degrees Celsius and has data type INT.
- **visibilityDistance** is another example of information being hidden in HTML code which does not present itself in the browser. This is a measure of approximately how far into the distance can be clearly seen by eye. This distance is stored as data type INT.
- **visibilityRating** is what the browser displays, and it is a categorical version of the visibility distance. This is stored as VARCHAR(5).
- **windDirection** is the measurement of a plane angle for wind direction, and gives the cardinal directions, for example S (south), NNE (north, north-east), etc. Its data type is VARCHAR(5).
- **windGust** maximum wind speed is mph, and is stored as data type INT.

- **windSpeed** is the average speed of the wind, measured in mph. This is stored as data type INT.
- **WX** is a general outlook on weather, and has data type VARCHAR(45).

Although many of the observations/forecasts carry the same meaning across the tables, there are some discrepancies. Namely, wind direction measurements are categorical for both forecast and official data, whereas they are discrete for citizen data. It should be noted that wind speed measurements change their units between knots and mph, this can be converted within queries however. And finally, temp is stored as data type DECIMAL(20,10) for citizen and official data, however it is stored as an INT for forecast data.

4.8 ER diagram

As databases grow, it becomes increasingly difficult to keep track of all the tables, the relationships between them, and the nature of these relationships. MySQL Workbench provides the means to create simple Entity Relationship (ER) diagrams. ER diagrams graphically illustrate entities and their relationships between each other within a database [37, 38]. ER diagrams help the database designer to visualise their database structure, and implement improvements. The ER diagram for `weatherDB` is shown in Figure 1.

All the relationships in this database are of the same nature: many-to-one. Many records in `citizenProfile` can reference each singular record `stationProfile`, many citizen observation tables can reference each singular record in `citizenProfile`, and many forecast and official observations can reference each singular record `stationProfile`.

4.9 MySQL Review

MySQL is a satisfactory DBMS for the most part. It is easy to download and install, and there is huge support and documentation available. Another great feature is transactions, where a sequence of SQL are statements executed as a single unit or not at all. This feature was particularly useful for when a user deletes rows by accident, they can quickly cancel the query, and no data will be deleted. This can save the loss of potentially important information. However, there are a few problems. Firstly, importing data was more difficult that one might imagine. MySQL refused to import any of the weather data for a long time, with no explanation as to why. When it finally became clear that the problem was the date format, there was no way to make MySQL more flexible in terms of accepting different `DATE` formats. This meant the dates had to undergo a thorough cleaning process, or importing them as data type `VARCHAR()`. Importing dates as `VARCHAR()` would mean losing all the power of date functions within SQL, which become vital throughout chapter 5. Secondly, when creating the database schema in MySQL Workbench, there is a drop down box for selecting the data type of the fields.

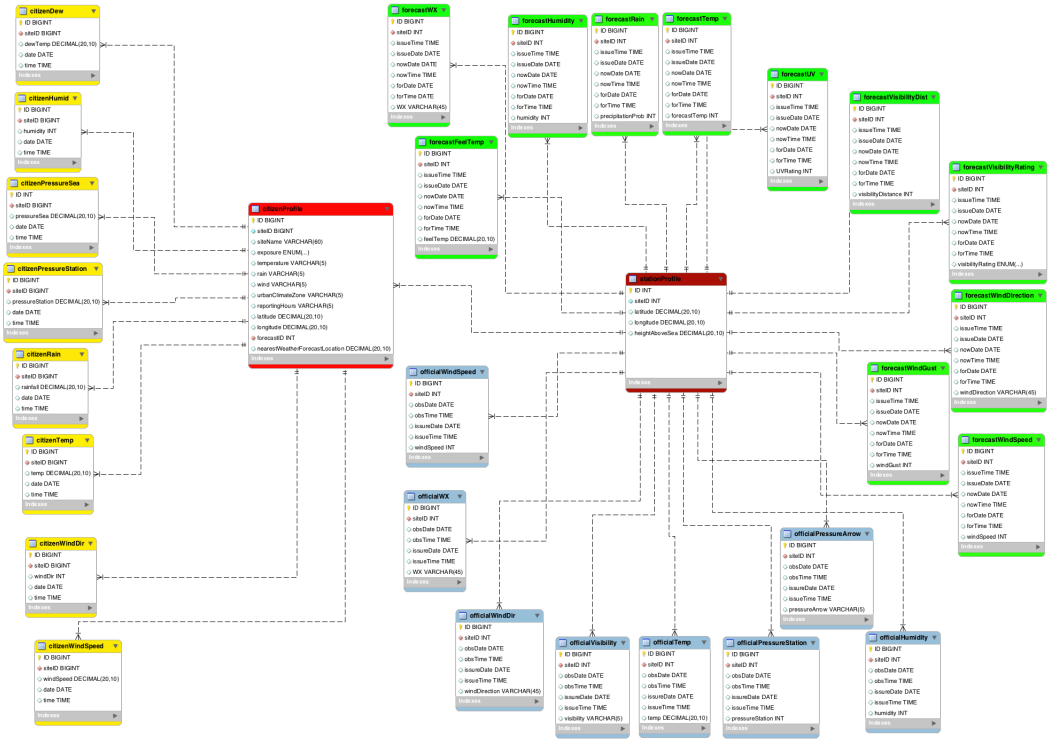


Figure 1: ER diagram for database `weatherDB`. Light blue coloured entities represent official weather observation tables, green represents forecast data, yellow represents citizen data, bright red represents the citizen station profiles, and maroon represents the official weather station profiles.

This can be misleading, as users may think that only the 5 data types in the drop box can be used, or even exist. Its only upon double clicking on the drop box that it allows users to manually type their desired data type.

5 Analysis

This chapter describes the analysis that was performed on the data, and the inferences which can be drawn from the results. We will begin by exploring the data, to try and discover and interpret any hidden patterns, distributions or obvious mistakes. Following this, we will adjust forecasts according to any patterns, and increase the granularity of weather forecasts. This amounts to part 5 - 7 of the generic data science project process, outlined in Section 1.3. Any necessary theoretical background material will first be reviewed, so that key concepts and terminology are understood. Note: when retrieving data through queries, the `DISTINCT` keyword was added so that repeated entries were not used for analysis, ensuring the validity of results. This section will assume total data veracity to begin with.

5.1 Background

5.1.1 Standard Deviation and Variance

Standard Deviation (SD) and variance are used to measure the dispersion of data. Standard Deviation is often denoted by σ, and the variance is often denoted by σ^2. High values of variance means the data are spread widely around the mean, and low values mean the data are not spread widely around the mean. It is defined as the expectation of the squared deviation of a random variable from its mean [39], and so it can only take non negative values. The variance can be calculated by:

$$\sigma^2 = \frac{\sum((x_i - \mu^2))}{n-1}.$$

Where x_i are observed values, μ is the mean, and n is the sample size. The standard deviation is the root of the variance. A low standard deviation indicates that data are not widely dispersed around the mean, whereas a large value indicates that the data are widely spread [40]. The standard deviation has the same units as the data it measures. The most common uses of standard deviations are normalising data, or constructing confidence intervals. A 95% confidence interval is found by:

$$CI = \bar{X} \pm 1.96 \times \frac{\sigma}{\sqrt{n}}.$$

Where $\bar{X}$ is the sample mean, 1.96 is the constant to specify the 95% level as the confidence interval, and n is the sample size.

5.1.2 Correlation

In statistics, correlation is any relationship between two variables, whether the relationship is caused directly by the other variable or not. The strength of this relationship can be measured by something called a correlation coefficient, the most common of which is Pearson's R, and is defined as follows:

$$R = \frac{1}{n-1} \sum_{x} \frac{(x_i - \bar{x})}{\sigma_x} \sum_{y} \frac{(y_i - \bar{y})}{\sigma_y}.$$

Pearson's R only measures linear relationships, however data can be transformed (assuming an appropriate transformation is known), before calculating R to get a measure of the relationship even when the relationship is non linear [41]. R can be negative, or positive, and ranges between -1 and 1. Where 1 would denote perfect positive correlation, 0 denotes no correlation, and -1 denotes strong negative correlation. Negative correlation means the two variables have an inverse relationship, and hence the data will produce a negative slope when plotted. It should be noted, that correlation does not necessarily imply causation, correlation can be caused in four different ways. Firstly, one variable, A, causes a change in another variable, B. Secondly, changes in B are caused by changes in A. Thirdly, no causal relationship at all, relationship is purely by chance. And finally, the changes in both A and B are actually caused by a third variable, C. This is known as a confounding variable [42].

5.1.3 P-values

A P-value is the chance of observing the data if the null hypothesis is true. It is commonly used in hypothesis tests to check for statistical significance. If the P-value is below a certain value (usually 0.05, although this can be changed depending on research needs), then we reject the null hypothesis. For example if we want to measure the differences between groups of people, then our null hypothesis is that all the people within that group have the same height. If the P-value is below 0.05 (assume a 95% significance test), then we reject the null hypothesis. Depending on the data at hand, we may wish to use different levels of significance. The most common is 95% , however, when we want to be more sure of rejecting the null hypothesis, we can set the required significance level to be 99% (P-values less than 0.01 mean null hypothesis is rejected), or even higher. However, we must be careful when using P-values for a number of reasons.

1. Just because the null hypothesis has been rejected does not necessarily mean that the alternative hypothesis is true, just that the observed data is inconsistent with the null hypothesis [43].
2. The statistical significance level is arbitrarily chosen, there is no rigorous mathematical reasoning for the confidence levels set.

3. When we have a large data set, then we are likely to reject the null hypothesis, even if it is not very different from the null hypothesis. This is called *Meehl's conjecture* [44, 45], and can result in correct null hypotheses being rejected.

5.1.4 Outliers

An outlier in a data set is an observation (or set of observations) which appear to be inconsistent with the data set [46]. We must always detect outliers, but whether we remove them or not is a statistical decision, based on a number of features. Outliers aren't necessarily errors, sometimes the outliers are very important for analysis, such as in fraud detection.

When deciding whether to remove outliers, we must consider a number of factors. Firstly, if the outlier appears to be within the realms of possibility, we must consider how much influence the point has on the data set. Informally, if a single outlier is not too far removed from the rest of the points, or there is a very large number of points, then it is unlikely to be highly influential on the data in terms of summary statistics or plots. If it is highly influential, then we must carefully consider whether we believe it to be a mistake, or just an unusual value. If a single outlier is not very influential, leaving or moving it will have little effect. Secondly, we must discover whether the outlier is an obvious inconsistency, i.e. a value which could not possibly represent a real world value for the subject that the data pertains. If this is found, the outlier should be removed in the majority of cases. Finally, if the outlier seems plausible and we have no reason to remove it, then it should be kept in the data set for both plotting and analysis.

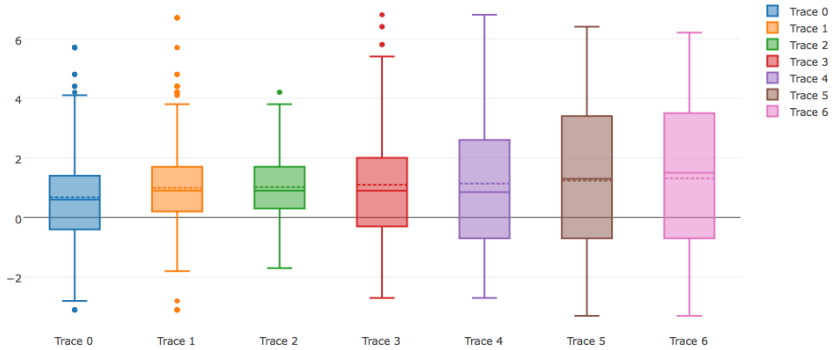


Figure 2: Box plots example. Difference in observed temperature and forecast temperature at different forecast dates for a particular site.

