Forecasting Grass Pollen in Cape Town:
A Comparison of Generalised Additive Models and Random Forests

Author:
Sky Cope
Chloë Stipinovich

Student Number:
CPXSKY001
STPCHL002

Research project submitted to the

Department of Statistical Sciences
University of Cape Town

In partial fulfillment of the requirements of the degree of
Statistical Sciences Honours

Supervised by
Dr. Birgit Erni

November 30, 2020

Plagiarism Declaration

1. I know that plagiarism is wrong. Plagiarism is to use another's work and pretend that it is one's own.
2. I have used a generally accepted citation and referencing style. Each contribution to, and quotation in, this tutorial/report/project from the work(s) of other people has been attributed, and has been cited and referenced.
3. This tutorial/report/project is my own work.
4. I have not allowed, and will not allow, anyone to copy my work with the intention of passing it off as his or her own work.
5. I acknowledge that copying someone else's assignment or essay, or part of it, is wrong, and declare that this is my own work.

Signature: _____

Chloë Stipinovich.
STPCHL002.

Signature: _____

Sky Cope.
CPXSKY001

Abstract

Airborne grass pollen is one of the main causes of allergic rhinitis in South Africa and around the world. Reliable and accessible pollen forecasts can help allergy sufferers, enabling them to appropriately plan their outdoor activities. In this paper we used generalised additive models (GAMs) and random forests to build seven day ahead grass pollen forecast models, using pollen count data collected at the South African Astronomical Observatory (SAAO) by researchers at the UCT Lung Institute. Variables used to predict pollen levels were primarily meteorological, but also included a vegetation index variable. To make our forecast easier to interpret, our models output probability distributions over five categories of pollen levels, from Very Low to Very High. Further, we created a Shiny app for potentially displaying the results of our forecast on a website. We found that the best-performing model was a GAM, which achieved a one day ahead ordinal Mean Absolute Error of 0.606 on unseen data, implying a reasonable level of agreement between observations and predictions. Unfortunately the pollen trap at the SAAO is not working at the time of writing, so our Shiny app does not currently display a real-time forecast.

Acknowledgements

We would like to thank our supervisor, Dr. Birgit Erni, for giving us the freedom to make this project our own, while making the time to provide us with as much guidance as we needed. Further, we would like to thank Dr. Dilys Berman and Dr. Jonny Peter from the UCT Lung Institute, for collecting the pollen count data and making them available to us. Finally, we are grateful to the NRF for financially supporting this research.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 The Need For Pollen Forecasts

It is currently estimated that between 20 and 30% of South Africans experience allergic reactions to pollen or fungal spores (Ajikah *et al.* 2020). Allergic rhinitis, conjunctivitis and asthma are all diseases exacerbated by exposure to heightened levels of fungal spores and pollen counts, and can have serious effects on quality of life. Potential costs due to these diseases are both direct and indirect: aside from the direct costs associated with medical treatment, indirect costs include lost productivity and days off work or school, resulting in a substantial socioeconomic burden (De Weger *et al.* 2013).

The rate at which carbon dioxide is being produced and retained is leading to an increase in global temperature. This has increased the 'growing time' of many plants (Teranishi *et al.* 2006; A. H. Fitter & R. S. R. Fitter 2002). Many plant species thrive under warmer, CO_2-filled environments. The resulting growth in vegetation increases the amount of pollen in the air. Devadas *et al.* (2018) state that the frequency of high pollen concentration days, as well as, asthma-triggering weather are projected to increase in the coming years. Diseases triggered by heightened pollen counts are likely to become even more serious public health threats as this projection is realised (De Weger *et al.* 2013).

Currently, pollen forecasts do exist in South Africa on weather sites and other such sites. However, these forecasts tend to be unreliable and provide no sense of uncertainty in their predictions. Bastl *et al.* (2017) assessed the performance of nine pollen forecasts across Europe, and found that the forecasts performed relatively poorly. The paper argues that accurate pollen forecasts are essential for bettering the quality of life for allergy sufferers. The paper additionally highlights the importance of probabilistic forecasting. Forecasts that include statements such as 'there is a high probability that grass pollen will be low today', rather than, 'grass pollen will be low today' better convey the inherent uncertainty associated with predicting the future states of complex systems. An understanding of this uncertainty is helpful for planning ahead.

There are various medical reasons to track and forecast pollen levels. (1) Immunotherapy is a preventative treatment for patients who experience sensitivity towards pollen and other allergens. It is crucial for the success of the treatment to begin implementation prior to the beginning of the pollen or grass season. According to the World Allergy Organisation (WAO) a sufficient pre-treatment period is essential to the success of Immunotherapy (Canonica *et al.* 2009). (2) If dependable pollen counts are available for a specific area, allergists are able to easily select the suitable aeroallergen skin prick for sufferers (Berman 2018). For example, some areas are more prone to grass pollen and some to tree or weed pollen depending on the

surrounding flora, wind direction and time of year. (3) Pollen forecasts have the capacity to aid medical professionals in the adequate provision of medication and preparation for patient visits (Smith & Emberlin 2005). Finally, (4) reliable and accessible pollen forecasts can be really useful for allergy sufferers, enabling patients to appropriately plan outdoor activities and stock up on necessary medication.

1.2 Aim

In this study, we aim to produce a reliable, week-ahead grass pollen forecast for Cape Town with historical pollen count data from the South African Astronomical Observatory, historical and forecasted weather data from Visual Crossings Weather and historical vegetation index data from NASA's MODIS satellite (Didan 2015). We additionally aim to make this forecast available to the public using an automated workflow whereby newly available data will be run through our forecasting model and published daily on a website.

1.3 Cape Town's Topographical Environment

Cape Town has a Mediterranean-like climate. Rain falls around 70 days of the year with about half this rainfall experienced over June, July and August (Cartwright *et al.* 2012). The weather is altered quite significantly by Table Mountain with areas closer to the mountain experiencing much more precipitation (Cartwright *et al.* 2012).

Cape Town is home to the Cape Floral Kingdom, supporting over 9 000 plant species of which more than two thirds are unique to this area (Joanne E. Taylor & Crous 2001). Much of the floral diversity is explained by the Fynbos biome, but since most Fynbos pollen dispersion is carried out by insects, birds and rodents, Fynbos pollen is not likely to contribute to pollen allergies. The most allergenic pollen in South Africa does not come from indigenous species, but rather from alien species such as Plane and Oak trees (Joanne E. Taylor & Crous 2001). A list of South Africa's most allergenic plant species can be found on The Real Pollen Count website and include trees such as Cypress, Oak, Plane and Olive, grasses such as Ryegrass, Winter, Wild Oat and Bunny grass, and weeds such as English Plantain, Dandelions and Reeds. Cape Town's highly populated suburbs are home to many of these alien species, such as Oak trees lining the street sides, and very few Fynbos species.

1.4 Data

1.4.1 Pollen Data

The pollen count data used in our analysis are recorded at The South African Astronomical Observatory (SAAO), in Observatory. The SAAO is positioned next to the Liesbeek river. It is surrounded by the Raapenberg Bird Sanctuary, Valkenberg Hospital and the River Club golf course (Berman 2018). As Dr. Dilys Berman (2018) explains in her thesis, the plants situated in the nearby park include *Eucalyptus*, *Pinus*, *Ulmus*, *Betulus*, *Kiggelaria Africana* (wild peach) *Rhus spp*, *Morus spp* and *Ficus spp*. The grounds of the site are large and away from big roads, making it an ideal location for a spore trap (Berman 2018).

At the SAAO, pollen counts are collected using a Burkard volumetric spore trap, positioned 5 meters above ground level (Berman 2018). The spore trap is designed to record pollen counts over a period of seven days. The trap allows 10 liters of air to flow through it per minute, where the air flows through specially-prepared sticky cellulose strips (Berman 2018). These strips are fixed to a drum, that rotates 360° every seven days. The cellulose strips are then

analysed using a microscope, and the number of total pollen spores are counted along three longitudinal traverses of the strip (Berman 2018). The location of the SAAO is shown in Figure 1.1.

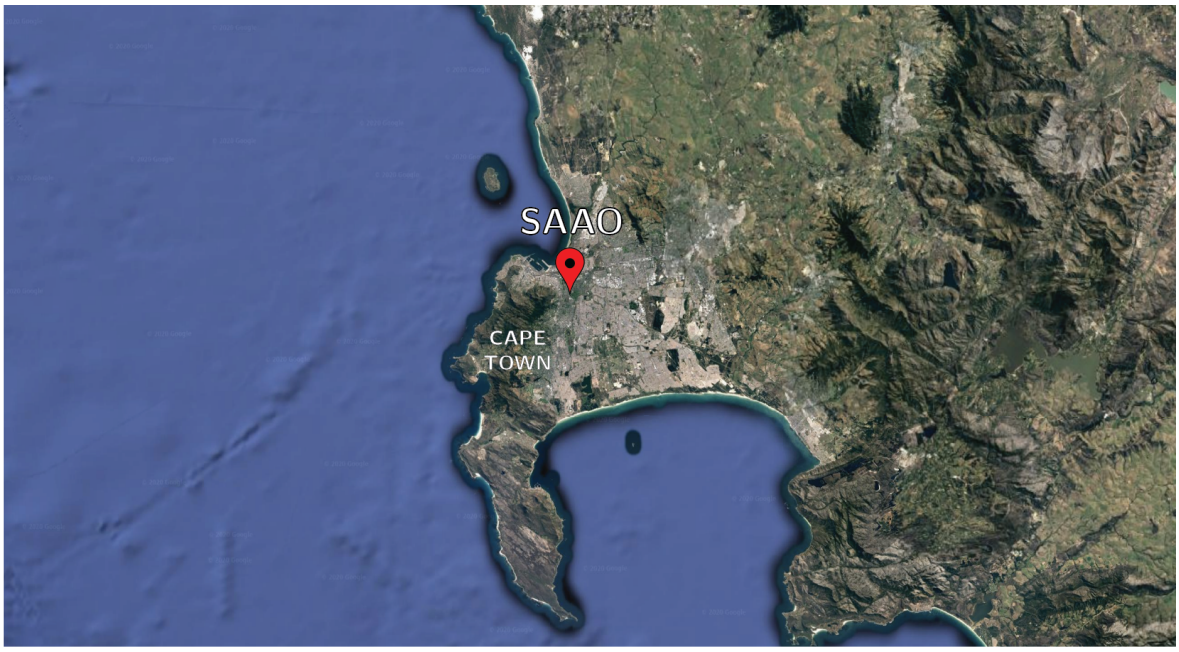


Figure 1.1: Satellite map showing the pollen sampling site.

Berman (2018) shows that between 2011 and 2014, grass pollen made up 23% of the total pollen index (TPI) at the SAAO site, second only to the Cypress tree at 24%. Myrtacea tree pollen came in third with 12.1% of the TPI. Grass pollen is the most widespread allergen in the world (Devadas *et al.* 2018). Our focus is therefore on developing a grass pollen forecasting tool for Cape Town, with the prospect of extending the tool in the future to other pollen types and locations.

1.4.2 Meteorological data

The meteorological data that we use in this study are collected from a weather Application Programming Interface (API) powered by Visual Crossings Weather. An API is a collection of programming instructions and standards that allows applications to access Web-based data and other services (Bloch 2006). An API allows multiple users to use the products made available by a software company. Using an API key and the GPS location of the SAAO, get requests are made to the API to obtain a series of weather features including measures of rain, wind speed, wind direction and temperature. Compared to the pollen data collection method, this method involves very little human input and so is a relatively sustainable and risk-free practice.

1.4.3 Vegetation Index data

Since Devadas *et al.* (2018) were able to successfully model pollen count variations using satellite-based measures of grass cover, we decided to use a satellite-based vegetation index in our models, in addition to the weather data. The index we use is called an Enhanced Vegetation Index (EVI), collected by NASA's MODIS satellite (Didan 2015). The satellite measures the degree of reflectance of blue, red, and near-infrared light, and uses this to measure the extent of vegetation in a given 250 m^2 region (Miura *et al.* 2000). NASA's user-friendly Application for Extracting and Exploring Analysis Ready Samples (AppEEARS) site

was used to extract the data (AppEEARS Team 2020). The time period and area must be specified by the user, and the following area surrounding the pollen collection site was specified, shown in Figure 1.2.

Figure 1.2: Vegetation Index area, downloaded from AppEEARS Portal.

Several variables are returned that summarise the different vegetation levels in all 250 m^2 pixels contained in the chosen area (Didan 2015). These summary statistics include the mean, median, inter-quartile range, etc. Since the vegetation index only updates every 16 days, we repeated each observed value of the index over the next 15 days, in order to deal with the absence of daily observations.

Chapter 2

Literature Review

First we summarise some key research findings relating to the spatial and temporal sources of variability in pollen concentrations. We then outline some approaches in the literature to forecasting grass pollen, which can be divided broadly into long and short term forecasts. Long term forecasts aim to make predictions about the start, duration and severity of the pollen season, while short term forecasts aim to make predictions about pollen concentrations in the near future. Finally, we outline some additional research that aims only to establish the strength and nature of observed relationships between pollen concentrations and other covariates, without forecasting anything.

2.1 Sources of Variability

Airborne pollen concentrations vary spatially between sites, and temporally within sites.

2.1.1 Spatial Variability

Spatial variability in airborne pollen concentrations is largely driven by differences in regional vegetation, elevation, and climate (Damialis *et al.* 2017; Haberle *et al.* 2014). The pollen found in a particular region is generally a reliable proxy of the region's vegetation and climate (Haberle *et al.* 2014).

However, Damialis *et al.* (2017) emphasise the importance of so-called 'transport incidents', where spores are carried long distances in the wind, away from their source locations. This means that the detected pollen spores at one site may not be reflective of the nearby vegetation. The distance a spore can travel is influenced by its areodynamic properties, which vary between pollen species (Damialis *et al.* 2017). For example, the *Pinaceae* or pine family is known for producing pollen with exceptional aerodynamic properties (Damialis *et al.* 2017). The result is that *Pinaceae* spores can travel long distances in the wind (Damialis *et al.* 2017). Similarly, the *Poaceae* family, consisting of 10 000 species of grass, also relies primarily on wind for transport (Ghitarrini *et al.* 2016). This family of flora produces large amounts of pollen that can travel kilometers from their source locations (Ghitarrini *et al.* 2016). Even longer-range transport incidents have been observed (Damialis *et al.* 2017). For example, Campbell *et al.* (1999) discovered pine and spruce pollen that had travelled roughly 3000 kilometers, landing in the Canadian Arctic.

2.1.2 Temporal Variability

Airborne pollen concentrations are determined in part by what Norris-Hill (1995) calls the 'flowering rhythms of plants'. These rhythms are seasonal, and are largely responsible for the

timing of the pollen season (Norris-Hill 1995). Biological rhythms are one source of temporal variability, but another key source is the weather (Aboulaich *et al.* 2013).

The weather affects pollen concentrations on two levels: before and during the pollen season (Aboulaich *et al.* 2013). Before the pollen season, temperature and rainfall can influence the development of plants and their flowers, which indirectly influences the amount of pollen that is released during the pollen season (Aboulaich *et al.* 2013). Precipitation (or its absence) affects pollen concentrations through two main mechanisms during the pollen season: pollen grains can be washed out of the air by rain, and the drying effect of a drought can promote the release of pollen grains (Aboulaich *et al.* 2013). Other meteorological variables include relative humidity, air temperature, wind speed, wind direction and sunshine hours (Berman 2018). Aboulaich *et al.* (2013) found a negative association between *Poaceae* pollen concentration and relative humidity. Transport of pollen is facilitated by the wind, and wind speed and direction are highlighted in the literature as important determinants of pollen concentrations (Berman 2018; Aboulaich *et al.* 2013). Temperature affects *Poaceae* concentrations during the pollen season, with Aboulaich *et al.* (2013) finding air temperature to be the strongest indicator of daily pollen concentrations. The relationship between air temperature and pollen concentration is somewhat nuanced: Norris-Hill (1997) proposed an optimal range of 21 to 27°C for pollen production, and Aboulaich *et al.* (2013) found a negative correlation between air temperature post-peak pollen season, and a positive correlation pre-peak. Supporting Norris-Hill's findings, Aboulaich *et al.* (2013) found that days with air temperatures between 21 and 27°C tended to have higher average pollen concentrations than other days. Finally, sunshine hours are positively correlated with pollen concentrations (Berman 2018).

2.2 Long Term Forecasts

Some pollen forecasts aim to predict the start, end and duration of the pollen season, as opposed to short term variations in pollen concentrations. An accurately-predicted start date allows allergists to commence immunotherapy with a sufficient buffer before the pollen season (Canonica *et al.* 2009). Additionally, knowing when pollen concentrations are likely to be highest allows sufferers to plan around that period. Many papers attempt to predict just this in a multitude of different methods. We outline a few papers and their methods below.

Zhang *et al.* (2015) used two observation-based techniques for predicting the start date of the pollen season (SD). The SD was defined as the day at which cumulative pollen counts surpass 5% of their annual total. The meteorological variables used included mean annual and monthly temperatures, cumulative annual rainfall, the previous year's SD, and pollen season duration (SL), amoung others (Zhang *et al.* 2015). The observation-based models were evaluated using cross-validation, where one year's data were omitted as the validation set and the other years were used to forecast SD and SL in the omitted year. This procedure was done repeatedly, to get an idea of the forcecast's accuracy (Zhang *et al.* 2015).

A study conducted in Poland by Stach *et al.* (2008) used multiple linear regression to predict key pollen dates such as the start, peak and end date of the pollen season. The start of the pollen season is stated in the paper as the day when the pollen count is greater than 30 grains per cubic meter. Meteorological variables such as maximum, minimum and average temperatures, rainfall, humidity, sunshine hours, and wind velocity were used in the model prediction. Only the linear regression model predicting the start date performed well with an R^2 value of 0.89. The peak and end date were not well captured.

Conducting research in areas of North London, Smith & Emberlin (2005) divided the pollen

season into pre-peak, peak and post peak stages. They thus needed to forecast when the start and end dates of the peak season were. The three variables considered in this forecast were 10-day averages of daily temperatures and rainfall as well as winter averages of the North Atlantic Oscillation (NAO). NAO is a change in atmospheric pressure at sea level that occurs in the North Atlantic Ocean (Visbeck *et al.* 2001). More of this study is discussed in the section below.

2.3 Short and Medium Term Predictions

For many years now, statisticians and allergists around the world have used various approaches to predict short term (one to two days ahead) to medium term (five to seven days ahead) airborne pollen concentrations.

Stach *et al.* (2008) also predicted short range pollen concentrations in Poland. Using regression analysis, meteorological variables such as maximum, minimum and average temperatures, rainfall, humidity, sunshine hours and wind velocity were used to predict pollen concentrations. The linear regression models achieved 61% and 70% accuracy in their 2005 and 2006 tests respectively (Stach *et al.* 2008).

