The School of Mathematics

THE UNIVERSITY
of EDINBURGH

Traffic accident prediction in Great Britain with KNN, GLM and ANN models: a comparative approach

by

Diego José Sánchez Martín

Dissertation Presented for the Degree of
MSc in Operational Research with Data Science

August 2022

Supervised by
Dr Julian Wang
Dr Julian Hall

Abstract

Several modelling approaches have been used for the prediction of traffic flow and some of its negative consequences, such as congestion and accidents. This dissertation focuses on the accidents prediction problem for Great Britain in the period 2016-2019. Firstly, statistical analyses are performed to select and encode the features of interest. The training, validation and test sets created include simulated instances of non-accidents events. Secondly, three different machine-learning techniques are applied: K-nearest neighbours, generalized linear model and artificial neural network. The results obtained are compared from both a performance and an interpretability perspective. The generalized linear model that uses a Poisson distribution with a canonical link function shows the best overall outcomes. On the whole, the findings presented here have interesting implications for the development of evidence-based policies aimed at reducing traffic accidents.

Keywords: traffic, accidents, Generalized Linear Model, interpretability.

Acknowledgments

I would like to express my gratitude to my supervisors Dr. Julian Wang and Dr. Julian Hall. Thank you for your guidance and advice, but also for allowing me the freedom to decide how to develop the project. I would be remiss in not mentioning Kaloyan Bukovski, from Sopra Steria, for his constant attention and enthusiasm.

Thanks should go as well to my personal tutor, Dr. Joerg Kalcsics, for his support during the year, and to Rosie Wilkie, who put me in contact with the Edinburgh International Data Facility and made sure I could access their server.

I could not have undertaken this journey without the generous funding from Fundación Rafael del Pino. Being a member of their fellow community is an immense pleasure and an exigent responsability I take on proudly.

I would also like to acknowledge my friends for keeping my spirits and motivation high those days when the deadline seemed to be approaching too quickly. Special thanks to Carlos Gandiaga, Adrián Garnier, Belén Irureta, Jaime Redondo and Rija Zafar for their editing help and feedback on preliminary drafts.

Last, but not least, I would like to thank my family although no words can fully express my gratitude. This graduation would not have been possible without them and I mean it in every sense of the word.

Own Work Declaration

I confirm that all this work is my own except where indicated, and that I have:

- Clearly referenced/listed all sources as appropriate
- Referenced and put in inverted commas all quoted text (from books, web, etc.)
- Given the sources of all pictures, data etc. that are not my own
- Not made any use of the report(s) or essay(s) of any other student(s) either past or present
- Not sought or used the help of any external professional academic agencies for the work
- Acknowledged in appropriate places any help that I have received from others (e.g. fellow students, technicians, statisticians, external sources)

I understand that any false claim for this work will be penalised in accordance with the University regulations

Contents

List of abbreviations

AI	Artificial Intelligence
ANN	Artificial Neural Network
ARIMA	Autoregressive Integrated Moving Average
EIDF	Edinburgh International Data Facility
GAM	Generalized Additive Model
GLM	Generalized Linear Model
INAR	Integer-valued Autoregressive
KNN	K-Nearest Neighbours
LSTM	Long Short-Term Memory
MAE	Mean Absolute Error
MLE	Maximum Likelihood Estimation
mph	Miles per hour
MSE	Mean Squared Error
NHS	National Health Service
NHTSA	National Highway Traffic Safety Administration
NTIS	National Traffic Information Service
OLS	Ordinary Least Squares
OSGB	Ordnance Survey of Great Britain
OSGR	Ordnance Survey Grid Reference
ReLU	Rectified Linear Activation Unit
SARIMA	Seasonal Autoregressive Integrated Moving Average
TLU	Threshold Logic Unit
UK	United Kingdom
USA	United States of America

List of Tables

List of Figures

1 Introduction

This is my final dissertation for the MSc Operational Research with Data Science at The University of Edinburgh. I consider this thesis embodies some of the most important learning outcomes expected from a student of the aforementioned degree: rigorous mathematical modelling, problem-solving skills and data analysis. The time spent working on it has been both challenging and rewarding. I really hope the time the reader takes to look through this paper is equally worth it.

1.1 Problem definition

The main goal of this dissertation is to analyse traffic accidents in Great Britain from a predictive approach. In order to do so, I will apply Data Science since this field encompasses a wide range of tools that combine statistics and programming with the purpose of obtaining knowledge from data. In our case of study, data describing recent traffic accidents having taken place in Great Britain will be used to train and test several machine learning models. The severity of the accidents will be taken as the variable to be predicted, since it is a good proxy for the importance of the event. Some of the key challenges of this problem are the big size of the datasets, the imbalance of severity outcomes and the presence of randomness in the data. They will be explained in greater detail along the thesis.