5.1.5 Box Plots

There are many techniques for detecting outliers, but perhaps the most simple is the box plot. Figure 2 depicts 7 different box plots. The "box" in a box plot is the quadrilateral between the lines at either end, or *whiskers*. Somewhere in the middle of this box there is a dotted line and a solid line. The dotted line represents the mean, and the solid line represents the median. The top and bottom edges of this box represent the upper (0.75) and lower (0.25) quartiles. The larger this box is, the greater the variability present among the data. The tips of the whiskers represent $1.5 \times IQR$, where the IQR is the Inter-Quartile Range, and is equal to the difference between the lower and upper quartile [47]. Observations which fall outside of these whiskers are defined as outliers, and each outlying point is plotted above or below the whiskers accordingly [48]. Figure 2 exhibits outliers in `trace0` and `trace1` on both sides of the whiskers, whilst `trace2` and `trace3` display outliers above the upper whisker. None of these outliers are particularly extreme, but they should be investigated regardless.

5.1.6 Regression

Regression is a supervised learning technique which is used to learn the causal relationship between one variable and another when the data is continuous. The data in this project is a supervised learning problem because we know the input, and the desired output. As temperature is not a categorical data type, regression is the correct learning algorithm. We have a dependent variable, Y, and one or more independent variables, $\mathbf{X} = (X_1 + X_2 + ... + X_k)$. The dependent variable is the target, and the dependent variables are the explanatory variables [50], and we can express their relationship as follows:

$$Y \approx f(\mathbf{X}, \boldsymbol{\beta}). \tag{1}$$

Where $\boldsymbol{\beta}$ are the unknown parameters. In order to run a regression, the number of data points, N, must be greater than or equal to the number of features, k. We also want to find a solution for the unknown parameters, $\boldsymbol{\beta}$, which minimises the differences between the measured and predicted values of $\mathbf{Y}$. There are many options for f, but when the dependent variable can be expressed as a linear sum of the independent variables, it is called *linear regression*. The following are assumptions made when performing a linear regression.

1. **Linearity:** This is the assumption that the dependant variable is related to the independent variables through some linear combination of the independent variables and the parameters.
2. **Homoscedasticity:** This means that all the random variables have the same finite variance.

3. **Independence of errors:** We want the errors to be Independent and Identically Distributed (IID). We can check this using a Q-Q plot, or through other methods.
4. **Minimal multi-colinearity:** Multi-colinearity is when the independent variables are not independent from each other. We want to avoid this as much as possible.
5. **No auto correlation within the data:** Auto correlation occurs when the residuals are not independent from each other.

Although we make these assumptions, many of them do not hold true in real life examples [51]. We call $\mathbf{X}$ the *design matrix*, which has a row of 1s so that it includes intercepts when multiplied out. Let's consider an example where we have n target variables, and k features. The design matrix, dependent variables, and explanatory variables have the following format:

$$\mathbf{X} = \begin{bmatrix} 1 & x_{11} & x_{12} & x_{13} & \dots & x_{1k} \\ 1 & x_{21} & x_{22} & x_{23} & \dots & x_{2k} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & x_{n2} & x_{n3} & \dots & x_{nk} \end{bmatrix}$$

$$\mathbf{Y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_k \end{bmatrix}.$$

With linear regression, the relationship is expressed in matrix form as:

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta}.$$

Or if there is a singular dependent variable:

$$y = \beta_0 + \beta_1 x_1 + ... + \beta_n x_n$$

If we only use one feature in linear regression, it is known as *simple linear regress*, but if we use more than one feature it is known as multiple linear regression.

Armed with proper notation, we may proceed to describe how regression coefficients are actually found. In order to determine $\boldsymbol{\beta}$, we need to introduce a new function, called the *cost function*. The cost function is the cost of making the output, given the

input. Intuitively, we want to minimise our costs. There are many candidates which may be used as the cost function, but perhaps the most common of these is the Residual Sum of Squares (RSS). The residual sum of squares provides a means to measure the discrepancy between the data and an estimation model [52]. In a model with a single explanatory variable, the RSS is found by:

$$RSS = \sum_{i}^{n} (y_i - f(x_i))^2.$$

Whichever values of $\boldsymbol{\beta}$ minimise the cost function are then used as the explanatory variables in the model. Once we have built our linear regression formula, we proceed to numerically express how well the regression line fits the data. There are a number of ways to do this, but the simplest approach is to construct R^2. To construct R^2, we must first define the Total Sum of Squares (SST), where:

$$SST = \sum_{i}^{n} (y_i - \bar{y})^2.$$

And hence R^2 is:

$$R^2 = 1 - (RSS/SST).$$

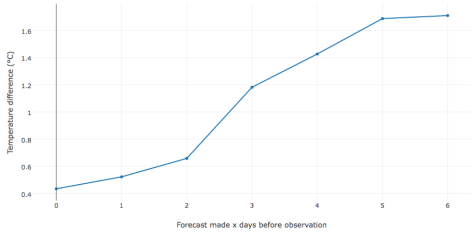
The last measurement we are going to introduce is studentized residuals. They are found by dividing a residual by an estimate of the standard deviation. Residuals are the difference between the observe values, and the predicted values. Without studentization, the standard deviations of all the residuals typically vary a great deal from residual to residual, hence it does not make sense to compare residuals without first transforming them in this way. Dividing by the standard deviation "normalises" the data, meaning they can be directly compared on the same scale. This makes studentized residuals a powerful tool in outlier detection. Formally, any studentized residual greater that $|3|$ is considered an outlier.

5.2 Data Exploration

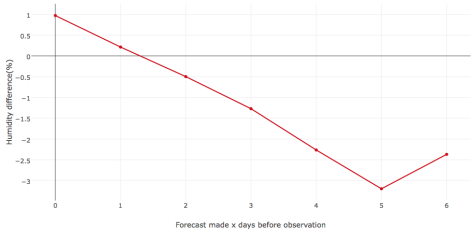
Exploratory Data Analysis (EDA) is a fundamental part of data analysis and understanding. It involves the use of graphics to understand data structure and distributions. Quantitative methods are not wrong per say, but they are incomplete. Without EDA we may forfeit insight into one or more aspects of the underlying structure of the data. This section will combine a mixture of quantitative and EDA methods.

Of the 8 observation types made by citizen sites, four can be directly compared, whilst one other can be compared in a more indirect fashion. The first two data sets being

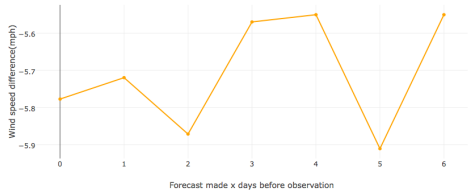
compared are the forecast data and the citizen data. Lets first examine how our predictions of the four directly comparable observations (temperature, humidity, wind speed, and wind direction) differ from forecasts made a range of days (0-6) before observation date. This is shown in Figure 3c.



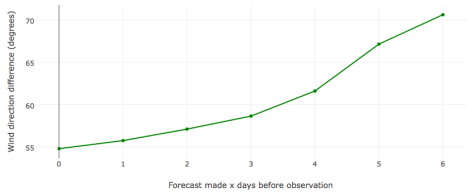
(a) Mean temperature difference.



(b) Mean humidity difference.



(c) Mean wind speed difference.



(d) Mean wind difrection difference.

Figure 3: The average differences in directly comparable observations.

Each of these plots shows the average difference between the different types of observation. Forecast times (i.e. the time in the future that the forecast predicts) are all within half an hour either side of the citizen weather station observations. There were 2,810,651 such temperature difference observations, 2,808,558 for humidity, 2,797,786 for wind speed, and finally, 2,709,855 for wind direction differences before the `distinct` clause weeded out any duplicates. Although there are multiple forecasts in a given day, they have been grouped together to give us an idea of general trends.

Before we examine Figure 3 in great detail, there are a few things to note. Firstly, wind speed was recorded in knots for citizen data and mph for forecast data, appropriate conversions have been made within the query. Wind direction was predicted as the cardinal directions for forecast data (e.g. N, SSE etc) and degrees for citizen data. Within the query, a "CASE" statement was used to convert North (N) to 0°, SE to 135°etc. A meaningful numerical conversion can now be made. Lastly, wind direction ranges from 0 to 360, meaning that although 355°is right next to 3°, the difference between these two would result in a larger number than the true difference. The following CASE statement in Listing 3 corrected this. Now the aforementioned example provides the correct answer of 8°, rather than 352°.

Listing 3: CASE statement in wind speed query resolving difference issue.

```
SELECT VARIANCE(windDirectionDeff.diff),
STD(windDirectionDeff.diff), MAX(windDirectionDeff.diff),
MIN(windDirectionDeff.diff), AVG(windDirectionDeff.diff),
count(*), CEIL(COUNT(*)/2) AS median
FROM (
  SELECT CASE WHEN directionDeff.diff > 180
  THEN ABS(360 - directionDeff.diff)
    WHEN directionDeff.diff < -180
    THEN  ABS(360 + directionDeff.diff)
    ELSE  ABS(directionDeff.diff) END AS diff
    FROM(
      ...
```

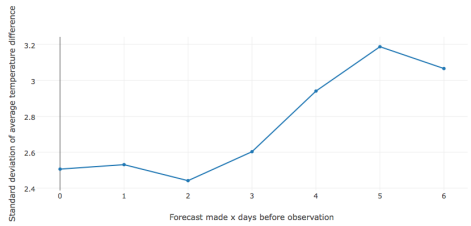
Where `directionDeff.diff` is the wind speed difference in degrees. Now that we are armed with more information, we can examine Figure 3. Figure 3a shows the temperature difference between citizen sites and forecast temperature. These temperature differences are all positive, meaning the forecasts are consistently underestimating the observed temperature (all differences are $citizenobservation - forecast$, meaning positive differences mean the forecast is underestimating the observation). This characteristic is exaggerated as the time to observation date increases. The plot is approximately linear, with some curving at the end points.

Figure 3b shows the average humidity difference in terms of percentages. The difference is always very small, even 6 days out (absolute value of approximately 3%). However, a clear downward trend is present where the forecast humidity is overestimating the observed humidity. This is linear for the most part, with exception for the 6 day forecast.

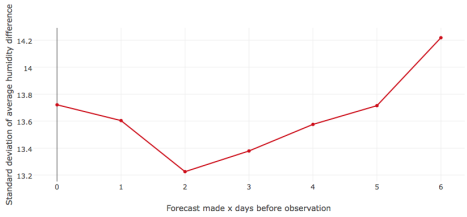
Figure 3c shows the average difference in wind speeds. We can see that wind forecasts don't really improve in terms of accuracy, and stay around the same value, which is approximately 5.6mph above the observed value.

Figure 3d shows the average difference in wind direction. The wind direction forecast was perhaps the weakest of the four in terms of error. Although we expect it to have a baseline error due to unspecific forecasts (compass divided into 16 rather than 360). However, this number is still high, with a minimum average difference of 55°. Again, there is a clear pattern whereby forecasts closer to the date provide more accurate average predictions, and this follows a slight curve. Some trends are emerging, and we have already spotted two places where there may be scope to improve forecasts: Figure 3a and Figure 3c. However, these plots do not provide a complete picture. Each individual site may tell a very different story, providing a wide range of values, creating scope for increased granularity. First we will explore the data further in Figure 4, which examines the standard deviations plotted against forecast time.