Another paper mentioned above also develops a pollen forecasting model. The week ahead model built by Smith & Emberlin (2005) divides the pollen season into pre-peak, peak and post-peak pollen seasons as explained above. They used three multiple regression models for the daily grass pollen count prediction in each of the stages. The variables considered for the 7-day ahead forecast included maximum daily temperature, average preceding pollen counts (2, 3, 4 and 5 days prior), wind speed and relative humidity (Smith & Emberlin 2005). Their 7-day ahead model performed sufficiently well on unseen data from the subsequent period achieving 62% accuracy in 2000, but poorly on unseen data two periods after as these data included higher pollen counts than were seen in the training data, achieving 47% accuracy. (Smith & Emberlin 2005) Their analysis shows that by breaking up the pollen season into three periods they were better able to identify the specific variables that effected the pollen count.

Using Cypress pollen count data collected in Spain over a period of 10 years, Valderrama *et al.* (2009) forecasts pollen concentrations using a two-step functional regression model. Importantly, the forecast was a short-term daily forecast, and predictions were made only during the peak pollen season (Valderrama *et al.* 2009). To account for seasonality, Valderrama *et al.* (2009) did not use one long time series, but instead truncated it into yearly intervals. These intervals were then subdivided into *past* and *current* intervals, where the past interval is prior to the forecasted period, and the current interval is the forecasted period (Valderrama *et al.* 2009). The approach aimed to predict the evolution of pollen concentration in the current period using only temperature data in the past period (Valderrama *et al.* 2009). Root mean squared errors (RMSE) were calculated for the predicted sample path and observed sample path, and the predicted sample path was compared with a Box-Jenkins moving average in order to validate the model (Valderrama *et al.* 2009). The results were impressive, and the model is able to explain 81.09% of the variation in the current period using data in the previous period.

Instead of forecasting pollen counts, De Weger *et al.* (2013) developed a 5-day ahead model for forecasting hay fever symptoms in the Netherlands. Their two-step approach first predicted grass pollen concentrations, and then used these predictions to forecast the intensity of hay

fever symptoms (De Weger *et al.* 2013). The model used recorded pollen counts and registered meteorological variables for the years 1994 to 2004 (De Weger *et al.* 2013). These variables included temperature (minimum, maximum and mean), as well as relative humidity and cloud cover. A multiple regression showed that the weather variables, along with the previous day's pollen count were effective at explaining variation in the following day's pollen count ($R^2 = 0.74$) (De Weger *et al.* 2013). Since the recorded counts and registered weather data for previous days are not available when forecasting several days ahead, forecasted weather variables were used in the final pollen and hayfever forecasts (De Weger *et al.* 2013). Multiple regression analysis was used for the pollen and hayfever models, and pollen counts were square rooted to improve the distribution of the response (De Weger *et al.* 2013). Covariates used in their pollen forecast model included forecasted daily temperature, as well as lagged pollen counts (De Weger *et al.* 2013). Further, derivatives of forecasted temperatures and pollen counts were added to the regression. Their model performs well against historical data, but this performance declines as the model predicts further ahead (De Weger *et al.* 2013).

2.3.1 Other Approaches

Some studies aim only to assess the strength and nature of observational relationships between pollen concentrations and other covariates.

Erbas *et al.* (2007) made use of generalised additive models (GAMs) in order to model ambient pollen concentration in Melbourne, Australia. GAMs are a slightly different multiple regression model to those mentioned above in that they allow for a more flexible relationship between the response and the covariates. GAMs are discussed in detail further on in the paper. Erbas *et al.* (2007) assumed that the daily pollen counts were Poisson distributed and then utilized non-parametric, penalized smoothing splines to model the count. Scatter plots were used to gauge which variables affected the pollen count. Mean temperature, mean wind speed and direction, mean humidity as well as the day of the pollen season from 1 October to 31 December were considered in the model. Erbas *et al.* (2007) used wind direction as a parametric term and the remaining variables as smooth, continuous parameters while opting to manually define the degrees of freedom where appropriate. The paper notes that GAMs are powerful tools as they make no assumption about the distributions of the weather variables. Additionally, they are able to account for autocorrelation in the daily pollen counts. Although no prediction was conducted, the paper concludes that GAMs could be particularly helpful in forecasting pollen concentrations.

A study done by Devadas *et al.* (2018) quite uniquely used satellite imagery of surrounding grass cover and its growing periods to give insight into the behaviour of airborne grass pollen. This study did not aim to forecast pollen concentrations, only to assess the vaibility of satellite imagery in explaining variations in pollen counts. The study is important as this method of remote sensing data collection does not rely on labour and expertise intensive pollen trap management and could add to the forecasting of pollen concentration in a meaningful way. The authors argue that a major weakness in current pollen forecasting methods is a lack of observation-based data such as where plants are situated, when their budburst and flowering periods are and other such data that can be captured via aerial images. The aim of the study was to evaluate whether or not satellite data could be used to enhance pollen forecasting models. Using a generalised additive model (GAM), Devadas *et al.* (2018) used mean monthly Moderate Resolution Imaging Spectroradiometer (MODIS) grass cover and greenness data. Their results were mixed between testing areas. The models for all three of the French sites performed well with R^2 values between 0.84 and 0.9, while those for the two Australian sites performed less well with R^2 values of 0.34 and 0.78 (Devadas *et al.* 2018). They indicate that the significantly greater wind speeds experienced at the Australian sites as well as the second,

unmonitored pollen season might explain their poor prediction results. The study found that remote satellite sensing data is a helpful and impactful means for the summarising of complex land cover status and can certainly be used to augment predictive models in the future.

Finally, Zewdie *et al.* (2019) used machine learning methods to model daily pollen levels, using pollen counts recorded at the University of Tulsa, Oklahoma, in the United States. The paper uses random forests, Neural Networks and Support Vector Machines (Zewdie *et al.* 2019). The data are not meteorological as is the case with most studies of this kind, but are a combination of environmental variables and NEXt-generation RADar (NEXRAD) radar variables (Zewdie *et al.* 2019). The results are good, especially considering that no weather variables are used. The best performing model on the test set is a random forest model, achieving an R-squared value of 0.37 (Zewdie *et al.* 2019).

Chapter 3

Methods

In order to successfully forecast pollen counts, we needed to use appropriate statistical models. Two model classes we used include generalised additive models (GAMs) and random forests. This chapter introduces these model classes.

3.1 Introduction

Many statistical methods are concerned with estimating the relationship between the covariates and the response (James *et al.* 2013). We assume there is some relationship, and then express it in general terms as

$$Y = f(X) + \epsilon, \tag{3.1.1}$$

where f is some function that represents the true underlying relationship between the Xs and Y, and ϵ is an error term that is independent of the Xs and centered about 0. The aim of these methods is to estimate f for the purposes of prediction, inference or both (James *et al.* 2013). The methods available all differ in their interpretability and flexibility. Highly flexible methods enable us easily to estimate f using many different shapes, while highly interpretable methods enable us to understand the underlying relationship between the covariates and the response (James *et al.* 2013). In general, there is a trade-off between interpretability and flexibility: the more flexible methods tend to be harder to interpret, and vice versa (James *et al.* 2013).

In this project we used two methods for forecasting grass pollen: generalised additive models (GAMs) and random forests. Both methods resolve the trade-off between flexibility and interpretibility in different ways: GAMs being the more interpretable but less flexible of the two, in general (James *et al.* 2013). These methods are explained in some detail below.

3.2 Generalised Additive Models

3.2.1 Basis functions

Many real-world data are not linear. While the assumption of linearity underpinning linear regression is often good enough, there are some cases where we cannot justifiably assume that the response is some linear function of the covariates (James *et al.* 2013). Below is an example of some simulated data that are clearly non-linear, with the true underlying relationship given as:

$$y = 2 + 5x\sin(4\pi x) + \epsilon, \text{ where } \epsilon \sim \mathrm{N}(0, 1.2) \tag{3.2.1}$$

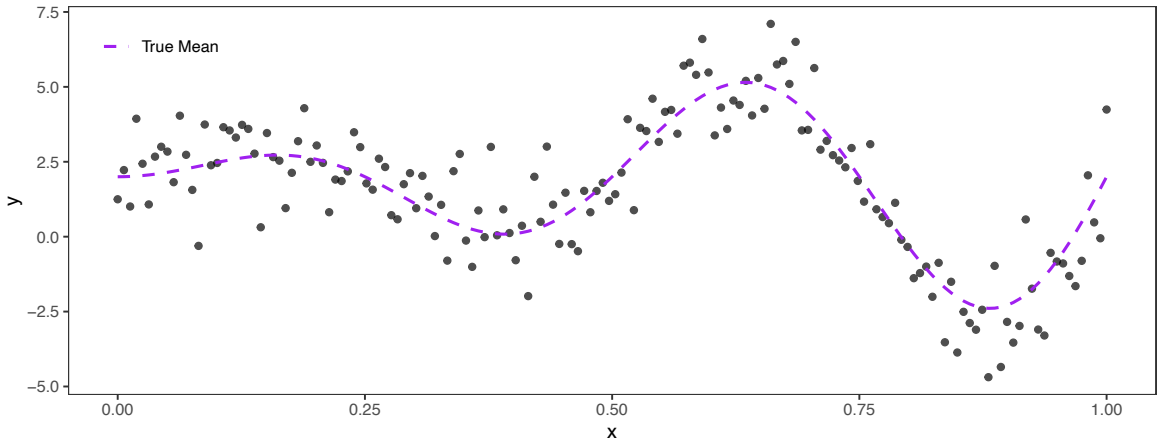


Figure 3.1: Non-linear simulated data, according to 3.2.1.

A simple linear regression model would be inappropriate for modeling these data, but other non-linear regression approaches are available to us. One way to model this non-linear function f is to approximate it with basis functions (James *et al.* 2013). A basis is a space of functions that contains f, or at least enables us to well approximate it using linear combinations (Wood 2017). The linear combination of the j-th basis function b_j, weighted by β_j is simply:

$$f_j(x_i) = \sum_{j=1}^{k} b_j(x_i)\beta_j \tag{3.2.2}$$

The idea that we can use basis functions to approximate f is used in polynomial, piece-wise polynomial and spline regression methods, amoung others.

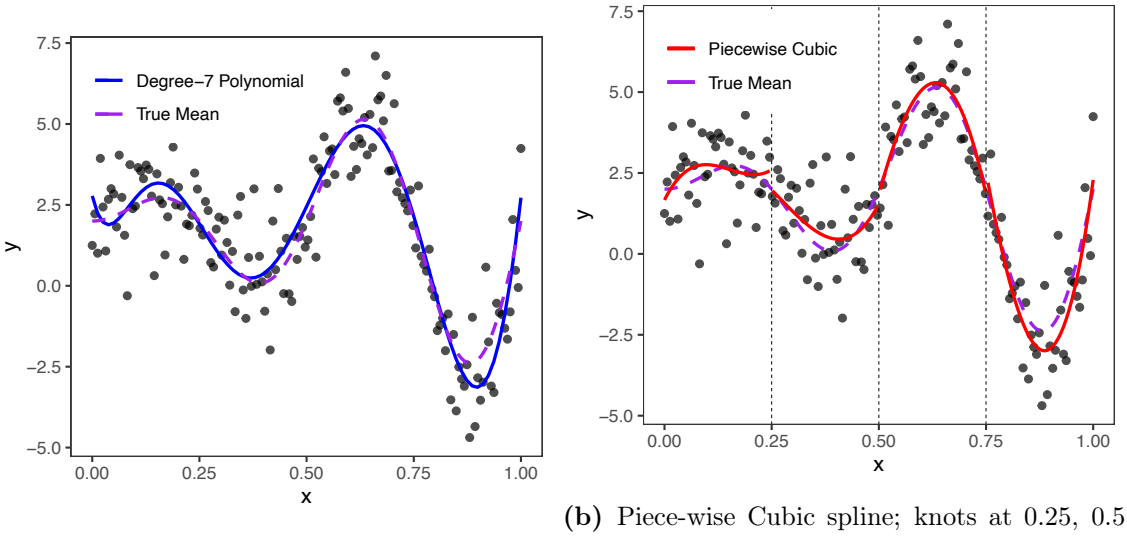
3.2.2 Polynomial and piece-wise polynomial regression

Polynomial regression models have the following general form, for a single explanatory variable:

$$y_i = \sum_{j=0}^{k} x_i^j \beta_j + \epsilon_i. \tag{3.2.3}$$

Clearly the j-th basis function is $b_j(x_i) = x_i^j$. The explanatory variable is simply raised to successively higher powers from 0 to k, and the corresponding β_js are estimated, leading to a non-linear fitted curve that approximates the true underlying relationship. k can be set to some reasonably low value, or can be tuned using cross-validation (James *et al.* 2013). One issue is that polynomials tend to extrapolate very poorly beyond the observed range of values, and predictions near the boundaries of this range tend to be unreliable too. Further, polynomials are non-local, in that parameters are estimated for the whole range of values: changes in any particular observation could affect the entire fitted curve (James *et al.* 2013).

Instead of using a single polynomial regression model, piece-wise polynomial regression involves fitting different models across different ranges of the x-axis. The data are split up at points along the x-axis called 'knots', and the regression coefficients change at the knots (James *et al.* 2013). Below are illustrations of polynomial regression and piece-wise polynomial regression, fitted to the above data:



(a) Degree-7 Polynomial.

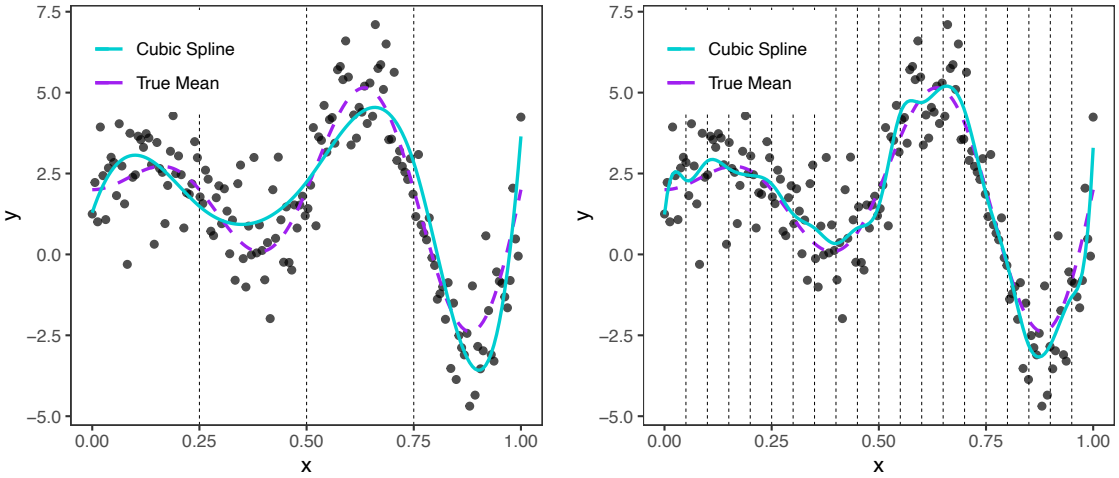
(b) Piece-wise Cubic spline; knots at 0.25, 0.5 and 0.75.

Figure 3.2: Polynomial regression versus piece-wise polynomial regression.

The degree-7 polynomial regression model fits quite well to the data, but over-predicts close to 0. The piece-wise cubic model also fits quite well, but there are clear discontinuities where we chose to place the knots. We'd like to get rid of the discontinuity here, as it is in general extremely unlikely that the true underlying relationship is in fact discontinuous wherever we choose to place the knots. This introduces the idea of a spline.

3.2.3 Regression splines

Splines are piece-wise regression models with added constraints (James *et al.* 2013). These constraints ensure that the fitted curve is maximally continuous at the chosen knots. Cubic splines are piece-wise cubic regressions with continuous first and second derivatives at each knot (James *et al.* 2013). Here two cubic splines are fitted, using different numbers of knots.



(a) Cubic Spline with 3 knots.

(b) Cubic Spline with 19 knots.

Figure 3.3: Two cubic splines with differing numbers of knots.

Clearly there is no noticeable discontinuity at any of the knots, which is an improvement over the piece-wise cubic model. However, the spline with three knots appears too smooth and

inflexible, while the spline with nineteen knots appears too wiggly, and is possibly overfitting. In general, increasing the number of knots increases a spline's 'wiggliness', but the fit of the model is dependent on both the number and location of the knots (James *et al.* 2013). We need a better way of choosing the number and location of the knots, because doing so manually is essentially arbitrary, and it is hard to know whether our choice is the optimal one. One way to get around this issue is by using penalised regression splines.

3.2.4 Penalised spline regression

Instead of choosing the number and location of the knots, we can set the number of knots to be much larger than needed and space them evenly (James *et al.* 2013). We can then control the wiggliness of the fitted curve with a single tuning parameter, λ. In estimating the model parameters, $\boldsymbol{\beta}$, we no longer minimise only the Residual Sum of Squares (RSS) but instead minimise the following expression

$$||\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}||^2 + \lambda \int [f''(x)]^2 dx. \tag{3.2.4}$$

The left term is the RSS. The right term is the integrated squared second derivative of the fitted smooth function, $f(x)$ (James *et al.* 2013). This second term is large when $f(x)$ is very wiggly and small when $f(x)$ tends to a linear function. As $\lambda \to \infty$, $f(x)$ approaches a linear function, with a second derivative of 0. When $\lambda = 0$ we just minimise RSS, so the regression spline is un-penalized, and will be as wiggly as possible (James *et al.* 2013). Now with a single parameter to tune, we are left with the question of how best to choose λ. Two methods for choosing λ are Generalised Cross-Validation (GCV) and Maximum Likelihood Estimation (Wood 2017).