The expected outcomes of the dissertation are outlined below:

- Predicting the severity of an event (either accident or non-accident) using surrounding conditions as variables.
- Gaining insight into which variables play a greater role increasing the severity of accidents.
- Comparing the performance of the machine learning models selected for the task.
- Discussing the trade-offs involved in maximizing prediction performance.

1.2 Partnership with Sopra Steria

The University of Edinburgh offers students the possibility of doing their final project in partnership with an external organization. The main perk of this arrangement for the student comes from gaining experience in a working environment that will be closer to the professional world outside academia. I wanted my dissertation to have this practical component and that is how Sopra Steria came along.

Sopra Steria is a French company recognised for their consulting, digital services and software development, with strong focus on digital transformation. It provides end-to-end solutions to make organisations more competitive by combining knowledge of a range of business sectors and innovative technologies [2]. Formally, Sopra Steria started in 2014 with the merge of Sopra (consulting firm founded in 1968) and Groupe Steria SCA (IT services company founded in 1969). As of 2022, Sopra Steria possess offices in nearly 30 countries and employs over 47,000 people [3]. Total global revenue for the first half of 2022 reached more than €2,500 million [49].

For this project, I will be working with Sopra Steria UK, second biggest country branch of the group in Europe after France. In the United Kingdom, Sopra Steria has many different clients that can be divided up into two groups: private and public sectors. In the private side, they work mainly for the retail, finance and airlines industries. The public entities advised by Sopra Steria include the NHS, the UK Visas and Immigration Department and several police forces. Given the problem addressed, the expertise of the company in the public domain is an invaluable asset for the project. I would like to thank once again both The University of Edinburgh and Sopra Steria for making the partnership possible. I strongly believe establishing connections between industry and academia is most beneficial not only for them, but also for society as a whole.

1.3 Informatics tools

A big amount of computing was required for this dissertation. Due to the size of the datasets used, my personal laptop alone was not powerful enough to handle them. For that reason, I was given access to a virtual machine hosted by the Edinburgh International Data Facility, which allowed me to run the models seamlessly.

The programming language used was Python. Its main advantages are that it is an open source language widely applied in industry and the vast supply of libraries that can be found for very different tasks. In my case, the most important libraries were Pandas for dataframes creation and manipulation and Matplotlib and Seaborn for visualizations. With respect to the models, KNN was imported from Scikit-learn, GLM came from Scipy and the ANN was designed with Keras. Some other libraries that were utilized for auxiliary tasks were Numpy, Datetime, Random, Astral, Timezonefinder and Faker.

Despite the fact that this project is individual, communication with my supervisors has been key. Therefore, Slack and Microsoft Teams were used as platforms for constant interaction. Moreover, the dissertation was written on LaTeX using Overleaf as online editor so that the ongoing text was accessible to everyone at all times.

1.4 Outline of the dissertation

The sections that follow are organized to show the workflow required to solve the problem proposed in 1.1. Firstly, section 2 sets the foundations for the dissertation by explaining my personal motivations to take on this project (2.1) and reviewing the most relevant literature on the topic (2.2). The mathematical basis for the models that will be applied is shown in section 3, making special emphasis in the pros and cons of each of them. Later on, section 4 makes a thorough analysis of the data available to describe overall trends and prepare the datasets that will be used in section 5, where each of the models are implemented and results are shown and discussed. Finally, section 6 presents the main findings and comments on lines of possible future development. There are also several appendices (A,B,C,D) containing extra information that is not essential to understand the dissertation but can be of interest.

2 Background

In the following, I will present some background information that will be of great importance for a better understanding of the dissertation as a whole. Firstly, it is fair and necessary to wonder about the value of the work I have carried out during the last few weeks. Although this dissertation is a formal requirement to obtain my MSc. diploma, I do not consider that my effort has been dedicated exclusively to achieve an academic qualification. Therefore, I would like to share some of my personal motivations and a brief overview of the reasons why I think looking at traffic accidents data is worth it. Secondly, a literature review is pertinent in order to incorporate to my dissertation previous work on the area. After all, even Sir Isaac Newton admitted having seen further in his research "by standing on the shoulders of giants". Moreover, this section will also justify why certain models —and not others— have been chosen to perform my predictions.

2.1 Motivation

I have been very interested in cars since a very early age. I remember playing with a remote control car when I was around six years old. I particularly liked trying to make it jump from the top step of my house's stairs. Although this might be seen as a child creating havoc for fun, what made me do it was not the desire to break my toy, but a genuine curiosity about how the speed of the car could change the outcome of my little driving adventure. If I made the car go fast enough, it did not overturn. In all fairness, you could call it my very first experiment.

Time went on and I learnt to drive a proper car, I promise that without any dangerous leaps over stairs. Aside from enjoying the sense of freedom and independence that driving gave me, I realized it could also become a source of inspiration for some of my academic projects. That's one of the best attributes of the field of Mathematics: it can be applied to almost anything imaginable. In my case, the results of this application were two undergraduate thesis: a dynamical model of a Tesla Model 3 using Simulink over MATLAB and a data science project on predictive maintenance for Scania trucks. It is rather appropriate that this dissertation keeps a similar overall topic but with a different angle: traffic safety.