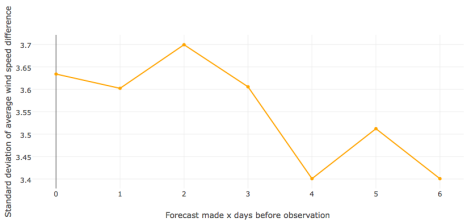
In Figure 4a there is a generally increasing trend as the days before observation date increases, however, this increase is small. Similarly for Figure 4b and d. Figure 4c has



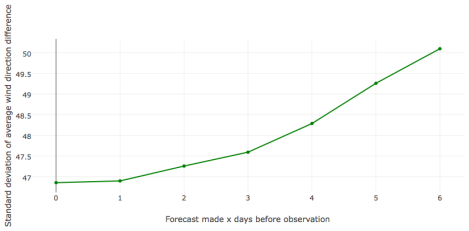
(a) SD of temperature difference.



(b) SD of humidity difference.



(c) SD of wind speed difference.



(d) SD of wind direction difference.

Figure 4: The standard deviations of differences in directly comparable observations.

a generally decreasing trend. This is somewhat surprising that forecasts made earlier vary less that those which have been made closer to observation time. The observed differences in measurements are evident, but these values do not vary a great deal.

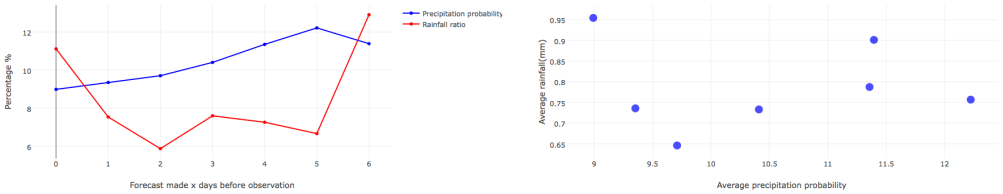
So far we have only explored the data which can be directly compared, however, there is another observation type which can be compared, albeit less directly. Citizen stations record rainfall amount within the last hour, whereas forecasts only provide a probability of precipitation. To compare these, we took the days when the probability of precipitation was non zero (when zero, it very rarely rained), and counted the number of hours where the recorded rainfall is at least 1mm (recall that the definition of precipitation probability is the probability than there will be at least 1mm of rainfall in the given window), and when there was less that 1mm. This data is displayed in Table 1.

Table 1: Rainfall data.

Days from obs	No rain	Rain present	Average rain	Average PP	Rain ratio (%)
0	409881	45576	0.95401	8.9916	11.1193
1	739986	55779	0.73565	9.3519	7.53784
2	279268	16401	0.64595	9.7088	5.87285
3	260184	19773	0.73298	10.4091	7.59962
4	258156	18748	0.78719	11.3562	7.26227
5	248113	16541	0.75666	12.222	6.66672
6	224132	28950	0.90092	11.3937	12.9164

Where `No rain` is the number of hours when there was no rainfall, `Rain present` is the number of hours when rainfall was recorded, `Average rain` is the average

recorded rain in mm, `Average PP` is the average precipitation probability expressed as a percentage, and finally `Rain ratio` is the `Rain present:No rain` ratio. Upon inspection of Table 1, we can see that the precipitation probability is, in general, a little higher than the observed rain. This is depicted by Figure 5a. Precipitation probability only drops below the rainfall ratio at the end points. Figure 5b shows the average precipitation probability plotted against the average rain. Surprisingly, there does not appear to be any trend. It seems intuitive that an increased likelihood of rain would result in higher rainfall, however it is not the case. In fact, the correlation is just -0.0348349.



(a) Precipitation probability and rain ratio Vs days before observation.

(b) Precipitation probability and rain ratio Vs days before observation.

Figure 5: Average rainfall Vs Average precipitation probability.

Weather observations are often related. For example, we would expect that humidity and rainfall are related, and hence a non-zero correlation co-efficient. Figure 6 examines how the different citizen weather observations are related to each other in terms of correlation co-efficients. We will not compare all observations, only those which can be sensibly compared. For example, it would not be sensible to compare wind direction and rainfall.

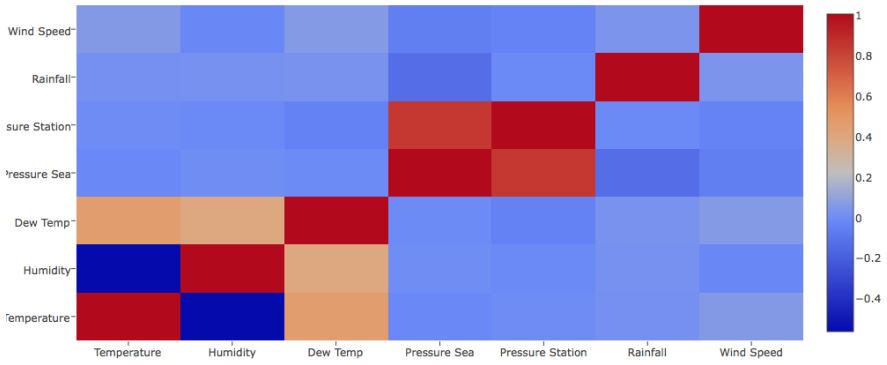


Figure 6: Correlation matrix.

There are some observations which are strongly correlated that we might have expected, such as temperature and dew temperature, and pressure at station height and pressure

at sea level. However, there are some correlations which are not as strong as we might have expected. Humidity and rainfall produced a low correlation coefficient with a value of 0.0138. It has an accompanying P-value of less than 0.00001, which is highly significant, however, this is likely a case of Meehl's conjecture, aforementioned in Section 4.1.3.

Temperature and pressure also have very low correlation coefficients, suggesting no relationship. This, at first, might seem surprising, particularly when we consider the ideal gas law, which includes temperature as a component of the formula which reads:

$$P = \frac{nRT}{V}.$$

Where P is the pressure, n is the number of moles of the gas, V is the volume of the gas in cubic metres, R is the ideal gas constant, T is the temperature of the gas in Kelvin. Again the P-value is less that 0.00001, suggesting a non-zero correlation coefficient, however this is likely to be another case of Meehl's conjecture, as there were 119298 relevant data points at the time of writing. In fact, due to the size of the data, it is likely that all P-values are likely to be misleading, as they were all significant at the 5% level bar one, when most coefficients were approximately zero. It is probable that the volume of the atmospheric air dominates the ideal gas equation, and small local changes in temperature are unlikely to have a significant impact on pressure.

We can also see that humidity is negatively correlated with temperature, with a coefficient of -0.571. This is because as temperature rises, the air can hold more water, and is further from its saturation point. Table 2 lists all correlation coefficients.

5.3 Temperature

In this section, we are going to take a closer look at temperature, which as mentioned in Section 5.2, is an observation type with scope for forecast optimisation and increased forecast granularity. The first thing to notice, is that the average difference between observed temperature at citizen sites and forecast temperature is 0.9231. However, this effect is exaggerated with forecasts further away from observation date, as shown in Figure 7. The next step was to look summary measures from each of these citizen stations individually. Figure 7 shows a sample of box plots from individual stations.

The vast majority of average temperature differences look similar to Figure 7a, most of what is left bears a closer resemblance to Figure 7b. What both plots have in common is that the further from the observation date, the more the forecasts underestimate, or the less the forecasts overestimate the observed weather, and the higher the variability tends to be. They differ in that forecasts like those in Figure 7a observe higher temperatures than forecast, where as the forecasts in Figure 7b tend to overestimate the temperature. It should be noted that the forecast is underestimating the temperature much more frequently than it is over estimating it.

Table 2: Correlation co-efficients.

Relationship	Correlation co-efficient	Number of data points
Temperature:Dew Temperature	0.4539434306231616	220296
Temperature: Pressure Sea	-0.0221775411627919	119009
Temperature:Humidity	-0.5710995167485311	222679
Temperature:Pressure (station)	-0.0099972255082910	103531
Temperature:Rainfall	0.0093485405954952	207551
Temperature:Wind Speed	0.0507805392363314	220055
Humidity:Dew	0.3852610803485919	220835
Humidity:Pressure (sea)	-0.0041663176694097	119298
Humidity:Pressure (station)	-0.0154181867460822	103402
Humidity:Rainfall	0.0138228072072678	207372
Humidity:Wind speed	-0.0266292885568611	219324
Dew temp:Pressure (sea)	-0.0123297081281974	118470
Dew temp:Pressure (station)	-0.0516475047186224	101872
Dew temp:Rainfall	0.0183242738507239	205355
Dew temp:Wind speed	0.0550603878568199	216970
Pressure (sea):Pressure (station)	0.8411169986825691	220
Pressure(sea):Rainfall	-0.1364204128259863	107531
Pressure(sea):Wind speed	-0.0654102223097972	117311
Pressure (station):Rainfall	-0.0150276525814894	100132
Pressure (station):Wind speed	-0.0468028908828401	102149
Rainfall:Wind speed	0.0299185219416078	206870

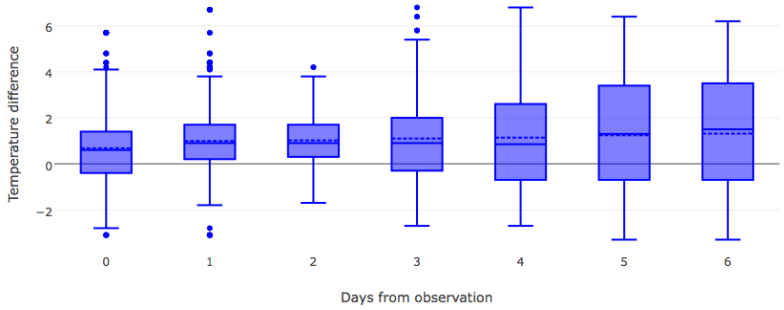
The difference between average citizen observed temperature and forecast temperature could be caused by the following:

1. Temperatures are genuinely being underestimated by forecasts, which can be checked by finding the differences between forecasts and official observation data. If the official observations are finding a similar pattern, then the patterns found from the citizen data seem more credible.

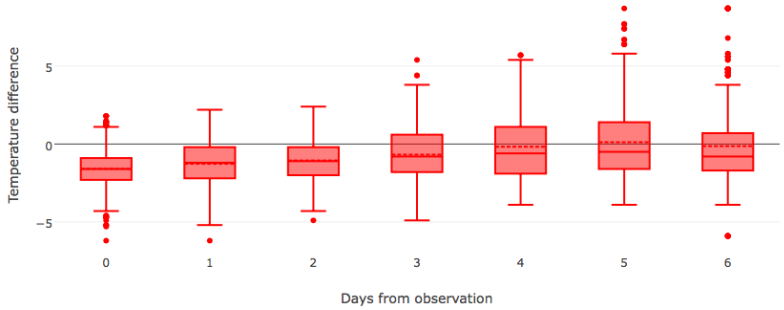
2. The citizen weather stations are making less accurate recordings , and this is the cause of differences between forecasts and citizen observations. This can also be checked by comparing forecast and official observations.

In either case, examining the relationship between the average temperature difference between citizen and forecast data, and official observational data and forecast data, is the next logical step. This is plotted in Figure 8.

Both the official data and citizen data average differences in temperature from the forecast data follow an almost identical trend, however, citizen data differences are just a little higher for each day. The citizen observations are higher by an average difference of 0.289°C on a given day. This is a desirable result because it shows two things: firstly, the citizen temperature differences are sufficiently close to official differences, and as-



(a) Box plots showing average temperature difference for station ID: 117944.