To choose λ using GCV we can first re-express (3.2.4) in a more computationally-attractive way (Hastie *et al.* 2001). Denote $\hat{\boldsymbol{f}}$ as the vector of fitted values, calculated as follows:

$$\hat{\boldsymbol{f}} = \boldsymbol{A}_\lambda \boldsymbol{y} \tag{3.2.5}$$

where $\boldsymbol{A}_\lambda$ is the smoothing matrix, or hat matrix (given the smoothing parameter λ) and $\boldsymbol{y}$ is the vector of observed response values (Hastie *et al.* 2001). We can re-write $\boldsymbol{A}_\lambda$ in the *Reinsch* form, where we introduce a new matrix, $\boldsymbol{S}$, that does not depend on λ:

$$\boldsymbol{A}_\lambda = (\boldsymbol{I} + \lambda \boldsymbol{S})^{-1} \tag{3.2.6}$$

Now (3.2.4) can be expressed instead as:

$$||\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}||^2 + \lambda \boldsymbol{\beta}^T \boldsymbol{S} \boldsymbol{\beta} \tag{3.2.7}$$

where the second term approximates the squared second differences of fitted values at each knot, added together. This second term behaves very similarly to the second term in (3.2.4), and is proportional to the wiggliness of the fitted curve. Cross-validation gives us a way to choose λ such that $\hat{f}$, our fitted curve, performs the best on unseen data (Wood 2017). The performance of the model on unseen data can be measured with the *ordinary cross-validation* score, $\mathcal{V}_o$:

$$\mathcal{V}_o = \frac{1}{n} \sum_{i=1}^{n} \left(\hat{f}_i^{[-i]} - y_i \right)^2 \tag{3.2.8}$$

where $\hat{f}_i^{[-i]}$ denotes the estimate of y_i for a model built using all observations except the i-th one (Wood 2017). The differences between the observed and fitted values are calculated, squared and averaged. $\mathcal{V}_o$ increases as models start to overfit or underfit. Therefore, a good

criterion for choosing λ would be to set it to the value that minimises $\mathcal{V}_o$. However, calculating $\mathcal{V}_o$ can be computationally-expensive, and usually the *generalised cross validation* score is used instead (Wood 2017). This uses the fact that

$$\mathcal{V}_o = \frac{1}{n}\sum_{i=1}^{n}\frac{\left(y_i - \hat{f}_i\right)^2}{\left(1 - A_{ii}\right)^2} \tag{3.2.9}$$

where $\boldsymbol{A}$ is the influence or hat matrix, which yields the fitted values $\hat{\boldsymbol{\mu}}$ when post-multiplied by $\boldsymbol{y}$, the vector of observed response values: $\hat{\boldsymbol{\mu}} = \boldsymbol{A}\boldsymbol{y}$ (Wood 2017). $\hat{f}_i$ is the estimate of y_i using a model fitted to the whole dataset. The GCV score $\mathcal{V}_g$, replaces the A_{ii} term with the mean of the diagonal elements of $\boldsymbol{A}$, $\mathrm{tr}(\boldsymbol{A})/n$, giving us the GCV score:

$$\mathcal{V}_g = \frac{n\sum_{i=1}^{n}\left(y_i - \hat{f}_i\right)^2}{\left[n - \mathrm{tr}(\boldsymbol{A})\right]^2}. \tag{3.2.10}$$

Below is a plot showing the GCV score, $\mathcal{V}_g$, for different values of λ, using the same simulated dataset as before:

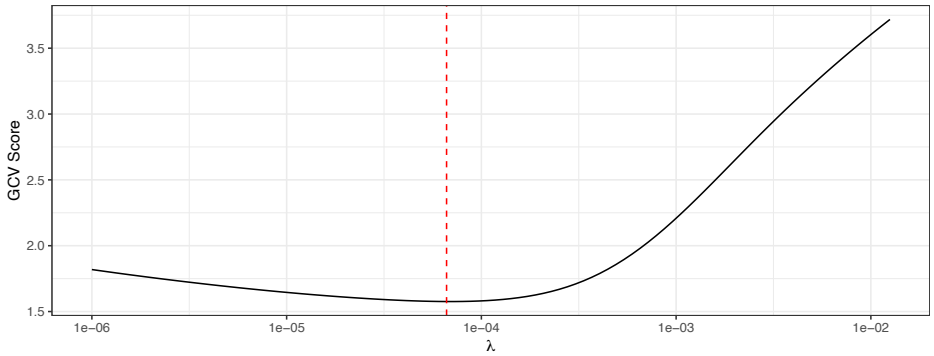
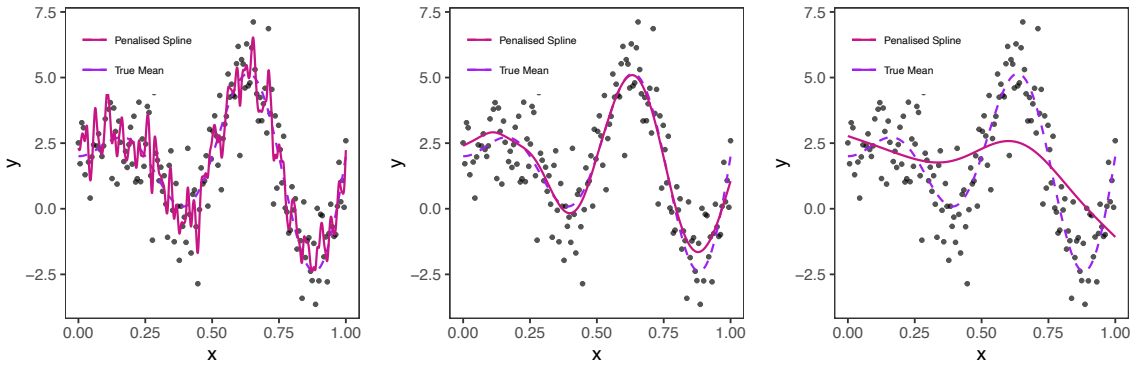


Figure 3.4: GCV Score as a function of λ.

In this case, $\mathcal{V}_g$ is minimised at $\lambda^* \approx 6.63 \times 10^{-5}$. Below three penalised regression splines are fit to the data, each with different values of λ: one that is too low, and that is too high, and one that is just right. All three splines are fit using cubic bases.



(a) $\lambda^* = 10^{-8}$ **(b)** $\lambda^* \approx 6.63 \times 10^{-5}$ **(c)** $\lambda^* = 10^{-2}$

Figure 3.5: Penalised regression splines, with different smoothing parameter values.

The fitted curve shown in Figure 3.5b fits the data well, and doesn't obviously appear to overfit or underfit. The other curves clearly overfit, shown in Figure 3.5a and underfit, shown

in Figure 3.5c. This is evidence that minimising $\mathcal{V}_g$ might be a good way to choose the smoothing parameter.

Penalised regression splines with cubic bases are in many cases good at modelling non-linear univariate data with normally-distributed responses. However, we might like to model data with several explanatory variables, or with non-normally distributed responses. This introduces the Generalised Additive Model (GAM).

3.2.5 Generalised Additive Models

GAMs enable us to model data with multiple covariates, where some, but not-necessarily all covariates are related to the response in some non-linear way (James *et al.* 2013). Further, GAMs are Generalised Linear Models (GLMs), as they are able to accommodate response variables that are not normally-distributed, like count or binary data (Wood 2017). GAMs have the following general structure:

$$g(\mathbb{E}(y_i)) = \beta_0 + f_1(x_{i1}) + \cdots + f_p(x_{ip}) \tag{3.2.11}$$

$$y_i \sim \text{EF}\left(\mathbb{E}(y_i), \phi\right) \tag{3.2.12}$$

where the response variable y_i follows some distribution in the Exponential family with mean $\mathbb{E}(y_i)$ and shape parameter ϕ, f_i represents the smoothed function of the corresponding covariate x_i and g represents the link function, which can be identity, logit or other (Wood 2017). The smooth functions are splines, which can have a variety of bases. The types of splines that can be used in GAMs include cubic, thin-plate, cyclic cubic, tensor product and seasonal regression splines (Wood 2017).

The Mixed GAM Computation Vehicle (MGCV) package is a highly capable suite of functions for fitting GAMs in R (Wood 2017). Below we briefly discuss how parameters are estimated in `mgcv`.

3.2.6 Parameter estimation

One algorithm used to estimate parameters is called penalized iteratively re-weighted least squares (PIRLS), where the penalised likelihood function is given as

$$l_p(\boldsymbol{\beta}) = l(\boldsymbol{\beta}) - \frac{1}{2}\sum_{j=1}^{m} \lambda_j \boldsymbol{\beta}^T \boldsymbol{S}_j \boldsymbol{\beta} \tag{3.2.13}$$

where $l(\boldsymbol{\beta})$ is the log-likelihood of the model (Wood 2017). The vector, $\boldsymbol{\beta}$, that maximises $l_p(\boldsymbol{\beta})$ is estimated numerically using Newton's Method (Wood 2017). The vector of smoothing parameters $\boldsymbol{\lambda}$ (one for each smooth function) is estimated using a generalised version of GCV, at every iteration of the PIRLS algorithm (Wood 2017).

3.3 Random Forests

Proposed by Breiman (2001) in his seminal paper, random forests are a widely used technique applicable to both classification and regression methods. The technique involves an ensemble of many randomized decision trees and the prediction of the ensemble is found by aggregating the predictions of the individual trees (Biau & Scornet 2016).

3.3.1 Decision trees

Tree-based methods involve the stratification, or segmentation, of the feature space into a number of rectangular regions (James *et al.* 2013). The mean, or the mode, of the observations in a specific region is then found and used to make predictions for future observations that fall within the same region (Hastie *et al.* 2001). Rather than allowing the tree to split on any number of variables simultaneously, decision trees only make binary spilts on one feature at a time (James *et al.* 2013). Figure 3.6 shows an example of feature segmentation that makes use of multiple splits. The construction of the vertical line requires more than one split in the feature space. Figure 3.7, on the other hand, uses only binary splitting and is the type of stratification used in the construction of random forests.

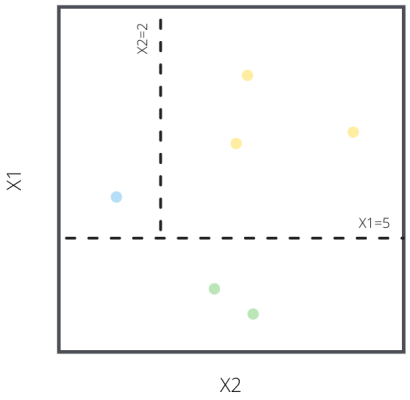


Figure 3.6: Example of a decision tree that uses multi-splits to segment a feature space.

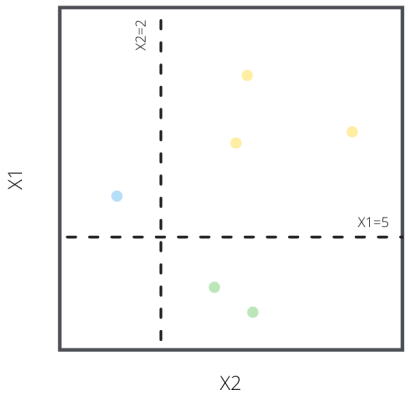


Figure 3.7: Example of a decision tree that uses only binary splits to segment a feature space.

3.3.2 Regression Trees

During regression problems, we start by splitting the feature space into two regions such that the residual sum of squares (RSS) is minimized. The response for each new region is then modelled as the mean of the observations in that space (James *et al.* 2013). Either of the first two regions is then split into two more regions and this process is continued until some stopping criterion has been reached (James *et al.* 2013).

Figure 3.8 is an example of a simple regression tree. The data are from the `Credit` data set from the `ISLR` package in R (James *et al.* 2017). The tree makes splits depending on an individual's credit rating, `Rating`, and average account balance, `Balance`, in order to predict their income. Each segmented region is the average income of all individuals in that region and is then the predicted value for any future observation that may fall in that same space. If the condition above a node is true, an observation moves left along a branch.

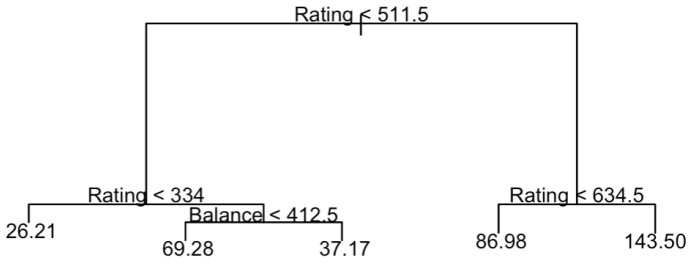


Figure 3.8: Example of a regression tree that makes splits depending on an individual's Credit Rating and Balance in order to predict their income.

There are two general steps (Hastie *et al.* 2001).

1. Divide the predictor space into J unique independent regions, $R_1, R_2, ..., R_J$, that minimize the residual sum of squares (RSS),
$$RSS = \sum_{j=1}^{J} \sum_{i:x_i \in R_j} (y_i - \hat{y}_{Rj})^2. \tag{3.3.1}$$
2. Assign the mean of the training set observations contained in region R_i to new observations that fall in the same region (Hastie *et al.* 2001). This mean is the predicted y_i value.

Finding the RSS for every possible split on all variables at every branch is computationally infeasible (Hastie *et al.* 2001). We thus use a top-down, greedy algorithm known as recursive binary splitting. The algorithm starts at the root node (top-down) and then makes splits on each new predictor space. The algorithm does not look forward to consider whether a worse split at the current branch might lead to a better split further on. The algorithm is thus a so-called greedy algorithm, considering only the best split at its current step (Hastie *et al.* 2001).

Mathematically, we start with all the data and consider a splitting variable X_j with a splitting point s and define the two prospective regions as

$$R_1(j,s) = \{X|X_j \leq s\} \quad \text{and} \quad R_2(j,s) = \{X|X_j > s\}. \tag{3.3.2}$$

We consider all possible splitting points for all predictor variables and find the best possible split by minimizing

$$\sum_{i:x_i \in R_1(j,s)} (y_i - \hat{y}_{R1})^2 + \sum_{i:x_i \in R_2(j,s)} (y_i - \hat{y}_{R2})^2, \tag{3.3.3}$$

where $\hat{y}_{Ri}$ is the average response in region i, given splitting point s and variable X_j (Hastie *et al.* 2001). Splitting in this manner is computationally feasible and can in fact be relatively quick if the number of predictor variables is small (Hastie *et al.* 2001). The process is repeated, finding the optimal j and s for the next new set of branches that will minimize the RSS. We conduct this process until some stopping criteria is reached.

If the stopping criteria is relaxed far enough, this method is able to perform perfectly on its training set. However, it is highly undesirable as it will inevitably perform poorly on unseen data. We can say that such a tree is too complex and results in overfitting. It is, therefore, appropriate to introduce a means of penalising the complexity of a tree.

3.3.3 Cost-complexity Pruning

Depending on the size of the tree, the process described above is likely to either over-fit or under-fit the data. Over-fitting occurs when the size of a tree becomes very large and its model is too complex, fitting to the noise of a particular data set rather than the underlying structure of the population. Under-fitting occurs when a tree size is very small and its model too simple. These types of models fail to capture important relationships present in the data (Hastie *et al.* 2001). In order to choose the size of the tree there are two possible solutions.

1. Continue to create new splits if, and only if, the resulting decrease in RSS exceeds some predefined threshold (Hastie *et al.* 2001).
2. Grow a much larger tree, T_0, stopping when some criteria is reached. For example, until no node contains more than n observations. This tree is then pruned down to some smaller subtree (Hastie *et al.* 2001).

The first of the two methods is ill-advised as it does not allow for cases where seemingly meaningless, current splits lead to superior splits later on. The second method is, therefore, preferred. We aim to select a subtree that results in the lowest error rate. We can conduct this test error analysis via cross validation, however, it is computationally infeasible to consider every possible subtree (Hastie *et al.* 2001). The pruning method that solves this problem is known as cost-complexity pruning. A set of subtrees is considered defined by the tuning parameter α (Hastie *et al.* 2001).

For every α there exists a subtree T contained in T_0 such that we minimize

$$\sum_{m=1}^{|T|} \sum_{i:x_i \in R_m} (y_i - y_{R_m})^2 + \alpha |T| \tag{3.3.4}$$

where $|T|$ indicates the number of terminal nodes, R_m is the region corresponding to the m^{th} terminal node and y_{R_m} is the average of the training observations in region R_m, the predicted value (Hastie *et al.* 2001). Inspecting the equation, we can see that when α is 0, T will equal T_0 as there will be no penalty for having a large tree. Hastie *et al.* (2001) describe the α value as a tuning parameter that controls the trade-off between a subtree's complexity and how well it fits the training data. It is evident that as the value for α increases, larger trees are penalised. The method prunes branches in a nested fashion, so we can quite easily generate the sequence of subtrees that result as a function of α (Hastie *et al.* 2001).

It is helpful to summarize this process in a step-wise notation based off a similar summary from James and other (Hastie *et al.* 2001).

1. Determine a training set. Grow a large tree using recursive binary splitting.
2. Stop growing when the subsequent split would lead to less than some pre-defined minimum node size.
3. Use cost-complexity pruning to determine the specific subtrees that result as a function of α.
4. Using K-fold cross validation, divide the training set into K equal subsets.
   (a) Conduct the first three steps on all but the Kth fold.
   (b) Find the mean squared error (MSE) for the prediction from the Kth fold.
5. Choose α and the corresponding subtree such that the average MSE is minimized.

3.3.4 Classification Trees

Classification trees differ from regression trees in that, rather than modelling a quantitative response variable, we now model a qualitative one, taking on classes $1, 2, ..., K$ (Hastie *et al.* 2001). Regression trees find the predicted value for a specific region by taking the mean response in that region. Classification trees, on the other hand, base their predictions on which response class is dominant in a specific region (Hastie *et al.* 2001). For example, if more than 50% of the training set responses in R_i are 'Yes' then region i's classification for any future observation is 'Yes'. We need to adjust the splitting criterion as well as the pruning method.

We grow a classification tree in a similar way to a regression tree in that we use recursive binary splitting. However, we can no longer use RSS as a splitting criterion (Hastie *et al.* 2001). Instead, we use the classification error rate. The classification error rate, E, for a particular region is the number of observations that do not belong to the dominant class in that region

$$E = 1 - \max_k(\hat{p}_{mk}), \tag{3.3.5}$$

where $\hat{p}_{mk}$ is the proportion of training observations of class k in region m (Hastie *et al.* 2001). In practice, however, splitting based on this criterion does not produce good trees and so we consider three different splitting criteria; (1) the Gini Index, (2) the Entropy Error and (3) the Deviance. These measures turn out to be more sensitive to changes in probabilities than the misclassification error rate (James *et al.* 2013).

The Gini Index is a measure of a node's impurity. It is defined by

$$G = \sum_{k=1}^{K} \hat{p}_{mk}\big(1 - \hat{p}_{mk}\big). \tag{3.3.6}$$

When $\hat{p}_{mk}$ is close to zero or one, the Gini Index takes on a small value, indicating the node is made up predominantly of the same class (Hastie *et al.* 2001).

Entropy and the Gini Index are in fact quite similar measurements. Entropy is defined by

$$D = -\sum_{k=1}^{K} \hat{p}_{mk} log\big(\hat{p}_{mk}\big). \tag{3.3.7}$$

Entropy will also take on a small value when $\hat{p}_{mk}$ is close to zero or one, indicating a low node impurity measurement when most observations in a node of the same class (Hastie *et al.* 2001).