(b) Box plots showing average temperature difference for station ID: 897886001.

Figure 7: Sample box plots for average temperature difference on individual citizen stations.

suming the official observations can be trusted, this means the citizen temperature observations are more credible. Secondly, the official data follows the same trend in that forecasts are underestimating the temperature. This trend exaggerated with forecasts made further from the observation date. It should be noted that these calculations were made using 877,229 official temperature differences, and 2,810,651 citizen temperature differences.

Next, we are going to check the standard deviations of average temperature differences for both citizen and official data. This is shown in Figure 9.

Both sets of standard deviations follow a similar pattern, except the standard deviations for citizen data are considerably higher. The effect gets exaggerated closer to the observation date.

It can sometimes be insightful to plot the standard deviations of each station on a line plot to see if anything interesting presents itself. There are too many stations to plot

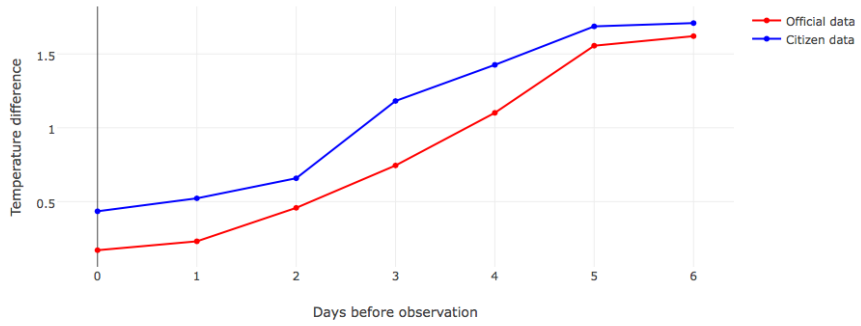


Figure 8: Official temperature differences and citizen data temperature differences Vs days before observation.

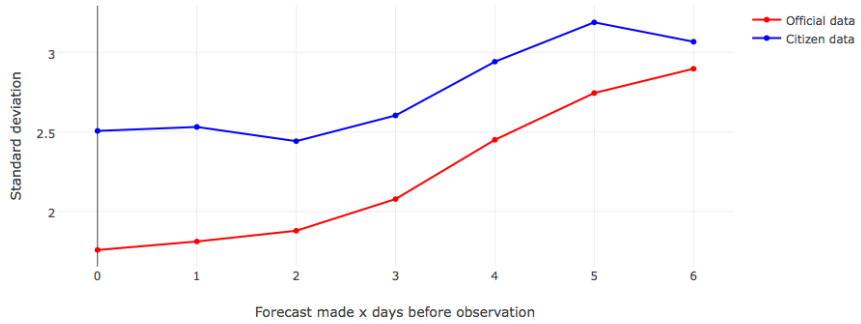


Figure 9: Standard deviations of official temperature differences and citizen data temperature differences Vs days before observation.

them all, however we can get a good idea of any patterns from plotting a sample of 20. Figure 10 is an illustration of this.

Figure 10 shows that most individual stations follow a similar pattern. There is a general upward trend as forecasts move further from the observation date, with a spike 1 day before the observation and a slight dip 6 days from the observation date. The standard deviations lines are more spread out closer to the observation date, and grouped much more closely further from the observation date due to a small number of observations exhibiting a softer decline.

5.4 Temperature Bands

Each citizen weather station has a list of ratings, including a temperature rating. It may be insightful to view the average temperature difference trend at each rating. Hereafter,

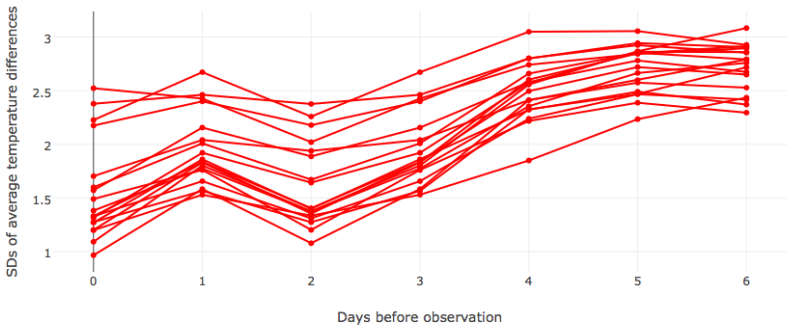
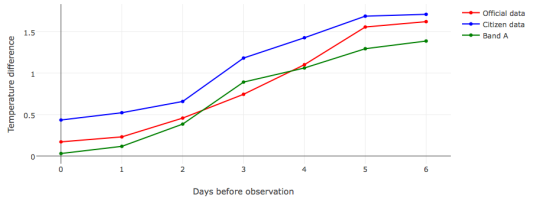
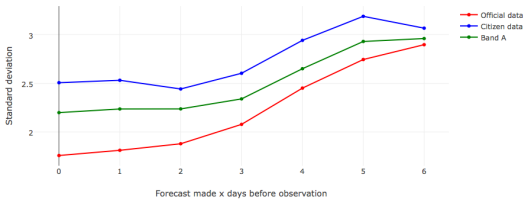


Figure 10: SD of a sample of individual citizen weather stations average temperature difference.

we will refer to temperature measurements which have a rating of 'A', for example, as *temperature band A*, or *band A* (more information on citizen site ratings at `http://wow.metoffice.gov.uk/support/siteratings#locationAttributesUCZ`). Figures 11-15 show the mean temperature difference and standard deviations of the mean difference against days before observation date for each weather band, along with the official and citizen temperature counterparts for comparison.



(a) Mean temperature difference Vs forecast length.

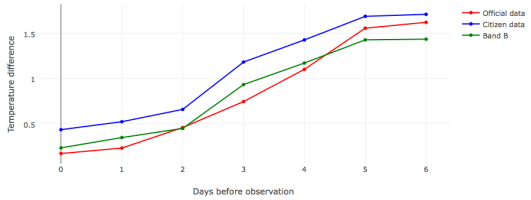


(b) Standard deviations Vs forecast length.

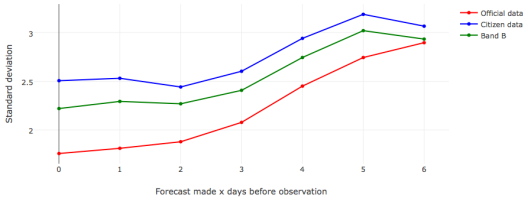
Figure 11: Band A mean and standard deviations for temperature differences Vs forecast length, along with their official and citizen counterparts.

Figure 11 shows the mean difference and standard deviations for band A. There are 146 weather stations that have a band A temperature rating, with 369,591 respective observations. This line is a much closer fit to the official observation data than the citizen data. Band A is closer to zero than official observations less that 2 days to observation, meaning the forecasts are more accurate closer to observation date. However, it slightly tails off as days to observation increase. Standard deviations are right in the middle between citizen data and official observation data. The maximum temperature difference is 15.8°C and the minimum is -13.9°C. Both the maximum and minimum observation seem too high to be inaccurate forecasts, and they are more likely to be observation errors.

Figure 12 shows the mean difference and standard deviations for band B. There are 252 citizen weather stations that have a band B temperature rating, with 807,997 respective observations. The mean temperature differences for band B have an almost identical shape to the citizen data whilst having a much better fit to the official observation data. It does, however, dip slightly at 6 days before the observation. The maximum temperature difference was 17.5°C, and the minimum was -17°C. Notice that the maximum and minimum have increased in absolute value, suggesting the that error size is increasing. Standard deviations also take a very similar shape to the citizen data, and although band B standard deviations are still between citizen and official observation standard deviation lines, it is now slightly higher, edging closer to the citizen standard deviation line.

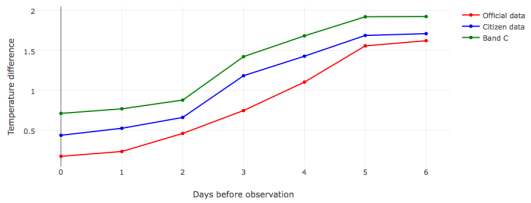


(a) Mean temperature difference Vs forecast length.

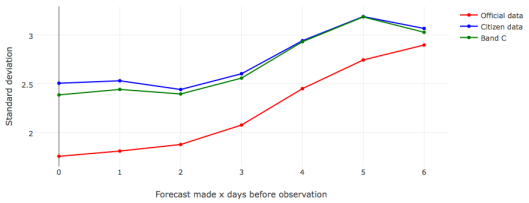


(b) Standard deviations Vs forecast length.

Figure 12: Band B mean and standard deviations for temperature differences Vs forecast length, along with their official and citizen counterparts.



(a) Mean temperature difference Vs forecast length.



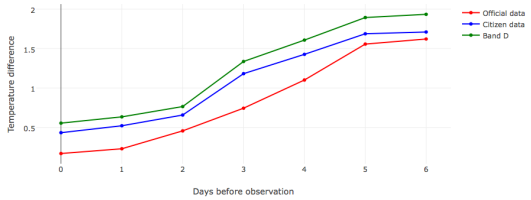
(b) Standard deviations Vs forecast length.

Figure 13: Band C mean and standard deviations for temperature differences Vs forecast length, along with their official and citizen counterparts.

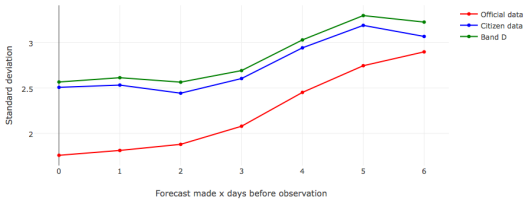
Figure 13 shows the mean difference and standard deviations for band C. There are 206 citizen stations that have a band C temperature rating, with 652,299 respective observations. Again, band C almost exactly emulates the citizen data mean difference, except this time it is higher than the citizen data. The maximum temperature difference was 19.7°C, and the minimum was -24°C. The absolute values of both maximum and minimum temperature difference have increased even further from band B, which is a telling

characteristic of lower quality recording apparatus. Standard deviations approximately follow the same trend for citizen data and band C.

Figure 14 shows the mean difference and standard deviations for band D. There are 219 citizen stations that have a band D temperature rating, with 686,166 respective observations. The mean differences fit fairly closely with citizen data, with band D being just a little higher. Band D has a slightly lower mean difference than band C. The maximum temperature difference was 23.3°C, and the minimum was -26.2°C. This is a slight increase in absolute value from band C, further demonstrating that lower quality recording equipment increases error size. The standard deviations of band D temperature differences approximately fit with the citizen data observation, except are a little higher.



(a) Mean temperature difference Vs forecast length.

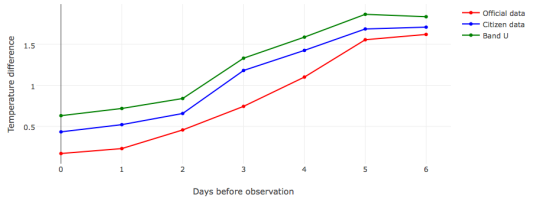


(b) Standard deviations Vs forecast length.

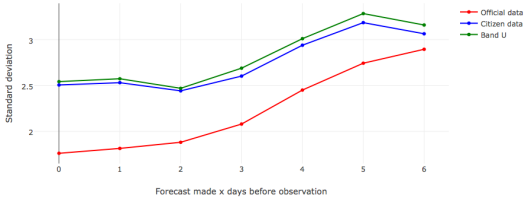
Figure 14: Band D mean and standard deviations for temperature differences Vs forecast length, along with their official and citizen counterparts.