The Deviance method assumes that the classification tree is a probability model. We assume that the conditional distribution of the response, given the predictor variables, follows a multinomial distribution (Gray & Fan 2008). Let K be the number of response classes in tree T such that the tree has $|T|$ terminal nodes. The maximum likelihood estimate of the probability of class k conditional on being in terminal node $m \in |T|$ is equal to the proportion of class k in terminal node m (Gray & Fan 2008). This is equal to $\hat{p}_{mk}$. We know that all the probabilities will be strictly positive and will sum to 1 for all $k = 1, 2, ..., K$. If Y_m is the set of categorical responses in the mth region then we say that Y_m is a random sample of size n_m from the multinomial distribution.

$$p(\boldsymbol{y}_m) = \begin{pmatrix} n_m \\ n_{m1} \dots n_{mn} \end{pmatrix} \prod_{k=1}^{K} \hat{p}_{mk}^{n_{mk}} \tag{3.3.8}$$

We can write the likelihood as

$$L = \prod_{m=1}^{M} p(\boldsymbol{y}_m) \propto \prod_{m=1}^{M} \prod_{k=1}^{K} \hat{p}_{mk}^{n_{mk}}. \tag{3.3.9}$$

The deviance associated with region m is then

$$D(m) = -2log(L) = -2 \sum_{m=1}^{M} \sum_{k=1}^{K} n_{mk} log\big(\hat{p}_{mk}\big). \tag{3.3.10}$$

When splitting in this manner we find the split that will best minimize the deviance. This splitting criterion, unlike the Gini Index and Entropy, takes into account the number of observations per region.

Now that we have defined the means by which we will grow a classification tree, we are left with the task of pruning a large tree into a more parsimonious and less complex tree. Any of the methods are appropriate for tree pruning, however, misclassification error is preferable when we are interested in the prediction accuracy of the tree (Hastie *et al.* 2001).

Figures 3.9 and 3.10 have been constructed using the popular `titanic` data set in R. The Figures illustrate how a more complex tree, as in Figure 3.9, might be pruned down to a more simple, tree as in Figure 3.10.

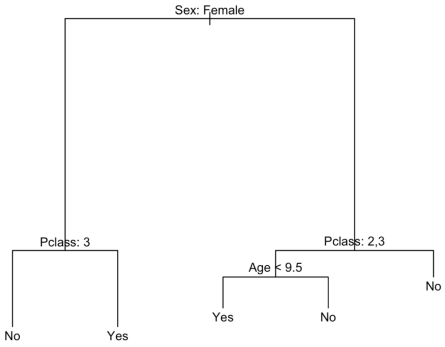


Figure 3.9: Example of a more complex classification tree that uses gender, passenger class and age to predict survival.

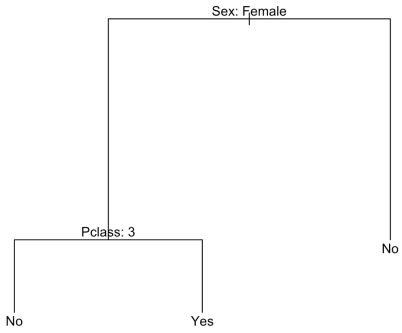


Figure 3.10: Example of a pruned classification tree that uses uses gender, passenger class and age to predict survival.

We have assumed, so far, that our predictor variables are all of a continuous nature. Decisions cater for categorical variables as well (Hastie *et al.* 2001). For example, in Figure 3.10 above, the second split is made based on passenger class, `Pclass`, such that passengers in class three are segmented into the left hand branch and those in classes one and two are segmented into the right hand branch.

We often care about the prediction uncertainty. Analysing Figure 3.10, for example, even though a `Male` is predicted not to survive, how many of the observations that fell in this segment did survive? This is called terminal node purity. It is helpful to create a confusion matrix, $\boldsymbol{C}$, indicating the proportion of incorrect results. We calculate the misclassification rate as

$$\text{Misclassification rate } = 1 - \frac{tr(\boldsymbol{C})}{N} \times 100 \tag{3.3.11}$$

where $tr(\boldsymbol{C})$ is the trace of the confusion matrix and N is the number of total observations.

3.3.5 Advantages and Disadvantages

It can be argued that trees mimic the way that humans think, making them very easy to explain to people who do not have a statistical background without any loss in meaning (Hastie *et al.* 2001). Additionally, trees handle categorical variables well compared to many other methods for which dummy variables are required (Hastie *et al.* 2001).

Although decision trees are simple yet powerful models, a plain decision tree's accuracy does not compare favourably with that of other machine learning approaches (Hastie *et al.* 2001; James *et al.* 2013). Furthermore, they are not robust. Their results change quite drastically when small changes to the training data are made. We are able to improve this error by aggregating many decision trees (Hastie *et al.* 2001).

3.3.6 Bagging

When constructing decision trees, two samples from the same population can produce drastically different results (Hastie *et al.* 2001). Bagging, otherwise known as bootstrapping aggregation, is a technique that helps solve the issue of high sampling variability. Bagging reduces the variance of an estimated prediction function (James *et al.* 2013). Consider the set $Z_1, Z_2, ..., Z_n$ of n independent observations. Suppose they each have variance σ^2. We find the variance for the mean, $\bar{Z}$, of the observations by taking

$$\frac{\sigma^2}{n}. \tag{3.3.12}$$

Taking the mean of a set of observations reduces the variance (Hastie *et al.* 2001). We can stipulate that, if we take many samples from the population and build a model on each sample, we could average the results and increase the prediction accuracy (Hastie *et al.* 2001). Let $\hat{f}^1(x)$, $\hat{f}^2(x)$, ..., $\hat{f}^B(x)$ be a set of models built on B training data sets from some population. Then

$$\hat{f}_{avg}(x) = \frac{1}{B}\sum_{b=1}^{B}\hat{f}^b(x). \tag{3.3.13}$$

In reality, we often do not have multiple samples, or training sets, from the same population. We get around this issue by bootstrapping. The bootstrapping technique is able to create multiple simulated samples by drawing from the initial sample with replacement.

When applying bagging to regression problems, we simply construct B training sets with B regression trees and average the results (Hastie *et al.* 2001). This is slightly different for classification trees. In classification trees we note the class predicted for each tree and make a prediction as the most commonly occurring class (Hastie *et al.* 2001).

3.3.7 Out-Of-Bag Error

When bagging resamples are drawn from the initial training set, on average a resample contains only 2/3 of the original observations, while 1/3 is left unused. These are known as the out-of-bag (OOB) observations (Hastie *et al.* 2001). We are thus left with $\frac{B}{3}$ predictions on average for the ith observation (Hastie *et al.* 2001). A single OOB prediction can be found for the ith observation by finding the average of the predicted responses in the case of regression and the most common in the case of classification (Hastie *et al.* 2001). We can do this for each observation to find the mean squared error (MSE) for a regression problem and the overall classification error for a classification problem. When B is large enough, OOB error is comparable with leave-one-out cross validation (Hastie *et al.* 2001).

3.3.8 Variable Importance

As could be imagined, the bagged model, being a compilation of B tree models, is more difficult to interpret compared to the original single decision tree model. It could therefore be argued that bagging improves the model at the cost of interpretability (Hastie *et al.* 2001). A helpful feature of bagged models, however, is that we can use the splitting criterion to gather a summary of the role each predictor played in the splits. This is known as variable importance. This is done by calculating the decrease in RSS in the regression case due to a split on predictor j. Similarly, for the classification case, we can calculate the decrease in the Gini Index and average over the B models (Hastie *et al.* 2001).

Figure 3.11 illustrates the relative variable importance for each of the predictors in the `Credit` data set when used to construct a bagged model. The importance is calculated by conducting bagging and recording an initial MSE, then permuting the variables one at a time and recording the decrease in mean squared error (MSE). Finally, the importance is calculated as a percentage of the initial MSE. There are many different ways to do this.

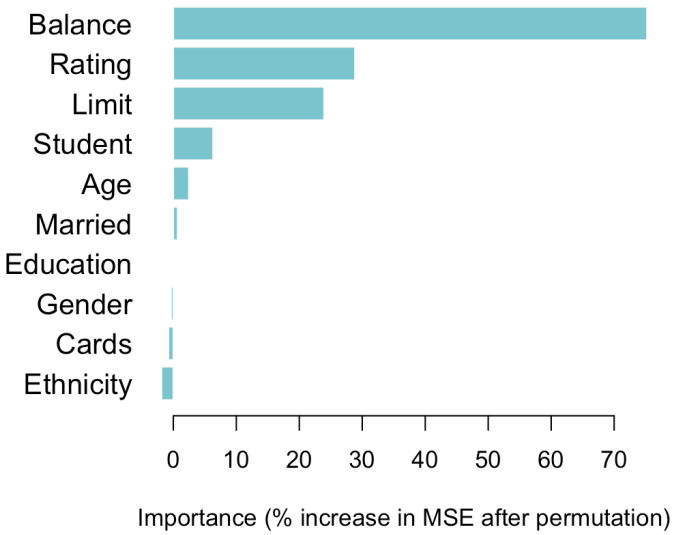


Figure 3.11: Variable importance for the `Credit` data set using a bagging model. `Balance`, `Limit` and `Rating` are the most significant variables in the model building.

3.3.9 Random Forests

Each tree in a bagged model is identically distributed, making the expectation of an average of B trees the same as the expectation of any one of the trees (James *et al.* 2013). This implies that the bias of a single bootstrap tree is the same as that of the individual (bootstrap) trees (James *et al.* 2013). Instead, we look to improve the results by decreasing the variance (James *et al.* 2013). Let us assume we have B identically distributed random variables each with variance σ^2 and correlation ρ. Averaging these random variables we find the variance of the average to be

$$\rho\sigma^2 + \frac{1-\rho}{B}\sigma. \tag{3.3.14}$$

The second term will tend to zero as B tends to infinity, however, the first term is bound by the correlation (James *et al.* 2013). If the B trees were independent, and not correlated, the average of the variance would decrease to

$$\frac{1}{B}\sigma. \tag{3.3.15}$$

Random forests differ from bagging models in that they decrease the correlation between bagged trees in order to decrease the variance.

We continue to build B decision trees using bootstrapping to acquire multiple training samples. In bagging we considered all p variables at every split. When building a random forest, however, we will only consider a random subset, m, of the predictor variables at each split (Hastie *et al.* 2001). Breiman (2001), in his seminal random forest paper, recommended $m = \sqrt{p}$ and $m = p/3$ predictor variables be considered at each split for classification and regression problems respectively.

On the whole, random forests are a powerful technique. They are able to fit complex, higher-order interactions among predictors (Quach 2017). They tend to perform well when the number of predictors surpasses the number of observations. They are also robust to predictor outliers and easily disregard unhelpful predictors (Quach 2017). Unlike regression models, all regression tree models are non-parametric and so do not require that the data be normally distributed or scaled. Missing data in the predictors need not be imputed and the techniques are relatively intuitive to most audiences. Random forests are easy to implement with the `randomForest` package in R (Liaw & Wiener 2002).

Chapter 4

Modelling

This chapter outlines the approaches we used in building our models. It begins with an exploration of our data. It then outlines the development of our random forest models as well as our GAMs. It concludes with an explanation of our procedure for defining pollen categories, as well as our testing methods. All statistical analysis was conducted using R (R Core Team 2020).

4.1 Data Exploration

In this section we aimed to assess the distributions of the response variable, as well as the distributions of the covariates. Further, we aimed to get a sense as to how each variable varies over the year, and how each variable may be related to the response. Data exploration can help us to identify any important patterns in the data, and to assess the strength and nature of the relationships between the covariates and the response. Table 4.1 shows summary statistics for all variables (excluding all vegetation index measures aside from median vegetation index). This table was generated using the `reporttools` package in R (Rufibach 2009).

Variable	**n**	**Min**	$\mathbf{q_1}$	$\widetilde{\mathbf{x}}$	$\bar{\mathbf{x}}$	$\mathbf{q_3}$	**Max**	**s**	**IQR**	**#NA**	**Units**
Pollen Count	1953	0.0	0.0	1.0	4.9	5.0	130.0	10.6	5.0	0	2D count
Veg. Index	1952	0.1	0.2	0.2	0.2	0.2	0.2	0.0	0.0	1	count/m^2
Min. Temp.	1953	1.1	10.0	13.1	12.8	16.1	23.1	4.1	6.1	0	°C
Max. Temp.	1953	11.7	19.0	23.1	23.1	26.8	38.3	5.1	7.8	0	°C
Humidity	1953	32.8	62.7	69.6	69.2	76.1	98.7	9.5	13.5	0	%
Rain	1953	0.0	0.0	0.0	1.8	0.8	71.6	5.1	0.8	0	mm
Wind Speed	1953	9.4	24.1	31.3	31.4	38.9	68.4	10.0	14.8	0	km/hr
Wind Direction	1953	63.5	172.9	190.6	201.8	227.1	337.5	45.8	54.2	0	°
Visibility	1913	1.2	10.0	11.5	12.4	14.6	24.0	3.3	4.6	40	km
Dew Point	1953	1.0	8.8	11.2	11.2	13.9	20.2	3.5	5.1	0	°C

Table 4.1: Number of observations, minimum, maximum, median, mean, quartiles, standard deviation, inter-quartile range, number of missing observations and units, by variable.

We have 1953 pollen count observations from 2011 to 2020. No pollen data were collected from 2015 to 2017 due to insufficient funding. These years were excluded from the analysis. The pollen counts are plotted in Figure 4.1.

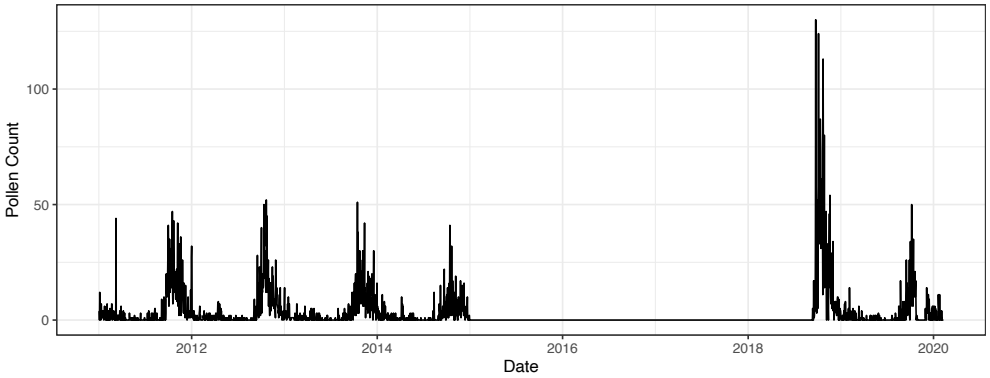


Figure 4.1: Recorded grass pollen counts, from the SAAO in Cape Town, with interrupted data collection from 2015 to 2017.

There is clearly a large degree of variation between years. The pollen season of 2018, for example, shows much higher counts than other seasons. Histograms of all variables are plotted in Figure 4.2.

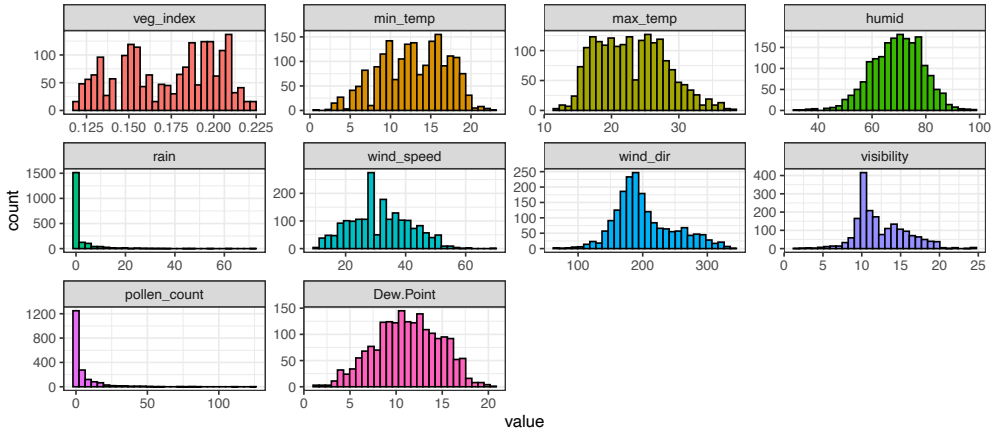


Figure 4.2: Histograms of all numeric variables.

The pollen counts are very clearly right-skewed, as is usually the case with count data. Most pollen counts are close to zero, while a few are much larger. Rain follows a similar right-skewed distribution. Next, the variables are plotted against days since the start of the year, in order to detect any seasonal patterns, in Figure 4.3.

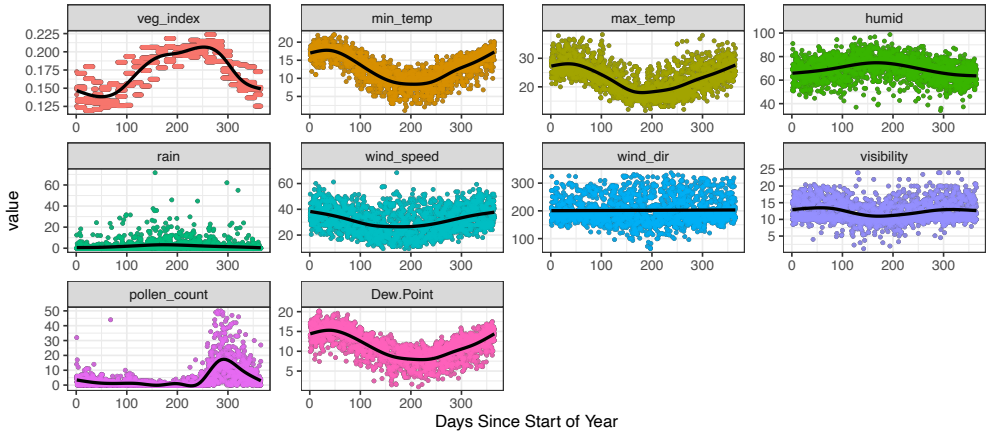


Figure 4.3: Numeric variables plotted against days since start of year (GAM smoothers shown in black).

Clearly most variables appear to vary seasonally. Vegetation Index and Rain are lowest in the summer, while the two temperature variables as well as Dew Point and Wind Speed are lowest in the winter. Pollen count has a strong seasonal pattern, such that we can divide up the year into an In Season period, during which pollen counts tend to be high, and an Out of Season period, during which pollen counts tend to be low. If the day of the year is less than 30 or greater than 240 we can consider that day to be In Season; otherwise we can consider it to be Out of Season. These rough cutoffs have been chosen using visual inspection of the seasonal patterns. This classification scheme is used later on to compare the model's performance on the two periods. Finally, each variable is plotted against pollen count in Figure 4.4.

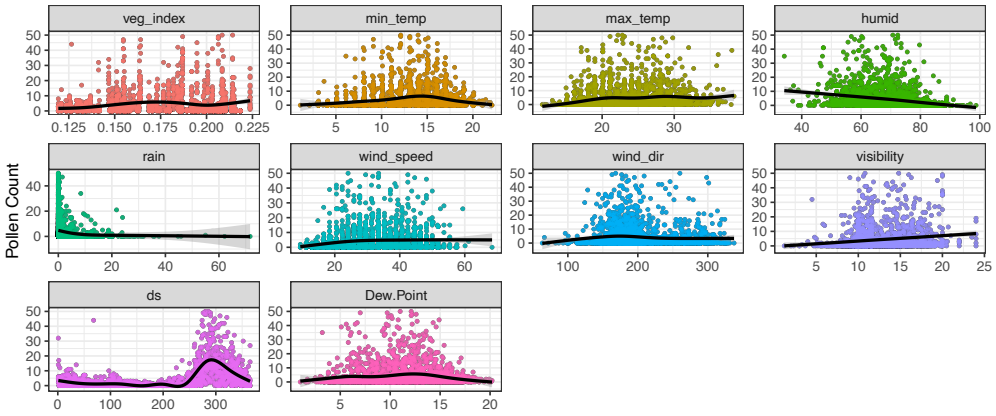


Figure 4.4: Numeric variables plotted against pollen count (GAM smoothers shown in black)

Most scatterplots, with the exception of the plot showing Pollen Count plotted against Days Since, do not show strong relationships. Based on the smooth functions shown in black, we can conclude that most of the relationships are non-linear. Further, we can conclude that Humidity appears negatively related to Pollen Count, while Visibility appears positively related.

4.2 Random Forests Model Building

Random forests are non-parametric, non-linear and are computationally efficient due to their so-called 'embarrassingly parallelizable' nature (Conn *et al.* 2015). There is no strict procedure, as there is with other regression models, on how to go about building an appropriate

random forest. In fact, random forests are often used to aid the variable selection process when building other models (Genuer *et al.* 2010). As such, random forest model building is mainly concerned with hyper-parameter tuning strategies and feature engineering. Although the default parameters are known to perform relatively well, tuning the hyper-parameters can certainly improve the performance (Probst *et al.* 2018). These hyper-parameters include the number of trees, the number of random predictor variables considered at each split, as well as the minimum number of nodes permitted in a tree.

Random forests are able to fit complex interactions with ease and so little manipulation of the predictor set was needed (Quach 2017). Two types of additional features, however, were included to aid the random forest performance: lagged values and temperature anomaly variables. To generate the `max_temp.anom` variable we regressed `max_temp` against `ds` using a GAM and extracted the residuals from this model. This feature measures the difference between the observed temperature and what we would expect given the time of year. A similar method is used to generate `min_temp.anom`.

It is common practice when building random forests to show variable importance plots and partial dependence plots. These plots, however, are important only when a user is interested in the effect of a particular predictor on the outcome: in other words, inference. The goal of this project is that of prediction and not inference. Variable importance plots provide a list of the most important predictors in descending order. This is unhelpful in our case as it is well-known that variable importance plots are biased in the presence of multicolinearity (Conn *et al.* 2015). Partial dependence plots show the nature of the relationship between the outcome and a particular feature (Friedman 2001). For classification-type random forests, partial dependence plots show the probability for a particular category given different predictor values (Molnar 2019). An important requirement for these plots is that of independent predictor variables. A breach of this assumption could lead to plots that are either improbable or impossible (Molnar 2019). As shown in Figure 6.2 many of our variables are highly correlated. Additionally, when lagged variables are added as predictor variables these also inevitably show high levels of multicolinearity. Variable importance plots and partial dependence plots would be not only be unhelpful as we care about prediction, but could also potentially be misleading. These plots will not be investigated.

We use the out-of-bag (OOB) error to select random forest hyper-parameters. Breiman (1996) empirically shows that the OOB estimate is as accurate as using a separate test set equal in size to the training set.

4.3 Hyper-parameter Tuning

4.3.1 Number of trees

The number of trees built in the random forest, T, corresponds to the number of Monte Carlo simulations and must be set by the user (Scornet 2017). Theoretically, it is always better to choose the highest possible value for T that is computationally feasible (Scornet 2017). In practice, however, it is unclear whether the model should build as many trees as is computationally feasible or whether fewer trees may be sufficient and in some cases provide better results (Probst *et al.* 2018). There is literature that argues that a minimum required number of trees can and should be computed. However, this minimum has been shown to vary depending on the model (Latinne *et al.* 2001). Ultimately, the user is left to estimate the number of trees suitable for their use-case. The default value using the `randomForest` package

in `R` is $T = 500$. We found that setting $T = 500$ resulted in an OOB error that did not appear to converge and so used $T = 2\,000$ to ensure convergence for each model.

4.3.2 mtry

`mtry` refers to the number of predictor variables randomly chosen for splitting. The default `mtry` value for classification models is `mtry` $= \sqrt{p}$ where p is the number of predictor variables. It is commonly accepted that the default value for `mtry` yields good predictive performance. However, there is no theoretical justification for this default value and it is rather based on the mathematical analysis presented by Breiman (Scornet 2017; Breiman 2001). Another consideration when setting `mtry` is the computational benefit of considering fewer variables at each split (Scornet 2017). A range of values around the default value are therefore considered for each model's `mtry` parameter while excluding very high `mtry` values to ensure computational efficiency.

4.3.3 Minimum number of nodes

The minimum number of nodes, or `min.node.size`, is the minimum size of a node before the tree is instructed to stop splitting (Scornet 2017). It is therefore a measure of model complexity and can control the depth of the tree. The larger the `min.node.size` the shorter the tree. The recommended min.node.size for classification models is 1, however, results can be improved if this hyper-parameter is tuned (Lin & Jeon 2006). We found that our models performed poorly when using small `min.node.size` values, likely because the trees were overfitting. Instead, values of 5, 10 and 20 were considered for `min.node.size`.

4.3.4 Splitting rule

Although the splitting rule is not strictly a hyper-parameter, it is one of the core properties which characterize a model (Probst *et al.* 2018). We use the Gini Impurity measure at each split.

4.3.5 Tuning Strategies

A grid search was conducted using the range of parameters defined in Table 4.2, where p is the number of input variables. This was conducted using the `ranger` package in `R` (Wright & Ziegler 2017).

Model	p	mtry Range	min.node.size Range
1.1	26	2:9	5,10,20
1.2	52	2:9	5,10,20
1.3	7	2:6	5,10,20
1.4	41	2:9	5,10,20

Table 4.2: Hyper-parameter grid search ranges for each of the models. Models 1.1 to 1.4 are the four random forest models investigated and defined in the subsequent section.

4.4 Models

Four models were considered. Model 1.1 included all variables available from the three data set types: pollen data, weather data and vegetation index data. Model 1.2 then included all variables included in Model 1.1 as well as the lagged features from variables with high variable importance. Model 1.3 was constructed as a very simple model considering only the `ds`

variable as well as lagged pollen counts. Finally, some literature suggested that a user might consider eliminating the less important variables from a model to improve computational efficiency (Speiser *et al.* 2019). We therefore developed Model 1.4, a variation of Model 1.2 which reduces the number of variables to those with more variable importance. The predictor variables used in each model have been captured in Table A.1 in the Appendix.

4.5 GAM Model Building

4.5.1 Response Distribution

Since GAMs are extended GLMs, we can choose to model the response with a non-normal distribution. Count data are integers and non-negative, so it is generally not appropriate to assume that the errors are normally-distributed (Ver Hoef & Boveng 2007). Usually, either Poisson, negative binomial or quasi-Poisson regression models are used to model count data (Ver Hoef & Boveng 2007). One common feature of biological data is overdispersion, which occurs when the variance of a dataset or model is greater than its mean (Ver Hoef & Boveng 2007). When overdispersion is present and the model chosen is unable to account for it, the model's inferences can be invalid. Poisson regression models assume that the mean of the response is equal to its variance, while negative binomial regression models allow for the variance to be greater than the mean (Ver Hoef & Boveng 2007). Therefore, negative binomial regression models can account for overdispersion, while Poisson regression models cannot (Ver Hoef & Boveng 2007). As such, it is appropriate to use a negative binomial distribution to model pollen counts, giving our GAM model the following functional form

$$Y_i \sim \text{NB}(\mu_i, \theta) \tag{4.5.1}$$

$$\text{E}(Y_i) = \mu_i \text{ and } \text{Var}(Y_i) = \mu_i + \frac{\mu_i^2}{\theta} \tag{4.5.2}$$

$$\mu_i = \exp(f(x_1) + \cdots + \beta_0) \tag{4.5.3}$$

where the i-th response value Y_i follows a negative binomial distribution with mean μ_i and shape parameter θ, and the fitted mean μ_i is obtained by exponentiating the linear predictor (Zuur *et al.* 2009).

4.5.2 Variable Selection

Two approaches for variable selection in GAMs are stepwise selection, generally using Akaike's Information Criterion (AIC), and a double penalty approach (Marra & Wood 2011). The usual single penalty approach penalises spline wiggliness, so as $\lambda_j \to \infty$ the j-th smooth function tends to a linear function (James *et al.* 2013). However, we may wish to shrink all coefficients associated with the smooth function of a particular variable to zero. By adding a second penalty we can down-weight even the linear components of smooth functions, effectively resulting in automatic variable selection: all coefficients associated with a the smooth function of a variable can be shrunk down to zero with the addition of a second penalisation term (Marra & Wood 2011). In the R package `mgcv`, double penalty variable selection is implemented by simply adding the `select = TRUE` argument to the `gam` function (Marra & Wood 2011). Marra & Wood (2011) compared numerous approaches for GAM variable selection and found that double penalty and shrinkage approaches were competitive relative to other approaches in terms of variable selection performance. Therefore, we used double penalty variable selection in our GAMs.

4.5.3 Moving Averages

In order to capture information about past values of the covariates and response, moving averages of the covariates and response were used. This information is helpful to a forecasting model: it is helpful to know the pollen levels or the amount of rainfall over the past week when trying to predict pollen counts over the next few days. We used seven day simple moving averages (MA) and exponentially-weighted moving averages (EMA). The EMA terms give more weight to recent lags as opposed to less recent lags, while the MA terms give equal weight to all lags. It seems plausible that knowing it rained yesterday is more useful than knowing it rained a week ago: the EMA terms consider this. The EMA of a series is given as

$$\text{EMA}_t = \beta X_t + \beta(1-\beta)X_{t-1} + \beta(1-\beta)^2 X_{t-2} \dots$$

where β is the smoothing coefficient, which determines how rapidly the weights decay: higher values of β decrease the weights more rapidly (Guthrie 2020). We chose a value of 0.3 for β.

4.5.4 Three Models

Three models were developed: one with EMA terms (Model 2.1), another with MA terms (Model 2.2) and a third model with fewer covariates (Model 2.3). The variables used for each model are tabulated below:

Model 2.1	**Model 2.2**	**Model 2.3**
EMA(Pollen Count)	MA(Pollen Count)	EMA(Pollen Count)
Max Temp, EMA(Max Temp)	Max Temp, MA(Max Temp)	Max Temp
Min Temp	Min Temp	-
Rain, EMA(Rain)	Rain, MA(Rain)	Rain, EMA(Rain)
Wind Speed, EMA(Wind Speed)	Wind Speed, MA(Wind Speed)	-
Veg Index	Veg Index	Veg Index
Wind Dir., EMA(Wind Dir)	Wind Dir., MA(Wind Dir)	-
Days Since	Days Since	-
Dew Point	Dew Point	-
Visibility	Visibility	-

Table 4.3: Three GAMs, and their covariates.

4.5.5 Smooth Functions

The thin-plate spline is the 'ideal smoother', according to Wood (2017). Hence it is the default spline in `mgcv`: its main downside is computational cost (Wood 2017). The cyclic cubic spline ensures that the start and end points of the spline both have the same values (Wood 2017). All variables except Days Since and Wind Direction were modelled using thin plate splines. Because the true effects of Days Since and Wind Direction on pollen count are cyclical (i.e. there should be no discontinuity between the fitted values at day 0 and day 365, or 0° and 365°), both variables were modelled using cyclic cubic splines.

4.5.6 Concurvity

Since the covariates are correlated with each other, these models suffer from concurvity. Concurvity occurs when a smooth function can be well approximated by linear combinations of other smooths (Ramsay *et al.* 2003). This can affect the standard errors of the model coefficients, leading to deflated p-values and potentially invalid inferences Ramsay *et al.* 2003. However, Lieberman & Morris (2014) find that even extreme levels of multicollinearity do not lead to decreases in predictive accuracy. Since our goal is to build a model purely for the

purposes of forecasting pollen levels reliably, the performance of the models on the validation and test sets is our primary concern. As such, we will not report the GAM outputs and plots for the three models, only the performance metrics on the validation and test sets.

4.6 Recursive Forecasting

In order to predict pollen concentrations one to seven days ahead, we used a recursive forecasting strategy. This means that we built models to predict one day ahead pollen concentrations and then use that prediction for forecasting the next day's pollen count. In other words, we have one model that we reuse as we predict further and further out, iteratively replacing observed pollen counts with predicted ones and observed values of weather variables with forecasted ones. The vegetation index data did not include forecasted values, and so the most recent observation is assumed to remain constant for the week ahead. This method is visualised in Figure 4.5.

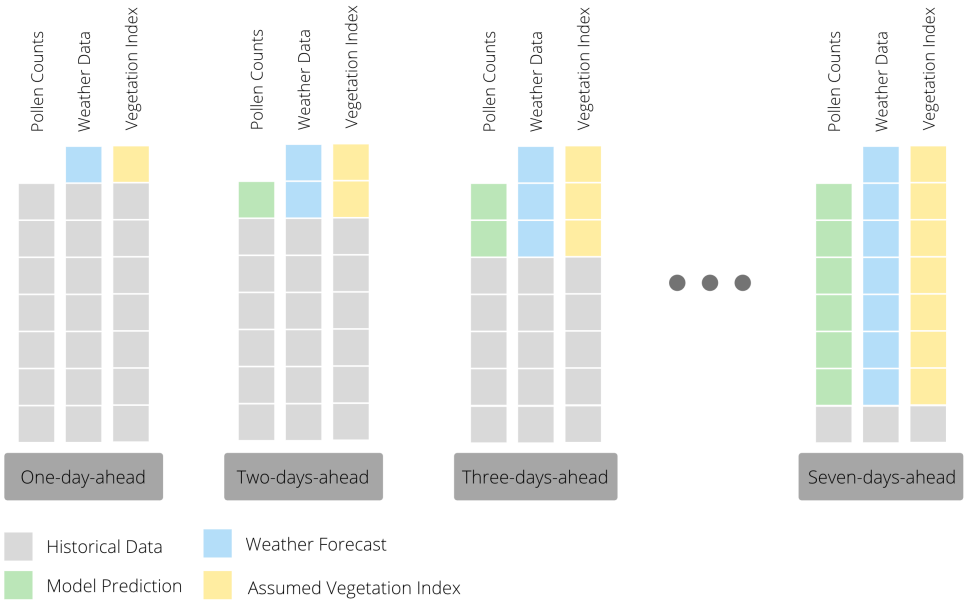


Figure 4.5: Illustration of our recursive forecasting method.

We use past pollen counts to predict pollen one day ahead. However, for subsequent days the observed lagged pollen counts are iteratively replaced with predicted counts, shown in green.

We chose the recursive method over the direct method. The direct forecasting method involves training independent models for each day ahead, and no model predictions are used to make predictions further out (Bontempi *et al.* 2013). The recursive method has the potential to accumulate forecasting errors: since the model at least partly uses forecasted values to make subsequent predictions, any errors can accumulate as the model predicts further out (Bontempi *et al.* 2013). However, the direct method has several drawbacks too, including an inability to consider statistical dependencies between consecutive forecasts (Bontempi *et al.* 2013).

4.7 Categorising Pollen Counts

4.7.1 The Need for Categories

Our models predict pollen counts over the next seven days. However, we believe it is more informative to ultimately predict categories rather than counts, since the end user of our forecast is unlikely to be able to interpret a numeric count prediction. Consequently, instead of reporting the numeric count prediction for each day we converted the daily posterior predictive distributions of pollen counts into probability distributions over five categories, from Very Low to Very High. Similar categories are used on the Real Pollen Count website.

4.7.2 Defining Categories

In order to report categories we needed some meaningful way of assigning categories to observed or forecasted counts. To ensure our categories were reasonably well-balanced, we created categories using four percentiles from the in-season pollen count distribution. These four percentiles were the 20-th, 40-th, 60-th and 80-th percentiles. A histogram of all pollen counts in our dataset (including data from in-season and out of season periods) coloured by category reveals the way we defined categories:

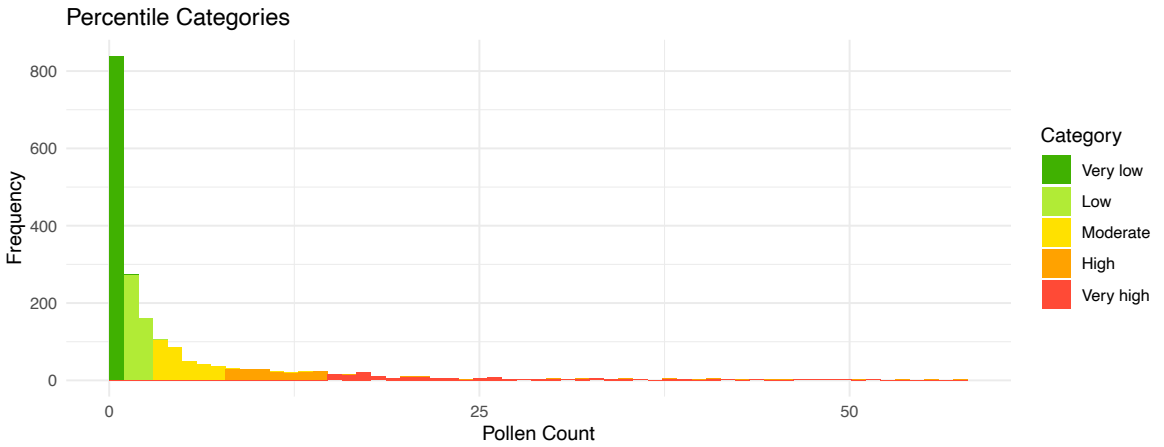


Figure 4.6: Histogram showing pollen categories, for In-Season and Out of Season periods, with large values omitted.

Most counts are classified as Very Low or Low, since outside the pollen season we do not often observe high pollen counts. A consequence of how we defined the classes is that the classes are well balanced during the pollen season and unbalanced outside the pollen season. Another method we considered for assigning categories to counts is a method used by the Real Pollen Count website, developed by Harriet Burge (Berman 2018). This method only uses four categories, and does not classify counts as Very Low. Further, the system used by Burge uses a 3-dimensional pollen count, which is the number of pollen grains per m^2 (Berman 2018). Our classification system uses 2-dimensional pollen counts, which can be converted to 3D concentrations by multiplying the 2D count by 0.72 (Berman 2018). Table 4.4 compares our percentile-based method with the method used by the Real Pollen Count website.