Figure 15 shows the mean difference and standard deviations for band U. There are 118 citizen stations that have a band U temperature rating, with 298,342 respective observations. Band U looks very similar to both band D in Figure 14, and band C in Figure 13. The mean difference line is just above the citizen data line, suggesting that band U is likely mainly composed of band C and D temperature ratings. The maximum temperature difference was 18.9°C, and the minimum was -39.9°C. Although the maximum temperature difference has decreased, the absolute value of the minimum has drastically decreased. Differences this low are clearly recording errors. The standard deviation line approximately fits to the citizen data standard deviation line.

Two clear trends emerge. As the temperature rating declines, standard deviations rise and average temperature differences rise. Bands A and B almost replicate the official observed temperature. From this, we infer that citizen stations with lower temperature grades are over estimating the temperature, and have higher variability. There are two reasons why this may be. Firstly, stations with lower temperature ratings have less sophisticated temperature recording apparatus. This less sophisticated apparatus may not measure temperature as accurately. This results in higher variability temperature differences, and hence standard deviations. Secondly, a station's temperature rating is not independent of other site ratings within the same station. Sites with low temperature ratings generally have lower ratings for other attributes, and this can lead to incorrect



(a) Mean temperature difference Vs forecast length.



(b) Standard deviations Vs forecast length.

Figure 15: Band U mean and standard deviations for temperature differences Vs forecast length, along with their official and citizen data counterparts.

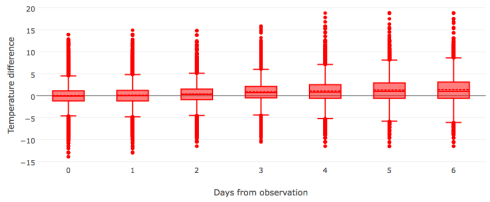
observations. For example, sites in built up areas will record higher air temperatures than rural areas. Citizen stations with lower temperature ratings are more likely to be in built up areas. Hence, site ratings, other than temperature, can be responsible for recording higher than expected temperatures.

Next we will examine the box plots for each temperature rating to see the variability difference for each band, and to check if one or more bands contain significantly more outliers than the other bands, and if so, are these outliers more in the positive or negative direction. Due to the higher standard deviations in lower ranked bands, we expect these to have broader whiskers and more outliers. These are shown in Figure 16.

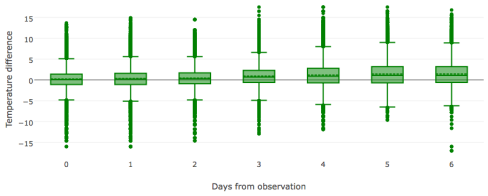
Figure 16a appears to have approximately equal outliers evenly spaced either side of the whiskers, possibly more above the upper whisker. Similarly with Figure 16b. However, Figure 16c and Figure 16e show a hugely disproportionate number of outliers outside the upper whisker. The outliers are also much more irregularly spaced. This trend can also be observed in Figure 16d, but to a lesser extent. This disproportionate number of outliers above the upper whiskers compared to the lower whiskers could be responsible for citizen sites overestimating the temperature.

In Figure 16c, d, and e there are clusters of observations below the lower whisker which have very large absolute temperature differences. These values are obvious inconsistencies. They could be clustered together because a single station recording the temperature incorrectly, and so all the temperature differences will bunch up around that value. Figure 16 exhibits the same trends we have observed thus far: the mean temperature difference and variation are increasing as days before observation increase. Please note the different scales.

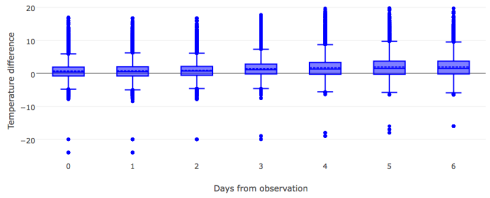
To demonstrate the disproportionate distributions of outleirs numerically, the outliers that were above the upper whisker, and below the lower whisker, along with average values of outliers from either side are displayed in Tables 3 through 7. If there is a disproportionate ratio of outliers above the upper whisker, than below the bottom whisker, then it may be the case that these disproportionately distributed outliers are responsible for citizen data over estimating the mean temperature difference. The column *Above*



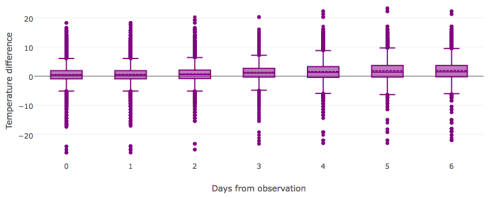
(a) Band A mean difference box plots.



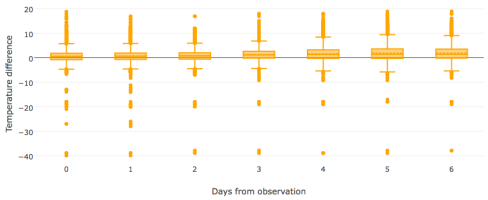
(b) Band B mean difference box plots.



(c) Band C mean difference box plots.



(d) Band D mean difference box plots.



(e) Band U mean difference box plots.

Figure 16: Box plots for mean temperature difference over the seven days for each temperature rating.

average is the mean value of the outliers that are above the upper whisker, similarly for *Below average*.

Table 3 shows that Band A outliers are the most evenly distributed, with approximately double the number of outliers above the upper whisker than below the bottom whisker. However, the Above count:Below count ratio generally increases as Days before observation increases. Also note that the absolute value of both the above and below average follow a similar trend.

Band B outliers follow a similar pattern in Table 4, except this effect is more exaggerated. The absolute values of the above and below averages, along with the above:below average ratio increases as days from observation increase.

The same trend can be observed for band C, D, and U, except this effect gets more and more amplified in lower band temperature ratings. This agrees with our conjecture that these outliers may be responsible for citizen weather stations overestimating the mean temperature difference.

Table 3: Band A outlier statistics, where "Above" means outliers that reside above the upper whisker, and "Below" means outliers which reside below the lower whisker.

Days from observation	Above count	Above average	Below count	Below average
0	1913	6.33857815	1168	-6.785787671
1	3259	6.64473764	1715	-6.742332361
2	1077	6.99090065	704	-6.340340909
3	970	7.828350515	545	-6.294311927
4	885	8.949378531	383	-7.009138381
5	924	9.780735931	247	-7.572064777
6	724	10.17803867	199	-7.798994975

Table 4: Band B outlier statistics, where "Above" and "Below" are defined as above.

Days from observation	Above count	Above average	Below count	Below average
0	3432	6.626515152	1451	-6.910751206
1	5783	7.260038043	2005	-7.153017456
2	2142	7.208496732	775	-6.596903226
3	1770	8.146327684	418	-6.932296651
4	1402	9.679957204	155	-7.712903226
5	1405	10.65295374	51	-7.864705882
6	1153	10.31786644	87	-8.228735632

The box plots in Figure 16 have been very telling. They illustrate how lower band temperature ratings have higher variability, overestimate temperatures, and that this phenomena may be due to disproportionately distributed outliers. After inspecting the numerical data in our tables, we can confirm that outliers are disproportionately distributed, which could be the root cause of citizen data over estimating temperature. To verify our conjecture, we are going to re-plot Figures 11 through 15, excluding the outliers from each query. All these outliers are not necessarily measurement errors, however, we suspect the vast majority are. Due to the size of the data, and hence a large number of outliers, it is unrealistic to inspect each outlier individually, and so removing all of the outliers may be the best compromise. This may result in lower standard deviations than that which should be expected. This phenomena of removing all outliers will be referred to as *aggressive outlier removal*. In each figure, we have kept the same "Citizen data" and "Official data" from before to use as a reference guide. Recall that "Citizen data" refers to the citizen data with no outliers removed.

Band A mean temperature observations in Figure 17 map very closely to the observational data, as they did before in Figure 11, however, they now tail off more aggressively as days to observation increases. Removing the outliers from band A temperature rating has had little effect on the mean temperature difference, as we might have expected upon inspection of Table 3. The standard deviations have changed a good deal more. Although the standard deviations map much closer to the standard deviations of the official data, they fall slightly below for each day. This is likely an effect of aggressive

Table 5: Band C outlier statistics, where "Above" and "Below" are defined as above.

Days from observation	Above count	Above average	Below count	Below average
0	3310	7.726102719	568	-6.063556338
1	5678	8.181102501	771	-6.275875486
2	2143	8.057489501	379	-5.903957784
3	1730	9.369768786	159	-6.302515723
4	1413	11.0338995	30	-9.633333333
5	1338	12.10112108	16	-13.225
6	1022	11.45362035	25	-10.22

Table 6: Band D outlier statistics, where "Above" and "Below" are defined as above.

Days from observation	Above count	Above average	Below count	Below average
0	2999	8.171257086	1126	-8.214742451
1	6202	8.205079007	1731	-8.303697285
2	1899	8.638599263	608	-7.907894737
3	1981	9.430792529	443	-7.761851016
4	1564	11.05236573	229	-9.040611354
5	1658	11.9208082	162	-10.25617284
6	1496	11.65815508	141	-12.10851064

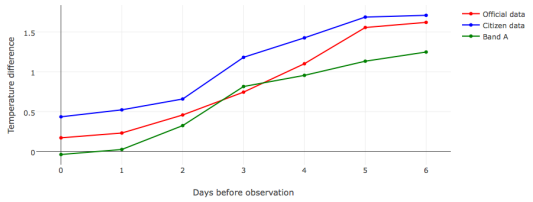
Table 7: Band U outlier statistics, where "Above" and "Below" are defined as above.

Days from observation	Above count	Above average	Below count	Below average
0	1749	7.553173242	328	-9.385365854
1	3489	7.707136715	576	-8.294444444
2	1155	7.724329004	249	-6.951004016
3	1076	8.683457249	146	-10.53219178
4	711	10.51125176	72	-15.71805556
5	745	11.46228188	56	-18.06428571
6	714	10.81792717	100	-11.692

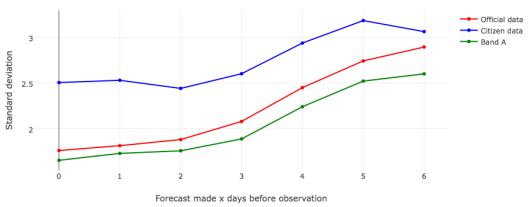
outlier removal, which is an expected consequence of deleting all outliers.

Band B mean temperature differences in Figure 18 now map almost perfectly to the observational data up to four days before observation, and after this it tails off, as in Figure 17a, albeit less so. The standard deviations map almost perfectly to the official data, showing great improvement from Figure 12b.

From Figure 19 and 20, we see that Band C and D mean temperature differences now map almost exactly to the citizen data, which is a great improvement from Figures 13a and 13b, which were significantly higher before outliers were removed. The standard deviations for both now map much closer to the standard deviations of the official observations, with both lines just slightly above those from official data, then fall slightly below 6 days before observation.

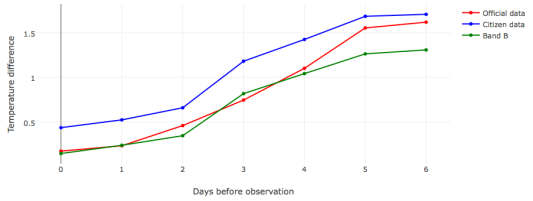


(a) Mean temperature difference Vs forecast length.