Category	Percentile method	Burge (1992) method
Very Low	count < 1	-
Low	$1 \geq$ count < 3	$0 \geq$ count ≥ 5
Moderate	$3 \geq$ count < 8	$5 <$ count ≥ 20
High	$8 \geq$ count < 14.8	$20 <$ count ≥ 200
Very High	$14.8 \geq$ count	count > 200

Table 4.4: Comparison of two methods for categorising pollen counts.

Since we never observe 3D pollen concentrations greater than 200 grains per m^2 in our data we might conclude that Burge's categories were designed using data collected in a location with a different climate and vegetation, and are unlikely to be appropriate in our case. Further, there is no way of categorising counts as Very Low using this method. For these reasons we used the percentile method for categorising counts.

4.7.3 Probability Distributions

Our seven-day ahead forecast reports probability distributions over the five pollen categories for each of the next seven days. The Random Forest returns predicted class probabilities directly, while the GAM requires us to convert the posterior predictive distribution of counts into a probability distribution of categories. We did this in the following way: for each day (1) we sampled 1000 values from the appropriate negative binomial distribution, with mean equal to the fitted mean pollen count obtained from the GAM regression model, (2) we calculated the proportion of the sampled pollen counts that fall into each category range[1]. The estimate for θ was obtained by setting the `scale` argument in the `gam` function equal to a negative number, which ensures that θ is estimated. An example is shown in 4.7, where 1000 values are sampled from a negative binomial distribution with $\mu = 20$ and $\theta = 1.5$:

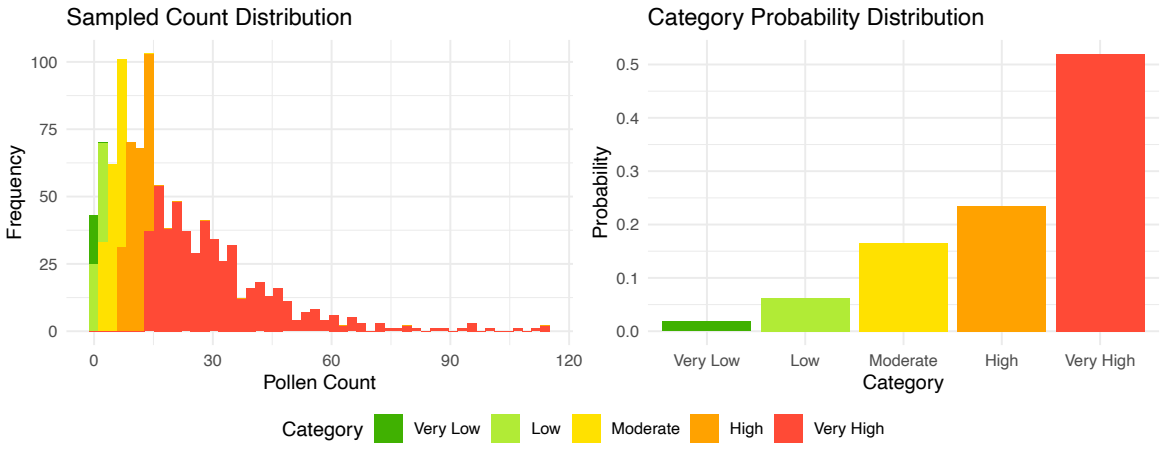


Figure 4.7: 1000 sampled values from NB($\mu = 20, \theta = 1.5$), with proportions calculated using categories defined in Table 4.4

4.8 Testing and Validation

Rigorous model validation and testing are crucial for assessing how well models are likely to perform on previously unseen data (James *et al.* 2013). A fundamental question is how to divide up the data into training, testing and validation sets (Tashman 2000). Given temporal

[1] An alternative method of converting the numeric counts to categories is to use the theoretical quantiles of the negative binomial distribution, and to assign probabilities using the cumulative distribution function.

dependence between observations in time series data, adaptations of the usual randomised holdout procedures are generally used to test and validate time series models (Cerqueira *et al.* 2020).

One simple approach is to split the available data in two, using the first portion for training the model and the second for testing and validation (Cerqueira *et al.* 2020). In our case we could hold out the final year for testing, the penultimate year for validation and the remaining years for training. However, given the large amount of variation in peak pollen concentrations between years our model performance metrics could be biased in one direction or another depending on which years we chose for training and validation. A more robust procedure involves using a rolling or sliding window for the test, train and validation splits (Tashman 2000). In such a procedure several potential training, testing and validation splits are considered. In our case we considered the following potential splits visualised in Figure 4.8 below. This is an adaptation of a diagram by Cochrane (2018).

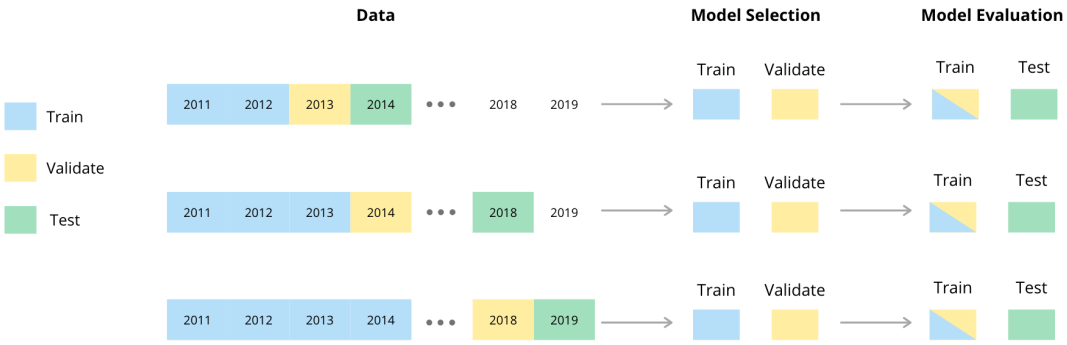


Figure 4.8: Illustration of our approach to split the time series data into three train, validation and test sets adapted from Cochrane (2018).

Model validation can be used for model selection as well as to detect any potential overfitting (James *et al.* 2013). It involves assessing the model performance on some unseen data before committing to a final model. The testing set is then used to evaluate the model's performance (James *et al.* 2013). Model validation is done by training each candidate model first on the years shown in blue and then predicting pollen concentrations for the years shown in yellow. The evaluation metrics are averaged over the three years and then the best performing model is trained on a combination of the years shown in blue and yellow and tested on the years shown in green.

4.9 Evaluation Metrics

4.9.1 Ordinal Forecasts

In order to assess the performance of our models it is essential that we use appropriate evaluation metrics. A good metric should, in our case, account for the ordinal nature of our problem. Many authors erroneously use evaluation metrics better suited to nominal or regression-type models in order to evaluate ordinal classifiers (Gaudette & Japkowicz 2009). In ordinal classification problems there is a meaningful way in which we can order the categories (Gaudette & Japkowicz 2009). In our case, we can meaningfully order the categories from Very Low to Very High. This has implications for what kinds of evaluation metrics we should use, since

not all prediction errors are equally bad given ordinal classes (Gaudette & Japkowicz 2009). For example, on a day with a Very High pollen level we would prefer our forecast to predict High than for it to predict Very Low.

One possible metric could be classification accuracy, which is simply the proportion of predictions that are correct. However, it is clear that such a metric is not well suited to ordinal classification problems, since it treats all errors as equal in magnitude. Two metrics well suited to ordinal classification problems are adaptations of Mean-Squared Error (MSE) and Mean Absolute Error (MAE) to work with confusion matrices (Cardoso & Sousa 2011). Given a $K \times K$ confusion matrix,

$$\text{MSE} = \frac{1}{N} \sum_{r=1}^{K} \sum_{c=1}^{K} n_{r,c}(r-c)^2 \tag{4.9.1}$$

$$\text{MAE} = \frac{1}{N} \sum_{r=1}^{K} \sum_{c=1}^{K} n_{r,c}|r-c| \tag{4.9.2}$$

where $r \in \{1, \ldots, K\}$, representing the rows; $c \in \{1, \ldots, K\}$ representing and $n_{r,c}$ is the number of elements contained in the r-c-th entry in the confusion matrix and N is the number of elements contained in the whole confusion matrix (Cardoso & Sousa 2011). These metrics take on values of 0 for perfect classifiers, and penalise inaccurate predicted classes in proportion to their distance from the true classes (Cardoso & Sousa 2011). Consider these two fictional confusion matrices for ordinal classifiers:

		Observed		
		Low	Med	High
Predicted	Low	1	0	3
	Med	0	1	0
	High	0	0	1

		Observed		
		Low	Med	High
Predicted	Low	1	0	0
	Med	0	1	3
	High	0	0	1

Both have the an accuracy of 0.5, since both classify three observations correctly and three incorrectly. However, the confusion matrices show that the classifier on the left appears to be doing a worse job than the one on the right. The classifier on the left incorrectly classifies three observations as Low that are actually High, while the classifier on the right classifies those same observations as Medium, which is a less severe error. The MSE and MAE can detect this difference in performance, while accuracy cannot.

4.9.2 Probabilistic Forecasts

As our forecast returns probability distributions over five categories of pollen levels, we would like to measure how well these probability distributions match the observed pollen categories. Ideal probabilistic forecasts should have good discrimination, in that they should assign high probabilities to the categories we ultimately observe (Assel *et al.* 2017). Ideal forecasts should also be well-calibrated, in that the model's forecasted class probabilities should agree with the observed distribution of classes over time (Assel *et al.* 2017). One metric that measures discrimination and calibration is the Brier Score (BS), developed by Brier (1950) for evaluating probabilistic weather forecasts. The score is calculated as follows:

$$\text{BS} = \frac{1}{N} \sum_{j=1}^{r} \sum_{i=1}^{N} (f_{ij} - E_{ij})^2 \tag{4.9.3}$$

where f_{ij} is the predicted probability of class j on day i and E_{ij} is a binary variable which is 1 if category j was observed on day i and 0 otherwise (Brier 1950). Lower Brier scores are preferable, and perfect models that always assign a probability of 1 to correct classes are given Brier scores of 0, while the worst possible models are given Brier scores of 2 (Brier 1950).

While the Brier score accounts for the probabilistic nature of our forecast, it unfortunately does not account for the ordinal aspect. Further, while accuracy is not the best metric to use for assessing ordinal or probabilistic forecasts, it is easy to communicate the meaning of accuracy to potential users of our forecast. For these reasons we calculated and reported accuracy, MSE, MAE and Brier scores for each model during testing and validation.

Chapter 5

Results

This chapter contains the results obtained by evaluating our models on the validation and testing years. The models are compared with respect to their performance on the validation years, and the best-performing model is examined in greater detail.

5.1 Model Selection

5.1.1 Random Forests

The out-of-bag (OOB) estimate attained when training a random forest is as accurate as using a separate test set equal in size to the training set (Breiman 1996). Therefore, it would not usually be necessary to use a separate validation set for choosing between competing random forests. However, since we need to compare of the random forest and the GAM with respect to validation set performance, we used the same validation set for both model classes.

A brief aide-mémoire on the models: Model 1.1 is built using all unaltered predictors, Model 1.2 then includes a series of lagged variables as well as the predictors from 1.1, Model 1.4 removes unhelpful predictors identified in 1.2 and finally Model 1.3 is the most simple model.

The four random forest models were tested on the validation sets as defined in Section 4.8. Measures for accuracy, mean square error (MSE), mean absolute error (MAE) and the Brier score were calculated for each of the years 2013, 2014 and 2018, and averaged over the three sets. These results are captured in Table 5.1 and are separated into Total, In Season and Out of Season. No Out of Season data are available for 2018. For this reason, Out of Season results are obtained from the years 2013 and 2014, while In Season results are obtained from the years 2013, 2014 and 2018. Further, Total results are obtained by averaging over all three validation set years, even though 2018 only contains In Season observations.

	Model	Classification Accuracy	MSE	MAE	Brier
In Season	**1.1**	0,266	2,292	1,159	0,803
	1.2	0,307	1,633	0,972	0,746
	1.3	0,365	1,589	0,921	0,762
	1.4	0,296	1,765	1,016	0,745
Out of Season	**1.1**	0,593	0,645	0,484	0,502
	1.2	0,611	0,598	0,457	0,483
	1.3	0,540	0,863	0,586	0,582
	1.4	0,621	0,581	0,445	0,487
Total	**1.1**	0,483	1,116	0,697	0,616
	1.2	0,492	1,007	0,663	0,588
	1.3	0,489	1,080	0,686	0,634
	1.4	0,489	1,046	0,675	0,590

Table 5.1: Random Forest validation metrics showing average classification accuracy, MSE, MAE and Brier score measure for each In Season, Out of Season and Total across the four models. Highlighted cells show the best achiever in each category.

Our goal is to choose the best-performing model, with a special emphasis placed on In Season performance. The results are mixed. If we inspect the In Season results alone, Model 1.3 outperforms the other models across all metrics other than the Brier score. We can thus conclude that Model 1.3 is the best model for In Season prediction. Model 1.4 performs the best Out of Season and Model 1.2 performs the best overall.

We ultimately care most about a model's ability to predict In Season. Although Model 1.3 is not the highest achieving model across each period, it is the preferred model for In Season and so will be used going forward.

5.1.2 GAMs

To recap, Model 2.1 includes EMA terms, 2.2 includes MA terms, and 2.3 is a simpler model containing only one EMA term. The average validation year performance metrics are shown in Table 5.2.

Period	Model	Classification Accuracy	MSE	MAE	Brier
In Season	**2.1**	0.416	1.246	0.774	0.652
	2.2	0.398	1.414	0.847	0.694
	2.3	0.413	1.299	0.796	0.686
Out of Season	**2.1**	0.626	0.444	0.396	0.469
	2.2	0.607	0.471	0.418	0.469
	2.3	0.603	0.500	0.430	0.480
Total	**2.1**	0.517	0.839	0.593	0.567
	2.2	0.509	0.907	0.616	0.575
	2.3	0.490	0.868	0.620	0.574

Table 5.2: GAM validation metrics showing average classification accuracy, MSE, MAE and Brier score measure for each In Season, Out of Season and Total across the three models. Highlighted cells show the best achiever in each category.

Model 2.1 has the best accuracy In Season and Out of Season, and consequently has the best accuracy overall. The more informative metrics also favour Model 2.1, which has the

lowest MSE, MAE and Brier score In Season. Overall, while none of the three models vastly outperform the others, Model 2.1 is the best performing model overall on the validation years, and will be used going forward.

5.2 Comparing the Best Two Models

The random forest model 1.3 and GAM 2.1 are identified as the best performing models for each model type. We intend on using a single model in the pollen forecaster and so must decide which of these two models to use. Figure 5.1 shows a comparison of their results. Accuracy, MSE, MAE and the Brier score are compared for In Season, Out of Season and Total.

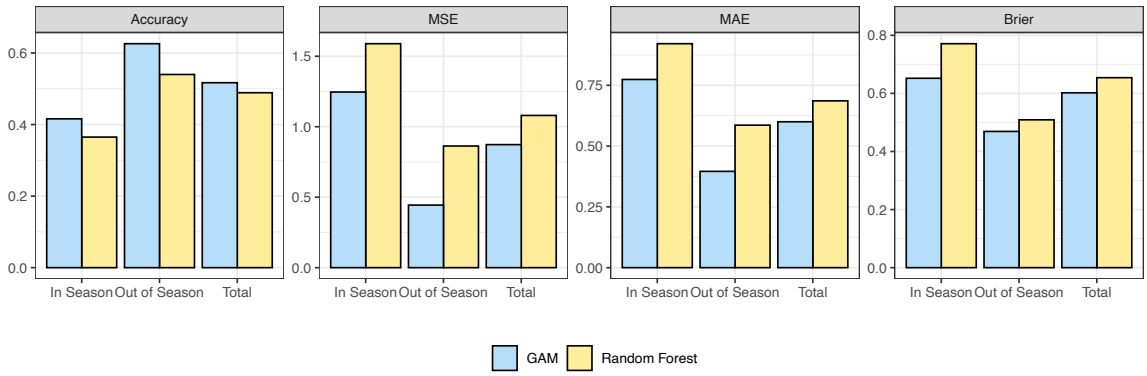


Figure 5.1: Comparison of validation set performance of random forest and GAM.

The GAM model outperforms the random forest across all four metrics and in all periods. The GAM achieves a greater classification accuracy, so it predicts the correct category more often than the random forest does. The GAM has a lower MSE and lower MAE and so when the categorical prediction is incorrect, the GAM predicts categories closer to the true category more often than the random forest. The GAM also has a lower Brier score and so is better skilled in its probabilistic forecasting ability in terms of predicting whether or not a category occurred. Importantly, the GAM significantly outperforms the random forest In Season for each of the MSE, MAE and Brier score.

5.3 Model Evaluation

Model 2.1, which is the GAM with exponentially-weighted moving average terms, performed the best on the validation sets. Our next step was to assess whether it was predicting reasonable and helpful results. Model 2.1 was evaluated on the testing years of 2014, 2018 and 2019, and metrics were averaged over these three years. These results are shown in Table 5.3.

Period	Accuracy	MSE	MAE	Brier
In season	0.395	1.303	0.818	0.674
Out of Season	0.636	0.462	0.393	0.463
Total	0.512	0.837	0.609	0.561

Table 5.3: Test set performance of GAM.

Although accuracy can be interpreted as a stand alone measure of model performance, as explained in Section 4.9, it is not an appropriate metric for ordinal classification. We can,

however, get a sense of how well our model is performing by comparing MAE to its possible range as well as considering what the MAE measure illustrates. MAE can take on a value from 0 to $1 - J$, where J is the number of categories. In our case the MAE is able to take on values from 0 to 4. The best GAM model has an average Total MAE of 0.609 and so seems to perform relatively well. Consider a model that consistently predicts either one category below or one category above the correct category. Using Equation 4.9.2 such a model would achieve an MAE of 1. Therefore, an MAE of 0.609 implies that the best GAM is off by less than one category on average.

From Equation 4.9.3 the Brier score can take on values from 0 to 2, with 0 being the best possible score. A Total average Brier score of 0.561 is therefore relatively good. Lastly, because the MSE is a squared metric it is able to identify whether a prediction is drastically different to the observation when compared to the MAE. Model 2.1 has a Total average MSE and MAE of 0.837 and 0.606 respectively. This is a small difference and implies that the the model does not predict far from the correct category on average.