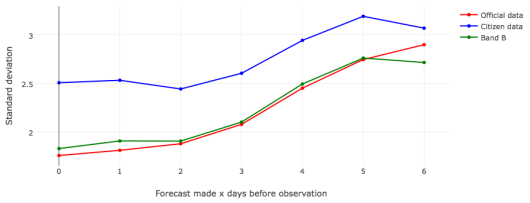


(b) Standard deviations Vs forecast length.

Figure 17: Band A mean and standard deviations with outliers removed for temperature differences Vs forecast length, along with their official and citizen counterparts from before as reference.



(a) Mean temperature difference Vs forecast length.

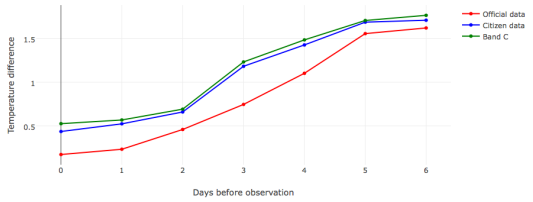


(b) Standard deviations Vs forecast length.

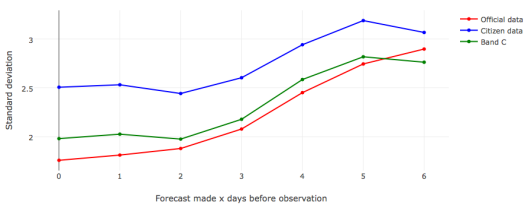
Figure 18: Band B mean and standard deviations with outliers removed for temperature differences Vs forecast length, along with their official and citizen counterparts from before as reference.

Band U approximately maps to the citizen data line too, however, the standard deviations map much more closely to the official data than band C or D. Note that in bands B-U there is a dip in standard deviations 6 days from observation for each. Removing the outliers have significantly changed the data for all bands, for both mean temperature difference and standard deviations.

Figures 17 - 21 show the marked improvement of removing outliers in mean temperature difference and standard deviations. This improvement is based on the assumption that citizen data should record similar temperatures to the observational data, and the observational data is a "gold standard" to compare to. However this assumption is flawed. It assumes that citizen data stations are distributed around the UK in a similar fashion to the official observation stations. This is not the case. Citizen stations are much more sparsely distributed in Scotland, for example, particularly in the highlands, and are much more dense in southern England. The problem with this, is that temperature differences may be not be uniform throughout the UK. The distribution of temperature differences may depend on geographical locations, which our assumption ignores. To test this, a heat map has been plotted of the temperature differences between

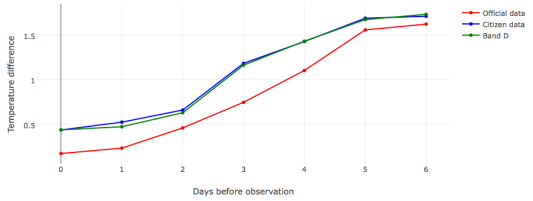


(a) Mean temperature difference Vs forecast length.

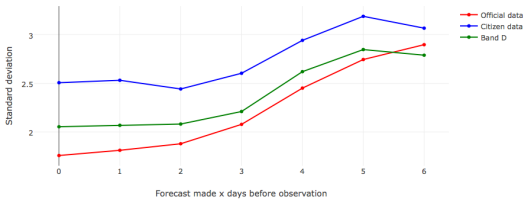


(b) Standard deviations Vs forecast length.

Figure 19: Band C mean and standard deviations with outliers removed for temperature differences Vs forecast length, along with their official and citizen counterparts from before as reference.

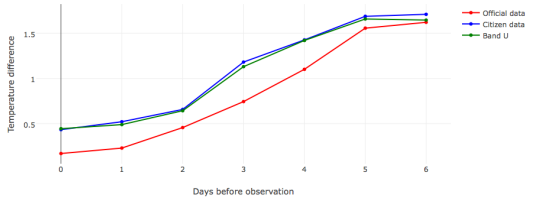


(a) Mean temperature difference Vs forecast length.

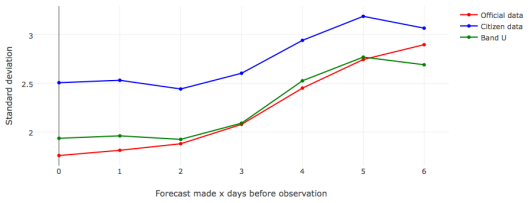


(b) Standard deviations Vs forecast length.

Figure 20: Band D mean and standard deviations with outliers removed for temperature differences Vs forecast length, along with their official and citizen counterparts from before as reference.



(a) Mean temperature difference Vs forecast length.



(b) Standard deviations Vs forecast length.

Figure 21: Band U mean and standard deviations with outliers removed for temperature differences Vs forecast length, along with their official and citizen counterparts from before as reference.

mean official temperature and forecast temperature for each. Figure 22 shows that the differences are approximately evenly spread, and that our assumption holds quite well.

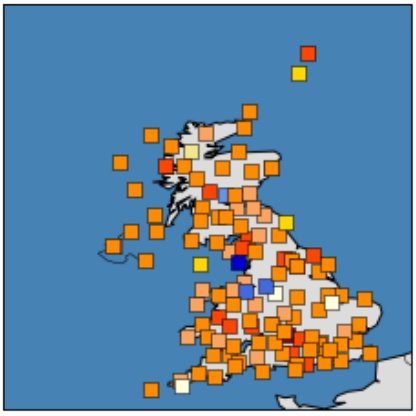
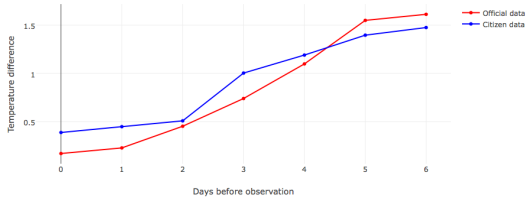


Figure 22: Heat map of official mean temperature differences.

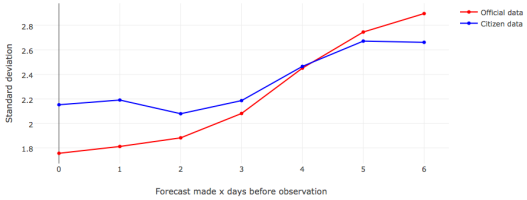
Figures 17 - 21 appear to suggest that simply removing the outliers will greatly improve the data. However, we have not considered the multi-dimensionality of the data. For example, removing the outliers 1 days before the observation on band A, may remove values which may not have been excluded from 3 days before the observation on band A. And so we lose much more data than expected. To test this, we created a new table, called `citizenTempOut`. This table was an exact copy of `citizenTemp` to begin with. We then began the process of eliminating the outliers for each specific band and "days from observation" permutation. The resultant data set was approximately a tenth of the size, and also greatly underestimated the temperature difference and standard deviations. Hence the multi-dimensionality of our data means a new outlier removal approach must be considered.

To counteract this, a new, less aggressive approach was adopted. For each temperature band, the outlier cut of point was calculated and the standard deviation of the temperature difference for each band was added to this. Meaning that we have increased the range of values which be kept, whilst still using a relative measure to do so. This allows most of the data to remain, whilst still removing extreme values. We will refer to this as *relaxed outlier removal*. The citizen mean temperature differences and standard deviations for all bands used relaxed outlier removal, and the official observation mean temperature differences and standard deviations are plotted in figure 23.

Figure 23 is a much better match to the data than Figure 9. Removing the extreme values makes the mean temperature differences fit very well for all days before observation, and the standard deviations are closer for all days greater than 2 days from observations date. The standard deviations are still markedly higher for 2 days to observation date and below. Figure 23 has been taken as sufficient proof that the relaxed outlier removal approach has improved the data set, and so all further analysis will be conducted using amended data set.



(a) Mean temperature difference Vs forecast length.



(b) Standard deviations Vs forecast length.

Figure 23: Citizen and official mean and standard deviations using relaxed outlier removal for temperature differences Vs forecast length.

5.5 Official Observations Vs Citizen Observations

To increase forecast granularity, there must be differences between the official observations and citizen observations. Otherwise, we could simply use the official observations to optimise forecasts. The mean temperature difference between citizen stations and official stations is still positive, with a value of 0.4234 and standard deviation of 1.150. This shows that there are marked differences between citizen and official sites. This is perhaps better depicted by Figure 24.

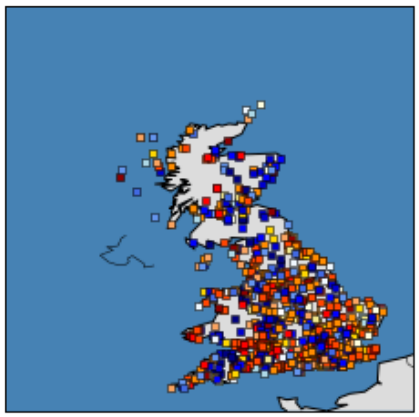


Figure 24: Heat map of the difference in mean temperature at each station.

There does not appear to be any discernible geographical pattern in where these differences occur. Before we attempt to optimise forecasts and increase granularity using the citizen stations, we are going to test the data veracity of the citizen sites more meticulously.

To do this, we can check the difference in mean temperature of citizen sites within 1.5 miles of official stations. The distance 1.5 miles has been chosen arbitrarily, but it stands to reason that these sites should be close to the official observations. The statistics in table 8 were generated from inspecting these sites.

Table 8: Temperature differences between citizen and official data.

AvgDiff	SDDiff	MaxDiff	MinDiff	MaxSD	MinSD
-0.14471597	0.652072201	1.536071	-4.12233	2.709771	0.610079

Where `AvgDiff` is the mean difference between citizen and official observations at each citizen station, `SDDiff` is the stand deviation of the mean differences for each site, `MaxDiff` is the maximum mean difference between citizen and official observations, `MinDiff` is the minimum mean difference between citizen and official observations, `MaxSD` is the maximum standard deviation, and `MinSD` is the minimum standard deviation.

The `AvgDiff` reasonably close to zero, as we might have expected from Figure 23a. However, `SDDiff`, `MaxDiff`, and `MinDiff` show that these values vary considerably. The mean difference of citizen stations below 1.5 miles is much closer to zero, and this could be due to official stations being in the country side, and so citizen stations close to this will too (recall that temperatures in built up areas are higher).This is not the result we desired in the interest of using citizen site to increase forecast granularity. It shows that citizen station vary more than should be expected so close to official stations, undermining citizen data credibility. Of course, this does not take into account differences in height above sea level, and assumes that the temperature should be approximately the same at two locations with 1.5 miles.

5.6 Optimising Forecasts Using Official Observations

In Section 5.2, it emerged that forecasts were underestimating temperatures at both citizen stations and official stations. We are working under the assumption that official stations do not record or upload errors, and so, it makes sense to first optimise forecasts using official observations. Figures 17 - 21 show that official temperatures are being underestimated to a greater extent further from observation date, and that the original forecasts are very close to the observed temperatures, and so we will use both of these as the features, or independent variables, in a linear regression equation, and observed temperature will be the dependent variable. The data was first split 75/25 as training and test data, where records were randomly sampled without replacement. It is important to randomly sample the data as records may be, and likely are, time dependent as weather trends may have changed as the summer draws closer to its end. The general linear model was then trained using the training data, and a model was inferred. The model co-efficients are shown in Table 9.

It can be more informative to view the forecast length separated out into 7 different coefficients, one for each forecast length. The right hand column contains the P-values for each coefficient, all of which are highly significant, suggesting that they are all important for the model. Next, we are going to examine the studentized residuals, which are shown in Figure 25, where red lines have been plotted on $|3|$ to check for outliers.