5.4 Checking for Autocorrelation

It is important that the residuals are stationary, and that no autocorrelation remains. If autocorrelation remained in the model residuals, we would consider using a Generalised Additive Mixed Model (GAMM), which would allow us to model the correlation structure in the residuals (Wood 2017). We fitted Model 2.1 to the entire dataset and plotted the deviance residuals as well as an Auto Correlation Function (ACF) plot of the deviance residuals. These plots are shown below:

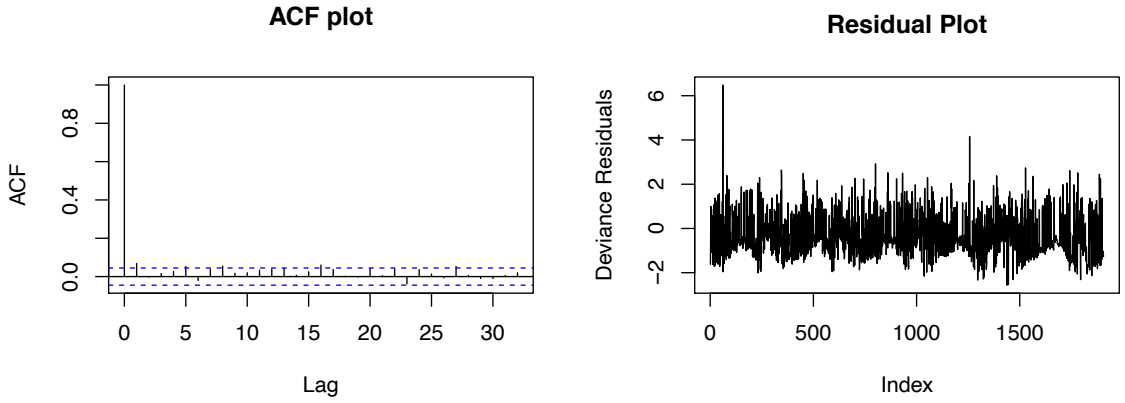


Figure 5.2: Deviance residuals plotted over time, with corresponding ACF plot.

The residual plot shows no clear patterns or trends in the residuals: they appear to be stationary. The ACF plot for lags 0 to 35 confirms that the residuals appear to be independent. This is confirmed with a Durbin-Watson test, showing no significant lag-1 autocorrelation in the residuals ($p = 0.14$). This was calculated using the `lmtest` package in R (Zeileis & Hothorn 2002). The inclusion of moving averages of lagged response and covariate values appears to have resulted in stationary residuals, and modelling the autocorrelation using a GAMM will not be necessary.

Chapter 6

Shiny App

6.1 Motivation and summary

We originally aimed to make the weekly pollen forecasts available on The Real Pollen Count website. The UCT Lung Institute would have control of the forecast. This means that the forecast would need to be updated on a weekly basis, and it must not require extensive RStudio knowledge on the part of the user. In an ideal world the process would be fully automated: the user would simply need to upload the past week's pollen counts and a workflow would perform the necessary steps to update the website without further need for user intervention. However, we did not manage to develop a fully-automated workflow.

Our semi-automated workflow enables us to use current data to generate a 7-day ahead forecast of pollen counts, and to display this forecast on a website. The basics of the workflow are as follows: (1) the data are obtained from four sources using an R script, (2) this script uses the data to output a .csv file containing the next seven days of pollen forecasts, (3) the .csv file is then uploaded to GitHub, and (4) a Shiny app uses the GitHub file to make the forecast available online. These steps are shown using a flowchart:

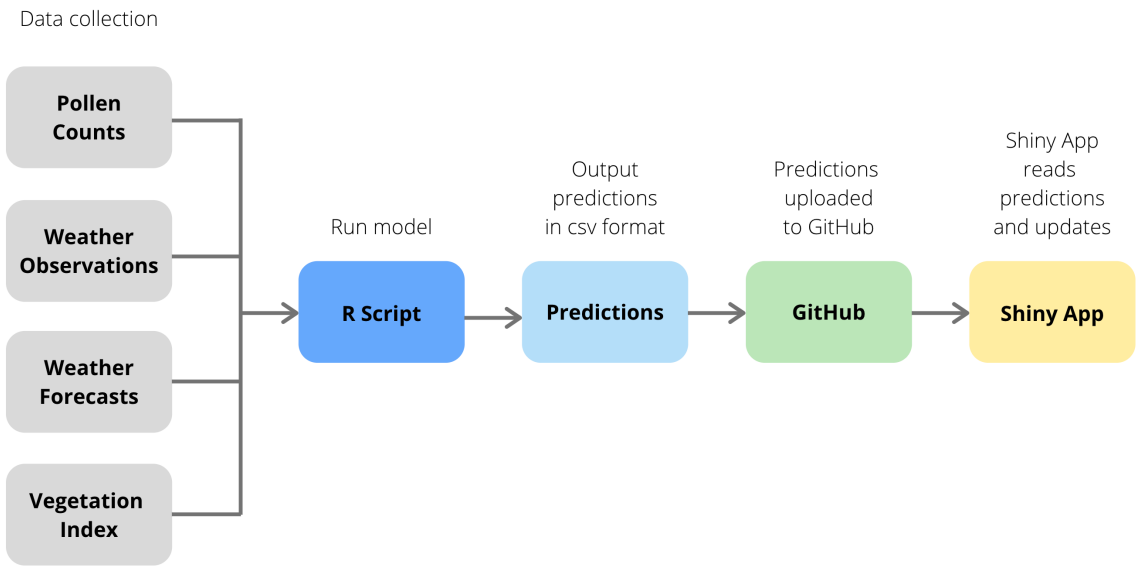


Figure 6.1: Flowchart of workflow.

6.2 Data collection

The first part of the workflow is data collection. The three different types of data need to be collected each week;

1. Pollen counts from the past week.
2. Historical weather data from the past week and weather forecasts for a week ahead.
3. Vegetation Index values from the past week.

The last seven days of pollen counts are stored in a .csv file on a GitHub repository. An R script downloads the .csv file from GitHub and stores it as a data frame. This file can be updated each week using GitHub's edit functionality. This code could potentially be adapted to download pollen count data from other online sources, like Google Sheets or Dropbox.

The historical and forecasted weather data, powered by the Visual Crossings Weather API, collected as described in Section 1.4.2. Each week a get request is made to the Visual Crossing Weather API for the various weather features used in the forecast. This request and data capturing is completely automated once the user has run the R script.

The vegetation index data are stored and accessed just like the pollen counts, using a .csv file on GitHub. The most recent vegetation index value must be accessed manually each week, using NASA's AppEEARS portal, and then entered manually into the GitHub file. The specific data product is the Terra MODIS Vegetation Index 250m 16-day, with the code MOD13Q1.006. The sample area is the same as used previously in this paper. This process is somewhat lengthy, and an improvement to our workflow would involve entirely automating the vegetation index data entry using an API.

6.3 Generating forecasts

Once the necessary data are downloaded, an R script generates the seven day ahead forecast and outputs a .csv file, which is then uploaded to GitHub. This file is then read by a Shiny App.

6.4 Shiny App

Shiny is an open source R package. It allows users to create stylish and powerful web applications using R. We chose to build our interface using Shiny because it is easy to use and has access to all of R's suite of plotting functionality. Shiny Apps run an R Script each time the web page is reloaded. Consequently, the new forecast is automatically downloaded from GitHub by the Shiny App each time the web page is reloaded. This means that the visualisation updates when new data are made available on GitHub. The interface visualises our most recent seven-day-ahead pollen forecast. This allows allergy sufferers to appropriately plan outdoor activities, as well as how much medication they may need to take.

Our aim was to create an interface that was informative, clear and attractive. We wanted our interface to be relatively information-dense but not overwhelming or hard to understand. Some ways of visualising our forecast might be easy for a statistician to understand but not for a layperson. The model should be attractive to the user and match the style of the current Real Pollen Count website.

Below is a screenshot of our completed Shiny app. A link to the app can be found here. The information displayed by our Shiny App is fictional and for illustration purposes only, since the pollen trap was still not functional at the time of writing.

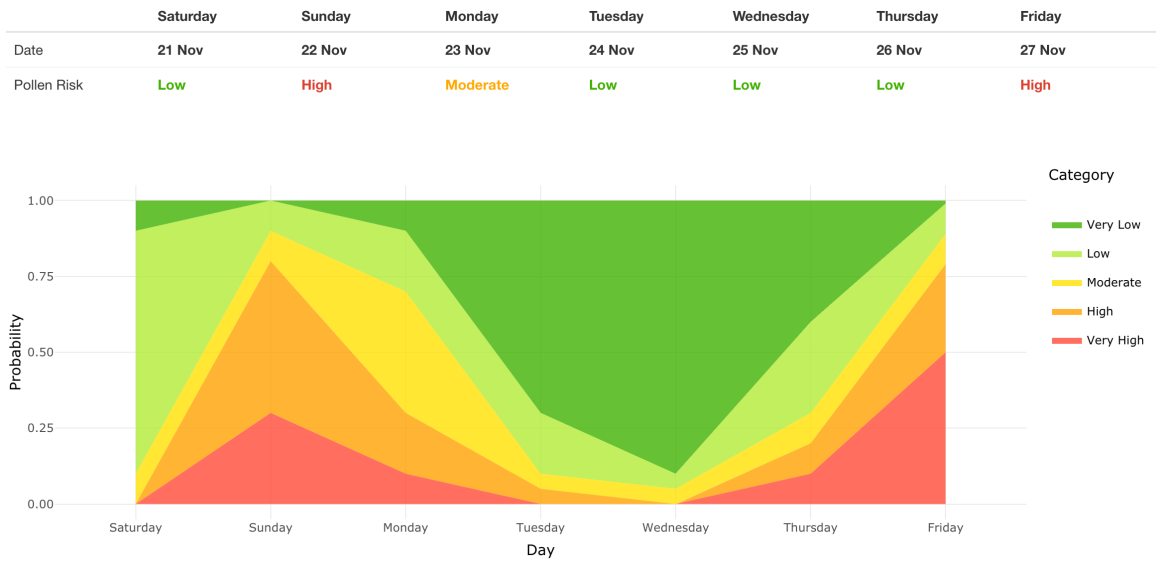


Figure 6.2: Screenshot of Shiny App interface.

The first part of our interface shows a summary of the forecasted level of pollen risk for each of the next seven days. We collapsed the five categories into three: Low, Moderate and High, and the app reports the most likely of the three. This is a simple metric that can tell the user most of what she needs to know at a glance. The ways in which we calculated the three new simplified categories are tabulated below:

Simplified Category	**Interpretation**
Low Risk	Pr(Low or Very Low)
Moderate Risk	Pr(Moderate)
High Risk	Pr(High or Very High)

Table 6.1: Creating simplified categories.

The second part of the Shiny app shows the user more information. Rather than reporting only the single most likely category it shows the probability distribution over five categories for each day. This is displayed using an area chart. We believe an area chart is preferable to a series of seven bar plots because it more clearly shows how the probability of each category changes over time. It also uses space more efficiently, because it shows the same information in a single plot.

Chapter 7

Limitations and Future Research

7.1 Data

We suspect that there is some element of measurement error in the pollen data owing to its trapping and collection method. There have been alterations to the pollen trap over the years, and the pollen is counted by hand by different people. Neither of these variations are documented in the data and might add to the variability seen over a period.

The fact that the pollen data are collected on a weekly basis presented two drawbacks relating to model accuracy and model functionality. Ideally, the model would be re-run each day, incorporating the most recent pollen counts, historical weather and vegetation index data. Without this daily pollen update we can make one seven day ahead prediction a week. For example, if we were to release the seven-day-ahead forecast on a Sunday, the forecast would not update until the following Sunday, when new pollen count data were made available. In this case, a user would only effectively have a two-day-ahead forecast on Friday, and this two-day-ahead forecast would not be as reliable as the forecasts for earlier days.

Although some missing data were present in our input variables, no method for imputing missing values was investigated and these were simply omitted. A published forecast would need to be able to appropriately manage any missing data and so this is a potential area of investigation.

Finally, we learned around the end of September 2020 that the pollen trap had stopped working. This meant that we were unable to have a live forecast available on our Shiny app.

7.2 GAMs and Random Forests

To avoid autocorrelation in our GAM residuals we created rolling moving averages of lagged values of the response and covariates. While we found this approach successful, it may be worth investigating the use of Generalised Additive Mixed Models as a different way of dealing with residual autocorrelation.

The random forests were built using the `randomForest` packing in R. This function uses the familiar k-fold cross validation technique when finding the optimal hyper-parameters. Although an appropriate training-test split was identified and applied as in Section 4.8 for validation, the k-fold cross validation used when training the models pays no attention to the temporal effects and dependencies in the data (Bergmeir & Benítez 2012). More appropriate cross validation techniques, similar to those used in Section 4.8, could be explored to potentially improve the random forest results.

7.3 Many-period-ahead forecasting

Our short-term forecast uses a recursive technique as defined in Section 4.6. This means that the same model is used to forecast one day out as to forecast seven days out. Additionally, predictions are fed forward and used to predict the following day's pollen count. A similar technique known as the direct-recursive method modifies this approach by constructing different models for each day-ahead prediction (Grigorievskiy *et al.* 2014). This method allows us to have different variance estimates for each model which may be more appropriate.

Another consideration when performing multi-period-ahead forecasting is that as a model predicts further out, the confidence with which it forecasts should decrease. A Bayesian-inspired, posterior predictive sampling technique could be considered, such that the posterior distribution of the predicted count is sampled from many times and passed into the subsequent day-ahead model (Thompson & Miller 1986). The general idea is to increase the variance as the number of days ahead increases.

7.4 Different Models, Pollen Types and Locations

Our analysis only considered random forests and GAMs, while many other suitable model classes are available and could be considered. Two methods we believe may yield promising results include gradient boosting and neural networks. Both are highly flexible predictive model classes, which could potentially improve on the GAM's performance. Further, our analysis only considered grass pollen in Cape Town, while data are available for other pollen types and locations. Extending the forecast beyond only predicting grass pollen in Cape Town would make the forecast useful to more people.

Chapter 8

Conclusions

We set out to create a reliable, accessible seven day ahead forecast for grass pollen in Cape Town. Such forecasts can be helpful for allergy sufferers, since having a good sense of the level of pollen over the next few days can help patients to plan their time outdoors appropriately. Reliable forecasts should produce results that do not tend to differ substantially from the true pollen level. Accessible forecasts should produce results that are widely available and easy to understand. To date, such forecasts do not exist for pollen levels in Cape Town.

In order to ensure our forecasts were reliable, we needed to use appropriate methods, as well as enough of the right data. We used pollen count data collected at the SAAO by researchers at the UCT Lung Institute, collected from 2011 to 2014, and then from 2018 to 2020. In addition, we used historical observations of weather variables collected by Visual Crossings Weather, as well as a historical Vegetation Index variable collected by NASA's MODIS satellite. Appropriate models need to account for the count aspect, seasonality and temporal autocorrelation present in pollen count data. Two model classes that met these criteria were GAMs and random forests, both being flexible machine learning methods. Using these two model classes to predict pollen counts, we found that a GAM using exponentially-weighted moving average terms performed the best among all the models we considered, achieving an ordinal one day ahead MSE of 0.837 and MAE of 0.606 on unseen data.

Further, we created a Shiny app, as well as the necessary R workflow, so that our forecast could potentially be made available online and updated every week. We designed the interface in order to be informative and easy for a non-statistician to understand. We also predicted pollen categories rather than counts, since most people won't be able to interpret a numeric count easily. Unfortunately, since the pollen trap was not functional at the time of completing this project, we do not currently have a real-time forecast available online.

While we found success with our approach, we identified several limitations and potential areas for future research. One limitation is the fact that we only forecasted grass pollen for Cape Town, while data for other pollen types and locations are available. We hope that future researchers will be able to improve on our forecast and extend it to more pollen types and locations. Ultimately, we hope that such a forecast will be made available online, updating each week and improving the lives of pollen allergy sufferers.

Bibliography

1. Aboulaich, N. *et al.* Effect of meteorological parameters on Poaceae pollen in the atmosphere of Tetouan (NW Morocco). *International journal of biometeorology* **57,** 197–215 (Jan. 2013).
2. Ajikah, L., Neumann, F. H., Berman, D. & Peter, J. Aerobiology in South Africa: A new hope! *South African Journal of Science* **116.** https://doi.org/10.17159/sajs.2020/8112 (July 2020).
3. AppEEARS Team. *Application for Extracting and Exploring Analysis Ready Samples (AppEEARS)* (NASA EOSDIS Land Processes Distributed Active Archive Center (LP DAAC), 2020).
4. Assel, M., Sjoberg, D. & Vickers, A. The Brier score does not evaluate the clinical utility of diagnostic tests or prediction models. *Diagnostic and Prognostic Research* **1** (Dec. 2017).
5. Bastl, K., Berger, U. & Bastl, M. Evaluation of Pollen Apps Forecasts: The Need for Quality Control in an eHealth Service. *Journal of Medical Internet Research* **19,** e152 (May 2017).
6. Bergmeir, C. & Benítez, J. M. On the use of cross-validation for time series predictor evaluation. eng. *Information sciences* **191,** 192–213. ISSN: 0020-0255 (2012).
7. Berman, D. *Variations in Pollen and Fungal Air Spora* PhD thesis (University of Cape Town, 2018).
8. Biau, G. & Scornet, E. A random forest guided tour. **25,** 197–227. https://doi.org/10.1007/s11749-016-0481-7 (2016).
9. Bloch, J. *How to Design a Good API and Why It Matters* in *Companion to the 21st ACM SIGPLAN Symposium on Object-Oriented Programming Systems, Languages, and Applications* (Association for Computing Machinery, Portland, Oregon, USA, 2006), 506–507. ISBN: 159593491X. https://doi.org/10.1145/1176617.1176622.
10. Bontempi, G., Ben Taieb, S. & Le Borgne, Y.-A. in (Jan. 2013). ISBN: 978-3-642-36317-7.
11. Breiman, L. *OUT-OF-BAG ESTIMATION* in (1996).
12. Breiman, L. Random Forests. eng. *Machine learning* **45,** 5–32. ISSN: 0885-6125 (2001).
13. Brier, G. W. Verification Of Forecasts Expressed In Terms Of Probability. *Monthly Weather Review* **78,** 1–3. https://doi.org/10.1175/1520-0493(1950)078%3C0001:vofeit%3E2.0.co;2 (Jan. 1950).
14. Campbell, I., McDonald, Flannigan, M. & Kringayark, J. Long-distance transport of pollen into the arctic [9]. *Nature* **399,** 29–30 (May 1999).
15. Canonica, G. W. *et al.* Sub-lingual Immunotherapy: World Allergy Organization Position Paper 2009. eng. *Allergy (Copenhagen)* **64,** 1–59. ISSN: 0105-4538 (2009).
16. Cardoso, J. & Sousa, R. Measuring the Performance of Ordinal Classification. *IJPRAI* **25,** 1173–1195 (Dec. 2011).