Table 9: Model co-efficients.

	Estimate	Std. Error	t-value	Pr(>|t|)
(Intercept)	0.6583124	0.049142947	13.395868	6.792516e-41
ForTemp	0.9987485	0.001590205	628.062818	0.000000e+00
ForLength0	-0.5067234	0.045960702	-11.025145	2.972624e-28
ForLength1	-0.4299624	0.045649715	-9.418732	4.634722e-21
ForLength2	-0.1679998	0.048109422	-3.492035	4.795105e-04
ForLength3	0.3367346	0.048574221	6.932374	4.156984e-12
ForLength4	0.7760983	0.048294319	16.070178	4.679978e-58
ForLength5	1.2121157	0.048380589	25.053761	3.314140e-138
ForLength6	1.1038832	0.049136137	22.465813	1.443247e-111

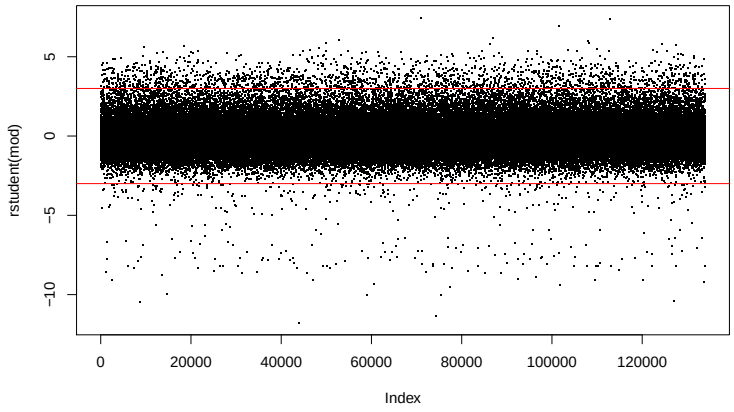


Figure 25: Studentized residuals from linear regression model on training data

The vast majority of the data falls within $|3|$, as required. There are some outliers, but no more than would be expected due to chance. The residuals below -3 appear to be more spread out that those above +3, however this is not overly concerning.

The next step is to test our normality assumption using a Q-Q plot. If the normality assumption holds, then the data should approximately fix to the plotted line. This is shown in Figure 26.

The Q-Q plot approximately follows the straight line for central values, but skews off significantly at either end, suggesting non-normality. Even when the data was transformed (log and cube root transformations were both attempted), it still did not fit to normality. We can explore this further by examining the residual plot show in Figure 27.

We can see from Figure 27 that the residuals are clearly correlated. In fact, the correlation coefficient is 0.495, suggesting that our normality assumption has been broken. Although this is not ideal, in the real world, often linear regression assumptions are broken. For these reasons, linear regression may not be the best way to model this data.

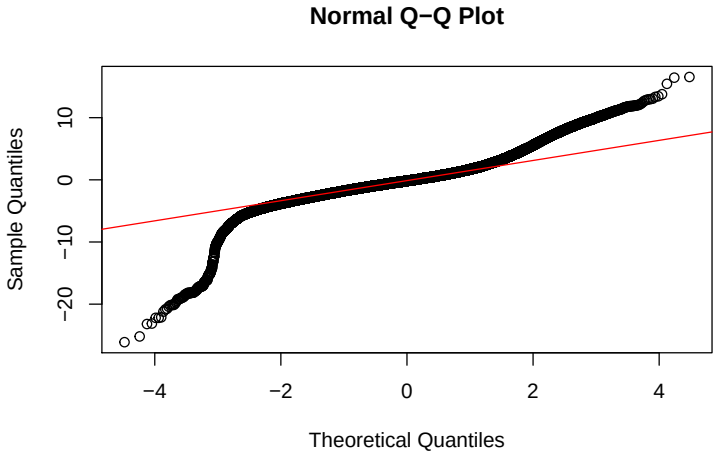


Figure 26: Q-Q plot for residuals on training data.

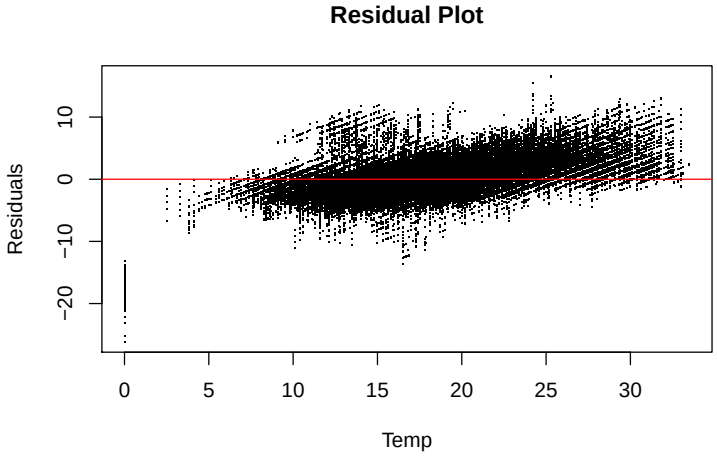


Figure 27: Residual plot.

However, the R^2 value for the test data achieved a value of 0.756. So even though our normality assumption has been broken, the model still fits the data reasonably well. Aside from the linear regression assumptions being broken, the trends observed over the course of this project cannot be said to hold true throughout a whole year. For this reason, increasing the forecast granularity has been performed separately, rather than combining these approaches as we would do if we knew this assumption held true all year. If the trends observed proved to hold throughout the year, then the optimisation and granularity increase could be combined trivially.

5.7 Using Citizen Sites To Increase Forecast Granularity

In Section 5.5, we showed that citizen data varied more than initially suspected. However, with no way to gauge the height of citizen stations, among other unknown factors, we can make a working assumption that this variation is due to actual differences in climate, rather than untrustworthy data. Under this working assumption, we will adjust forecast granularity using citizen stations. We are going to employ a rather basic, but effective climatological hybrid model, aforementioned in Section 2.1.2. This method assumes that the official observations on forecasts sites are correct at that site, but the surrounding citizen stations observe different weather because the weather has genuinely changed between the forecast site and each citizen station. These changes could be due of random variation caused by the chaotic nature of the atmosphere. Alternatively, these could be due to weather phenomena smaller than the forecast gird points, or small scale terrain features, not resolved by the primitive equations. We are also going to assume that forecasts are only wrong through random error, rather than exhibiting any trends. Although we have proved throughout this section that temperature forecasts do follow a trend, we are unable to ascertain if this trend only holds during the summer, or for all year round. If we knew if this trend held true all year, then we could trivially overlap this optimisation with increasing forecast granularity.

Our approach to increasing the forecast granularity using this citizen data, is to average the temperature differences between citizen station observations, and official observations at forecast sites. This difference will then be added to official observations to represent that particular location. A demonstration is displayed in Table 10.

Table 10: Climatological model demonstration.

statID	citTemp	offTemp	diff	forTemp	adjustedForTemp
12009	13.9817204	12.9774193	1.0043010	13.0108	14.0150537
189265	15.9808118	16.5025830	-0.5217712	15.8487	15.3269372
117944	16.5920454	16.4227272	0.1693181	15.8333	16.0026515
23535387	14.8641791	16.3731343	-1.5089552	15.7836	14.2746268

Where `statID` is the citizen station ID, `citTemp` is the mean temperature recorded by the citizen station, `offTemp` is the mean official temp recorded at the nearest forecast site, `diff` is the mean difference between official and citizen observations, `forTemp` is the mean forecast temperature for that area, and `adjustedForTemp` is the adjusted forecast temperature which represents the area around that citizen station, thereby increasing granularity.

Table 10 displays how forecasts can be adjusted using a climatological hybrid model. Although it is very simple, it has improved the forecasts of 3/4 of this random sample. Upon inspection of the rest of the stations, we can see that this ratio reflective of the population. This adjusted forecast will be set as the temperature forecast for the citizen station and its surrounding area, and hence, forecast granularity has been improved. This model is elegant in its simplicity, and achieves desirable results. However, it is

based on a number of assumptions which haven't yet been tested, and climatological models should be based on years of data instead of data which has been collected over the course of one summer. It also assumes the forecasts are only wrong through random error, rather than exhibiting a pattern.

6 Conclusions and Further Work

This chapter concludes the work undertaken in this project. Firstly, we will evaluate the techniques we employed to obtain and store data. Then, any important insights in our data will be outlined, before reviewing any forecasts optimisations performed. Finally, we will explore any further work which could be undertaken regarding this project, or any of it's subject matter.

6.1 Data Retrieval And Storage

Ideally, the Met Office would have been able to provide all of the requested data already neatly saved in consistent formats, stored in carefully structured tables, and ready to be imported into a database. Unfortunately, they were unable to help, and alternative data sources were explored. The best candidate was web scraping. Bots can scrape data directly into databases, and can scrape data from many web pages in little time. There is lots of support and documentation available for web scraping, and bots can be scheduled to run at all hours using a cron. However, several problems were encountered, including:

1. Weather varies significantly from month to month, with new trends and different behaviours each season. Bots can only scrape data which is online at the time of scraping. Which meant the only weather data available was the weather observed during a single summer, and so any inferences could not be said to hold throughout the calender year.

2. If a website is updated, the the bots which scrape from it must be be adjusted accordingly. Depending on the size of these updates, redesigning the bots can be extremely time consuming. Unfortunately, the Met Office took it upon themselves to update their website twice. This was extremely frustrating, particularly when similar types of data were stored in different formats between updates. This meant the data had to undergo significant cleansing.

3. The date formats used by the Met Office were not compatible with MySQL, and so unfortunately more data cleaning and bot adjustments were necessary.

Once the bots were scraping the data in the correct format, they could be stored directly into a database. In hindsight, it is better practice to design the database prior to designing the bots. Bots had to write into each table with exactly correct formats. Designing the bots first meant that data had to be reorganised, and bots required significant redesign.

MySQL was an excellent relational DBMS for storing, accessing, updating, deleting, and backing up data. It presented no unavoidable problems. A relation DBMS was chosen because the data is structured, joins were necessary for analysis, and the relationships between tables emulated real life relationships in the data. The database in this

project was normalised to 3rd normal form. Normalising data in this way makes the data more understandable, improves the performance of queries, reduces data duplication, and is generally considered good practice. Note: it is very important to appropriately index tables to achieve good performance in queries.

6.2 Data Understanding

Unfortunately, not all the citizen observation types could be directly compared with forecast types. For example, there are no forecasts available for pressure. Of the four data types that were directly comparable, temperature and wind speed presented themselves as candidates for forecast optimisation and increased granularity. They were candidates because the forecasts for these observations were consistently incorrectly predicting their behaviour.

Before taking a closer look at these candidate observation types, correlations between the citizen observations were examined. This provided a mixture of expected and unexpected results. The low correlation coefficient for humidity and rainfall was a stand out surprising result.

Of the two candidate observation types, temperature was chosen for further investigation as more stations recorded temperature than wind speed, and so there was a greater wealth of data to explore. Two important trends emerged: firstly, weather forecasts were significantly underestimating the observed temperature. Secondly, this effect was exaggerated with forecasts further from observation date. When this was compared with official observations, it became apparent that the same two trends were present in official observations, but to a slightly lesser extent.