17. *Climate Change at the City Scale* (eds Cartwright, A., Parnell, S., Oelofse, G. & Ward, S.) https://doi.org/10.4324/9780203112656 (Routledge, June 2012).

18. Cerqueira, V., Torgo, L. & Mozetič, I. Evaluating time series forecasting models: an empirical study on performance estimation methods. *Machine Learning* **109,** 1–32 (Nov. 2020).

19. Cochrane, C. *Time Series Nested Cross-Validation* (2018). https://towardsdatascience.com/time-series-nested-cross-validation-76adba623eb9.

20. Conn, D., Ngun, T., Li, G. & Ramirez, C. Fuzzy Forests: Extending Random Forests for Correlated, High-Dimensional Data. eng (2015).

21. Damialis, A. *et al.* Estimating the abundance of airborne pollen and fungal spores at variable elevations using an aircraft: How high can they fly? *Scientific Reports* **7,** 1 (Mar. 2017).

22. De Weger, L., Beerthuizen, T., Hiemstra, P. & Sont, J. Development and validation of a 5-day-ahead hay fever forecast for patients with grass-pollen-induced allergic rhinitis. *International journal of biometeorology* **58** (June 2013).

23. Devadas, R. *et al.* Dynamic ecological observations from satellites inform aerobiology of allergenic grass pollen. *Science of The Total Environment* **633,** 441–451. ISSN: 0048-9697. http://www.sciencedirect.com/science/article/pii/S0048969718309501 (2018).

24. Didan, K. *MOD13Q1 MODIS/Terra Vegetation Indices 16-Day L3 Global 250m SIN Grid V006* 2015. https://lpdaac.usgs.gov/products/mod13q1v006/.

25. Erbas, B., Chang, J.-H., Newbigin, E. & Dhamarge, S. Modelling atmospheric concentrations of grass pollen using meteorological variables in Melbourne, Australia. eng. *International journal of environmental health research* **17,** 361–368. ISSN: 0960-3123 (2007).

26. Fitter, A. H. & Fitter, R. S. R. Rapid Changes in Flowering Time in British Plants. *Science* **296,** 1689–1691. ISSN: 0036-8075. eprint: https://science.sciencemag.org/content/296/5573/1689.full.pdf. https://science.sciencemag.org/content/296/5573/1689 (2002).

27. Friedman, J. Greedy function approximation: A gradient boosting machine. eng. *The Annals of statistics* **29,** 1189–1232. ISSN: 0090-5364 (2001).

28. Gaudette, L. & Japkowicz, N. *Evaluation Methods for Ordinal Classification* in. **5549** (May 2009), 207–210. ISBN: 978-3-642-01817-6.

29. Genuer, R., Poggi, J.-M. & Tuleau-Malot, C. Variable selection using random forests. eng. *Pattern recognition letters* **31,** 2225–2236. ISSN: 0167-8655 (2010).

30. Ghitarrini, S., Tedeschini, E. & Frenguelli, G. *Phenological analysis of grasses (Poaceae) as a support for the dissection of their pollen season in Perugia (Central Italy).* in (July 2016).

31. Gray, J. B. & Fan, G. Classification tree analysis using TARGET. eng. *Computational statistics data analysis. Computational Statistics Data Analysis* **52,** 1362–1372. ISSN: 0167-9473 (2008).

32. Grigorievskiy, A., Miche, Y., Ventelä, A.-M., Séverin, E. & Lendasse, A. Long-term time series prediction using OP-ELM. eng. *Neural networks* **51,** 50–56. ISSN: 0893-6080 (2014).

33. Guthrie, W. F. *NIST/SEMATECH e-Handbook of Statistical Methods (NIST Handbook 151)* en. 2020. https://www.itl.nist.gov/div898/handbook/.

34. Haberle, S. G. *et al.* The Macroecology of Airborne Pollen in Australian and New Zealand Urban Areas. *PLoS ONE* **9** (ed Bohrer, G.) e97925. https://doi.org/10.1371/journal.pone.0097925 (May 2014).

35. Hastie, T., Tibshirani, R. & Friedman, J. *The Elements of Statistical Learning* (Springer New York Inc., New York, NY, USA, 2001).

36. James, G., Witten, D., Hastie, T. & Tibshirani, R. *ISLR: Data for an Introduction to Statistical Learning with Applications in R* R package version 1.2 (2017). https://CRAN.R-project.org/package=ISLR.

37. James, G., Witten, D., Hastie, T. & Tibshirani, R. *An Introduction to Statistical Learning: with Applications in R* https://faculty.marshall.usc.edu/gareth-james/ISL/ (Springer, 2013).

38. Joanne E. Taylor, S. L. & Crous, P. W. Biodiversity in the Cape Floral Kingdom: fungi occurring on Proteaceae* *Paper presented at the Asian Mycological Congress 2000 (AMC 2000), incorporating the 2nd Asia-Pacific Mycological Congress on Biodiversity and Biotechnology, and held at the University of Hong Kong on 9-13 July 2000. *Mycological Research* **105,** 1480–1484. ISSN: 0953-7562. http://www.sciencedirect.com/science/article/pii/S095375620862033X (2001).

39. Latinne, P., Debeir, O. & Decaestecker, C. *Limiting the Number of Trees in Random Forests* in. **2096** (July 2001), 178–187. ISBN: 978-3-540-42284-6.

40. Liaw, A. & Wiener, M. Classification and Regression by randomForest. *R News* **2,** 18–22. https://CRAN.R-project.org/doc/Rnews/ (2002).

41. Lieberman, M. & Morris, J. The Precise Effect of Multicollinearity on Classification Prediction. **40,** 5–10 (Jan. 2014).

42. Lin, Y. & Jeon, Y. Random Forests and Adaptive Nearest Neighbors. *Journal of the American Statistical Association* **101,** 578–590. ISSN: 01621459. http://www.jstor.org/stable/27590719 (2006).

43. Marra, G. & Wood, S. Practical variable selection for generalized additive models. *Computational Statistics Data Analysis* **55,** 2372–2387 (July 2011).

44. Miura, T., Huete, A., Didan, K., Van Leeuwen, W. & Yoshioka, H. *An assessment of the MODIS vegetation index compositing algorithmusing quality assurance flags and Sun/view angles* in. **2** (Feb. 2000), 545–547 vol.2. ISBN: 0-7803-6359-0.

45. Molnar, C. *Interpretable Machine Learning. A Guide for Making Black Box Models Explainable* https://christophm.github.io/interpretable-ml-book/ (2019).

46. Norris-Hill, J. The modelling of daily Poaceae pollen concentrations. *Grana* **34,** 182–188. https://doi.org/10.1080/00173139509429041 (June 1995).

47. Norris-Hill, J. The influence of ambient temperature on the abundance of Poaceae pollen. *Aerobiologia* **13,** 91–97 (June 1997).

48. Probst, P., Wright, M. & Boulesteix, A.-L. Hyperparameters and Tuning Strategies for Random Forest. eng (2018).

49. Quach, A. Extensions and Improvements to Random Forests for Classification. eng. *All Graduate Theses and Dissertations.* ISSN: 6755 (2017).

50. R Core Team. *R: A Language and Environment for Statistical Computing* R Foundation for Statistical Computing (Vienna, Austria, 2020). https://www.R-project.org/.

51. Ramsay, T. O., Burnett, R. T. & Krewski, D. The Effect of Concurvity in Generalized Additive Models Linking Mortality to Ambient Particulate Matter. eng. *Epidemiology (Cambridge, Mass.)* **14,** 18–23. ISSN: 1044-3983 (2003).

52. Rufibach, K. reporttools: R Functions to Generate LaTeX Tables of Descriptive Statistics. *Journal of Statistical Software, Code Snippets* **31.** http://www.jstatsoft.org/v31/c01/ (2009).

53. Scornet, E. Tuning parameters in random forests. eng. *ESAIM. Proceedings and surveys* **60.** ISSN: 2267-3059 (2017).

54. Smith, M. & Emberlin, J. Constructing a 7-day ahead forecast model for grass pollen at north London, United Kingdom. *Clinical and experimental allergy : journal of the British Society for Allergy and Clinical Immunology* **35,** 1400–6 (Oct. 2005).

55. Speiser, J. L., Miller, M. E., Tooze, J. & Ip, E. A comparison of random forest variable selection methods for classification prediction modeling. *Expert Systems with Applications* **134,** 93–101. ISSN: 0957-4174. http://www.sciencedirect.com/science/article/pii/S0957417419303574 (2019).

56. Stach, A., Smith, M., Prieto Baena, J. & Emberlin, J. Long-term and short-term forecast models for Poaceae (grass) pollen in Poznań, Poland, constructed using regression analysis. *Environmental and Experimental Botany* **62,** 323–332. ISSN: 0098-8472. http://www.sciencedirect.com/science/article/pii/S0098847207001748 (2008).

57. Tashman, L. Out-of sample tests of forecasting accuracy: a tutorial and review. *Int J Forecasting* **16** (Jan. 2000).

58. Teranishi, H., Katoh, T., Kenda, K. & Hayashi, S. Global warming and the earlier start of the Japanese-cedar (Cryptomeria japonica) pollen season in Toyama, Japan. *Aerobiologia* **22,** 90–94 (June 2006).

59. Thompson, P. A. & Miller, R. B. Sampling the Future: A Bayesian Approach to Forecasting From Univariate Time Series Models. *Journal of Business & Economic Statistics* **4,** 427–436. eprint: https://www.tandfonline.com/doi/pdf/10.1080/07350015.1986.10509542. https://www.tandfonline.com/doi/abs/10.1080/07350015.1986.10509542 (1986).

60. Valderrama, M., Ocaña, F., Aguilera, A. & Ocaña-Peinado, F. Forecasting Pollen Concentration by a Two-Step Functional Model. *Biometrics* **66,** 578–85 (Aug. 2009).

61. Ver Hoef, J. & Boveng, P. Quasi-Poisson vs. negative binomial regression: How should we model overdispersed count data? *Ecology* **88,** 2766–72 (Dec. 2007).

62. Visbeck, M. H., Hurrell, J. W., Polvani, L. & Cullen, H. M. The North Atlantic Oscillation: Past, present, and future. *Proceedings of the National Academy of Sciences* **98,** 12876–12877. ISSN: 0027-8424. eprint: https://www.pnas.org/content/98/23/12876.full.pdf. https://www.pnas.org/content/98/23/12876 (2001).

63. Wood, S. *Generalized Additive Models: An Introduction with R* 2nd ed. (Chapman and Hall/CRC, 2017).

64. Wright, M. N. & Ziegler, A. ranger: A Fast Implementation of Random Forests for High Dimensional Data in C++ and R. *Journal of Statistical Software* **77,** 1–17 (2017).

65. Zeileis, A. & Hothorn, T. Diagnostic Checking in Regression Relationships. *R News* **2,** 7–10. https://CRAN.R-project.org/doc/Rnews/ (2002).

66. Zewdie, G. K., Liu, X., Wu, D., Lary, D. J. & Levetin, E. Applying machine learning to forecast daily Ambrosia pollen using environmental and NEXRAD parameters. *Environmental Monitoring and Assessment* **191.** https://doi.org/10.1007/s10661-019-7428-x (June 2019).

67. Zhang, Y., Bielory, L., Cai, T., Mi, Z. & Georgopoulos, P. Predicting onset and duration of airborne allergenic pollen season in the United States. *Atmospheric Environment* **103,** 297–306. https://doi.org/10.1016/j.atmosenv.2014.12.019 (Feb. 2015).

68. Zuur, A. *et al. Negative Binomial GAM and GAMM to Analyse Amphibian Roadkills* 383–397. ISBN: 978-0-387-87457-9 (May 2009).

Appendix A

Model	Input Variables
1.1	Count, VI_minimum, VI_maximum, VI_mean, VI_sd, VI_UQ, VI_U1.5.IQR, veg_index, VI_L1.5.IQR, VI_LQ, min_temp, max_temp, Temperature, Dew.Point, humid, rain, wind_speed, wind_dir, day, month, year, ds, season, pollen_cat, mintemp_anom, maxtemp_anom
1.2	Count, VI_minimum, VI_maximum, VI_mean, VI_sd, VI_UQ, VI_U1.5.IQR, veg_index, VI_L1.5.IQR, VI_LQ, min_temp, max_temp, Temperature, Dew.Point, humid, rain, wind_speed, wind_dir, day, month, year, fyear, ds, season, pollen_cat, mintemp_ano, cat_lag4, cat_lag5, count_lag1, count_lag2, count_lag3, count_lag4, count_lag5, temp_lag1, temp_lag2, temp_lag3, dew_point_lag1, dew_point_lag2, dew_point_lag3, maxtemp_lag1, maxtemp_lag2, maxtemp_lag3, maxtemp_anom_lag1, maxtemp_anom_lag2, maxtemp_anom_lag3, humid_lag1, humid_lag2, humid_lag3
1.3	ds, pollen_cat, count_lag1, count_lag2, count_lag3, count_lag4, count_lag5
1.4	Count, VI_mean, VI_sd, VI_UQ, VI_U1.5.IQR, veg_index, VI_LQ, min_temp, max_temp, Temperature, Dew.Point, humid, wind_speed, wind_dir, day, month, ds, season, pollen_cat, mintemp_anom, maxtemp_anom, count_lag1, count_lag2, count_lag3, count_lag4, count_lag5, temp_lag1, temp_lag2, temp_lag3, dew_point_lag1, dew_point_lag2, dew_point_lag3, maxtemp_lag1, maxtemp_lag2, maxtemp_lag3, maxtemp_anom_lag1, maxtemp_anom_lag2, maxtemp_anom_lag3, humid_lag1, humid_lag2, humid_lag3

Table A.1: Table showing the predictors for each of the random forest models.

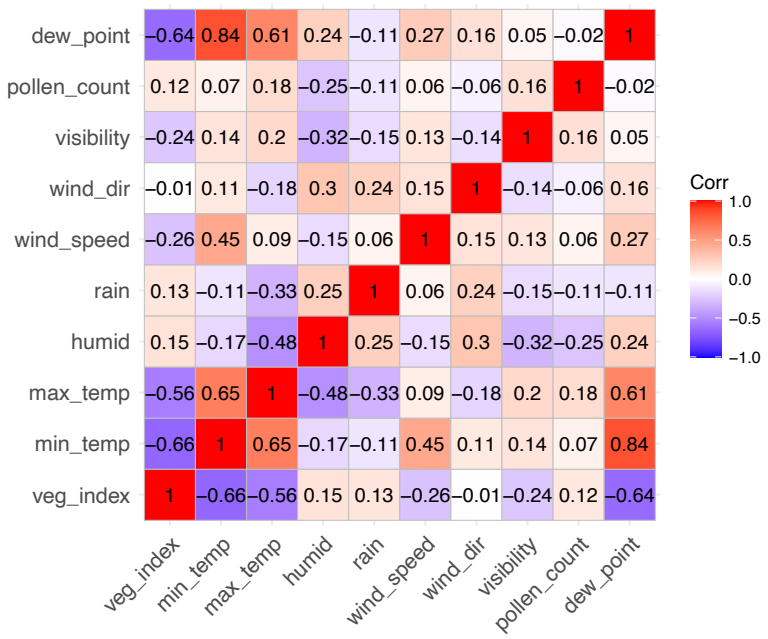


Figure A.1: Correlation Matrix

Season	Model	Year	Classification Accuracy	MSE	MAE	Brier
In Season	1.1	2013	0,412	1,425	0,850	0,733
		2014	0,261	2,085	1,118	0,810
		2018	0,125	3,366	1,509	0,867
	1.2	2013	0,372	1,385	0,872	0,710
		2014	0,277	1,777	1,034	0,760
		2018	0,271	1,738	1,009	0,768
	1.3	2013	0,331	1,696	0,980	0,796
		2014	0,324	1,791	1,007	0,796
		2018	0,439	1,280	0,776	0,694
	1.4	2013	0,338	1,541	0,932	0,709
		2014	0,270	1,858	1,061	0,758
		2018	0,280	1,897	1,056	0,766
Out of Season	1.1	2013	0,539	0,784	0,569	0,520
		2014	0,648	0,505	0,400	0,485
	1.2	2013	0,588	0,676	0,500	0,496
		2014	0,633	0,519	0,414	0,470
	1.3	2013	0,574	0,662	0,505	0,545
		2014	0,506	1,064	0,668	0,618
	1.4	2013	0,608	0,642	0,475	0,499
		2014	0,633	0,519	0,414	0,476
Total	1.1	2013	0,482	1,062	0,692	0,611
		2014	0,485	1,171	0,702	0,622
	1.2	2013	0,497	0,974	0,656	0,586
		2014	0,486	1,039	0,670	0,590
	1.3	2013	0,472	1,097	0,705	0,651
		2014	0,506	1,064	0,668	0,618
	1.4	2013	0,494	1,020	0,668	0,587
		2014	0,483	1,073	0,682	0,593

Table A.2: Complete random forest results, as used by Table 5.1

Season	Model	Year	Classification Accuracy	MSE	MAE	Brier
In Season	2.1	2013	0.347	1.105	0.767	0.606
		2014	0.383	1.692	0.94	0.679
		2018	0.519	0.942	0.615	0.672
		2019	0.283	1.275	0.899	0.67
	2.2	2013	0.41	1.201	0.785	0.71
		2014	0.331	1.857	1.015	0.687
		2018	0.452	1.183	0.74	0.684
	2.3	2013	0.369	1.055	0.767	0.688
		2014	0.288	1.747	1.021	0.688
		2018	0.581	1.095	0.6	0.681
Out of Season	2.1	2013	0.608	0.454	0.412	0.469
		2014	0.643	0.434	0.379	0.469
		2019	0.524	0.774	0.571	0.581
	2.2	2013	0.613	0.464	0.412	0.47
		2014	0.6	0.478	0.423	0.467
	2.3	2013	0.619	0.472	0.411	0.488
		2014	0.586	0.527	0.448	0.471
Total	2.1	2013	0.496	0.736	0.58	0.568
		2014	0.537	0.941	0.606	0.565
		2019	0.481	0.809	0.612	0.556
	2.2	2013	0.41	1.201	0.785	0.71
		2014	0.494	1.04	0.661	0.568
	2.3	2013	0.369	1.055	0.767	0.688
		2014	0.466	1.022	0.68	0.569

Table A.3: Complete GAM results, as used by Table 5.3

The R code used for modelling and testing, as well as the code used to create our Shiny App are available on our GitHub repository at the following link.

Disclaimer: This work is based on the research supported in part by the National Research Foundation of South Africa. Opinions, findings and conclusions or recommendations expressed in this paper are of the author(s) alone, and the NRF accepts no liability whatsoever in this regard.