Each citizen station had a set of site ratings, including a temperature rating. The two aforementioned trends were investigated further by examining the citizen temperature differences for each temperature rating, or band. It emerged that as the citizen station temperature rating got worse, temperatures observations were being over estimated further. The cause of this was deemed to be due to a disproportionate number of outliers in lower bands, and these outliers exhibited more extreme values, most of which were above the upper whisker, thereby overestimating the observed temperature. An attempt was made to remove these outliers from each band, but due to the multi-dimensionality of the forecast data, outliers were being removed too aggressively. A more relaxed, but still relative approach was then employed which removed only the extreme outliers. The data fit much closer to official observations, which was assumed to be infallible throughout this project, and so was a "gold standard" to compare with. The mean temperature differences between citizen stations with these extreme outliers removed, and official data was plotted on a map to establish that citizen stations exhibit no discernible geographical pattern in terms of temperature differences. Following this, citizen stations were found to significantly differ with official stations they were close to, potentially undermining the credibility of their observations. However, moving forward, this was assumed to be legitimate differences due to a range of factors other than distance.

6.3 Increasing Granularity and Optimisation

There is overwhelming evidence suggesting that forecasts significantly underestimate observed temperatures at both official and citizen stations. This left scope for optimising these forecasts based on the differences between forecast temperature and official temperature observations. The problem with optimising the data based on the available data set, is that trends observed over the course of this project cannot be said to hold throughout the year as seasons change. The results from this optimisation should be approached with caution. A machine learning approach was adopted in order to learn the regression coefficients from a training set, and subsequently evaluate the model on the test set. A studentized residual plot demonstrated that most of the data fell within $|3|$ as required, and the model achieved a R^2 value of 0.756, suggesting a good fit. However, some of the regression assumptions were broken, including non-normality of residuals. Due to no evidence suggesting that the temperature trends explored in this project hold throughout the year, this optimisation was not employed on the forecasts before increasing forecast granularity.

The forecast granularity was adjusted using a simple climatological hybrid model. This adjusted the forecasts based on the mean difference of a particular citizen stations temperature observations, and official temperature observations from the nearest forecast site. This adjusted forecast was assumed to represent the location of each citizen station and its immediate surroundings.

6.4 Further Work

The most obvious extension to this project, is to collect data over a whole year, and if possible, collect data over several years. The biggest problem with analysing data in such a small time frame, is that we have no way to test if observed trends hold throughout a full year. All the data in this project was collected in summer, when the weather is most stable. Weather in changing seasons (autumn, spring) is much more volatile, and so different results would likely be obtained. With more data, we could construct different learning algorithms, which would applicable during different times of the year. Once confidence in forecast optimisation has been achieved, this can be combined with a forecast granularity increase formula to fully optimise weather forecasts.

There were two candidate observation types which had the potential for increased granularity and optimisation, however, only one of these was explored. A good place to continue this project would be to explore the wind speed data, and if applicable, attempt to optimise it, and increase forecast granularity.

In this project, all citizen weather observations were recorded, however extreme. This meant they had to be deleted at a later stage. Weather observations from citizen stations are unlikely to vary a great deal from their closest official weather station. Instead of just removing every outlier without further investigation, outlying values should be

checked against official observations at the closest official weather station to inspect whether abnormal values were also recorded at that station. If so, the outlier remains, if not, the value should be removed. The removal cut off point could first be chosen arbitrarily. Later, this could be made more sophisticated using a formula which takes other factors into account, such as the time of year, or time of day. For example, during changing seasons, the cut off point would likely be less strict than during stable seasons. Introducing a sanity cap would filter out the most obvious inconsistencies prior to this.

All weather stations record measurement time, and forecasts also have a forecast time. Throughout this project, the time to forecast was measured in days. It might be interesting to measure the hours until observation date, or even minutes. This might achieve smoother plots. Additionally, all observations which matched forecast time in a single day were bunched together. However, weather observations might be more or less accurate for different times of the day, which this assumption ignores. This could be explored further with relative ease by adding in additional where clauses into queries.

The citizen sites were used to increase the forecast granularity. This process was based on a number of unproven assumptions. These should be investigated. Following this, these adjusted forecasts should be loaded into a GUI, and the data could be massaged into the grid so that it blends with the adjustments at different sites.

Throughout this project, we have assumed that official weather stations are infallible. However, it contained a number of obvious inconsistencies. The same outlier removal approach should be performed on the official data, so that fairer comparisons can be drawn.

6.4.1 Extensions to Machine Learning

The learning algorithm used in Section 5.6 was linear regression, however the data broke a number of linear regression assumptions, and may not have been the ideal supervised learning strategy. Another candidate is *backward propagation of errors*, or *backward propagation*. Backward propagation is excellent for learning in multi-layered neural networks, such as that presented with mulit-dimensional forecast data. Backward propagation has two phases:

1. **Propagation:**
    - Forward propagation of a training pattern's input through the neural network in order to generate the propagation's output activations.
    - Backward propagation of the propagation's output activations through the neural network using the training pattern target in order to generate the deltas (the difference between the targeted and actual output values) of all output and hidden neurons.
2. **Propagation:**
    - Multiply its output delta and input activation to get the gradient of the weight

- Subtract a ratio from the gradient of the weight [53]

The ratio is within the range [0,1], and is responsible for calculating the learning rate. Phase one and two are repeated until network performance is beyond a user determined threshold. Due to its ability to deal with multi-dimensional data, backward propagation is an ideal candidate to better model weather data.

References

[1] University of Edinburgh, EPCC, Lecture notes, Data analytics with HPC.

[2] Wikipedia, Weather forecasting, https://en.wikipedia.org/wiki/Weather_forecasting#Modern_methods

[3] Met Office, Supercomputers, http://www.metoffice.gov.uk/news/in-depth/supercomputers

[4] Thompson Higher Education, Chapter 14, Weather forecasting, http://www-das.uwyo.edu/ zwang/atsc2000/Ch14.pdf

[5] University of Illinois, Other forecasting methods, Climatology, http://ww2010.atmos.uiuc.edu/(Gh)/guides/mtr/fcst/mth/oth.rxml

[6] The NOAA, Billion-Dollar Weather and Climate Disasters: Overview, https://www.ncdc.noaa.gov/billions/overview

[7] Fatalities due to weather hazards, B. Geerts and E. Linacre, http://www-das.uwyo.edu/ geerts/cwx/notes/chap03/nat_hazard.html

[8] Met Office, UK synoptic and climate stations , http://www.metoffice.gov.uk/public/weather/climate-network/#?tab=climateNetwork

[9] Journal of Statistical Software, November 2015, Volume 68, Book Review 3.

[10] Web Scraping with Python - Collecting Data from the Modern Web, Ryan Mitchell, June 2015

[11] w3schools, Introduction to HTML, http://www.w3schools.com/html/html_intro.asp

[12] Data Journalism Handbook, Getting data from the web, http://datajournalismhandbook.org/1.0/en/getting_data_3.html

[13] Beautiful Soup Documentation 4.4.0, Beautiful Soup Documentation https://www.crummy.com/software/BeautifulSoup/bs4/doc/

[14] Mechanize, http://wwwsearch.sourceforge.net/mechanize/

[15] About tech, What is a Database, http://databases.about.com/od/specificproducts/a/whatisadatabase.htm

[16] Tutorial point, DBMS - Data schemas, http://www.tutorialspoint.com/dbms/dbms_data_schemas.htm

[17] University of Edinburgh, EPCC, Lecture notes, fundamentals of data management, lecture 03.

[18] Wikipedia, SQL, https://en.wikipedia.org/wiki/SQL

[19] University of Stamford, Machine LEarning, Lecture 1

[20] Met Office, DataPoint, http://www.metoffice.gov.uk/datapoint

[21] Admins choice magazine, Crontab âĂŞ Quick Reference, Setting up cron jobs in Unix and Solaris, http://www.adminschoice.com/crontab-quick-reference

[22] Robots exclusion standard, Wikipedia, https://en.wikipedia.org/wiki/Robots_exclusion_standard

[23] Techopedia, Foreign Key, https://www.techopedia.com/definition/7272/foreign-key

[24] Wikipedia, foreign key, https://en.wikipedia.org/wiki/Foreign_key

[25] Sheldon Robert (2005). Beginning MySQL. John Wiley & Sons. pp. 119-122

[26] Techopedia, One-to-Many Relationship, https://www.techopedia.com/definition/25122/one-to-many-relationship

[27] Microsoft, technet, Types of Table Relationships (Visual Database Tools), https://technet.microsoft.com/en-gb/library/ms190651(v=sql.105).aspx

[28] Met Office, WOW, Site Ratings, http://wow.metoffice.gov.uk/support/siteratings

[29] World Urban Database, Local Climate Zones, http://www.wudapt.org/lcz/

[30] Wikipedia, Dew Point, https://en.wikipedia.org/wiki/Dew_point

[31] Live Science, What is dew point?, Marc Lallanilla, February 11, 2014 http://www.livescience.com/43269-what-is-dew-point.html

[32] Wikipedia, Humidity, https://en.wikipedia.org/wiki/Humidity

[33] Wikipedia, Atmospheric pressure, mean sea level pressure, https://en.wikipedia.org/wiki/Atmospheric_pressure#Mean_sea_level_pressure

[34] Met Office, Official blog of the Met Office new team, what is "feels like temperature", https://blog.metoffice.gov.uk/2012/02/15/what-is-feels-like-temperature/

[35] Met Office, UV index forecast, http://www.metoffice.gov.uk/health/public/uvindex#?tab=map&map=MaxUV Index&zoom=9&lon=-2.50&lat=54.72&fcTime=1469372400

[36] Met Office, Key to symbols and terms, http://www.metoffice.gov.uk/guide/weather/symbols

[37] Techopedia, Entity Relationship Diagream(ERD), https://www.techopedia.com/definition/1200/entity-relationship-diagram-erd

[38] Webopedia, entity relationship diamgram (model), http://www.webopedia.com/TERM/E/entity_relationship_diagram.html

[39] Wikipedia, Variance, https://en.wikipedia.org/wiki/Variance

[40] Wikipedia, Standard deviation, https://en.wikipedia.org/wiki/Standard_deviation

[41] University of Edinburgh, EPCC, Lecture notes, Data analytics with HPC, lecture 14.

[42] Elements of Statistics - Fergus Daly, Et Al

[43] Wikipedia, P-value, Overview and controversy, https://en.wikipedia.org/wiki/P-value#Calculation

[44] Meehl, P. E. (1978). Theoretical risks and tabular asterisks: Sir Karl, Sir Ronald, and the slow progress of soft psychology. Journal of Consulting and Clinical Psychology, 46, 806âĂŞ834.

[45] Risk Assessment and Decision Analysis with Bayesian Networks (2012), Chapter 1, Fenton and Neil

[46] Barnett & Lewis, Outliers in statistical data. Wiley, New York, NY, 3rd edition, 1994

[47] Wikipedia, Interquartile range, https://en.wikipedia.org/wiki/Interquartile_range

[48] Variations of Box Plots, Robert McGill, John W. Tukey and Wayne A. Larsen, The American Statistician, Vol. 32, No. 1 (Feb., 1978), pp. 12-16

[49] Chicago Tribune, Ask Tom, http://articles.chicagotribune.com/2011-03-08/news/ct-wea-0309-asktom-20110308_1_humidity-cold-winter-air-moisture

[50] Sykes, Alan O. "An introduction to regression analysis." (1993).

[51] Statistics solutions, Assumptions of Linear Regression, http://www.statisticssolutions.com/assumptions-of-linear-regression/

[52] Wikipedia, Residual sum of squares, https://en.wikipedia.org/wiki/Residual_sum_of_squares

[53] Rumelhart, David E.; Hinton, Geoffrey E.; Williams, Ronald J. (8 October 1986). "Learning representations by back-propagating errors". Nature. 323 (6088): 533âĂŞ536.