Unfortunately, dying as a result of a traffic accident is far more common than most people realise. According to the road safety charity Brake [8], about 1.35 million people in the world lose their lives every year because of road accidents. Many more suffer significant injuries. In fact, this kind of accidents is the first cause of death for people aged from 5 to 29 years old. Given these figures, it is understandable that the United Nations (UN) declared the period 2021–2030 as the Second Decade of Action for Road Safety [20], with a goal of reducing road traffic deaths and injuries by at least 50 percent from 2021 to 2030. Indeed, the period 2011-2020 was the first decade of this kind. Although progress was made, the importance of the problem encouraged the UN to extend the special awareness phase ten more years. After all, road transport remains a vital development factor [50], so efforts to make sure it can happen as smoothly as possible are of great economic interest.

In this dissertation, the focus will be on traffic accidents taking place in Great Britain. For this reason, let's take a look at Figure 1 to find out about some statistics for casualties occurred over the last two decades[1]. The graph shows the total number of people being either killed or seriously injured as a result of a road accident every year from 2000 to 2020. This includes drivers involved in those accidents, but also vehicle passengers, pedestrians and cyclists. It is important to keep this in mind because the same definition will be used all along the dissertation when referring to traffic accidents casualties.

Even if 2020 should be considered an outlier due to mobility being greatly disrupted as a result of the Covid-19 pandemic, the graph shows a clear downward tendency. We started this millennium with

[1]Source: graph prepared by the author using data available on demand at https://roadtraffic.dft.gov.uk/custom-downloads/road-accidents

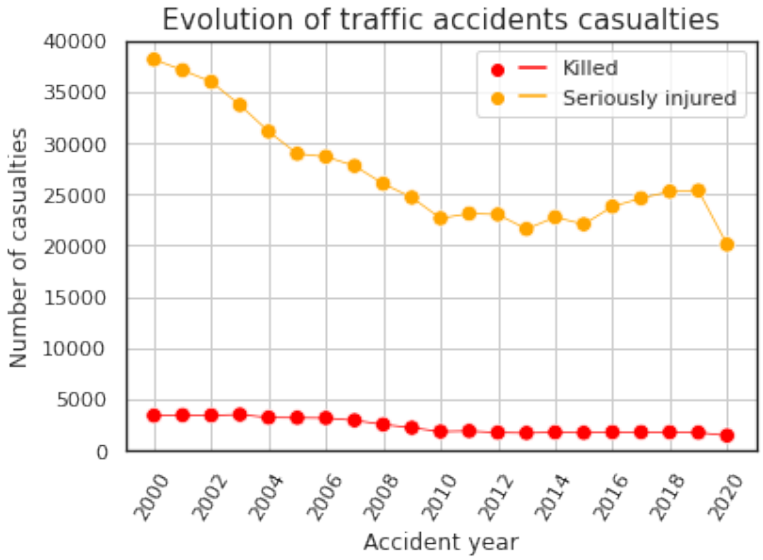


Figure 1: Evolution of casualties (deaths and serious injuries) in Great Britain in the period 2000-2020.

around 40,000 people seriously injured and 4,000 deaths a year as a consequence of traffic accidents. By 2019, those numbers had fallen to 25,000 and 1,700 respectively. This improvement is especially remarkable because the number of registered vehicles in the UK has increased from 27.2 million to 32.7 in the same period according to Statista [48], which is a 20% growth. Having said that, the fact that roads are safer than they used to be does not imply that they are *safe*. Absolute casualties numbers are still high and research on how to reduce them will always be helpful. I expect that this dissertation can be my humble academic contribution towards the ambitious goal set by the UN of halving the number of people who die on the road.

It is widely believed that our best opportunity to produce a significant and relatively fast improvement in this issue comes from autonomous cars. In fact, this idea is supported by the United States Department of Transport in its recent report *Ensuring American Leadership in Automated Vehicle Technologies* [14]. Among other benefits that this technology could bring to the USA from a strategic standpoint, the report mentions that road casualties would be greatly reduced. Vehicles that can drive by themselves follow traffic rules better than humans do and therefore should be safer. Actually, Tesla, the car manufacturer whose vehicles possess the most advanced self-driving capabilities at time of writing this dissertation, publishes data on the topic on a regular basis [28]. According to them, and looking at the most recent available figures (corresponding to the fourth quarter of 2021), we have that:

- The average distance traveled for an automobile in the USA to suffer a crash is 484,000 miles. Tesla takes this number directly from statistics offered by the National Highway Traffic Safety Administration (NHTSA).
- The average distance traveled for a Tesla automobile to suffer a crash is 1.59 million miles when no self-driving feature is activated.
- The average distance traveled for a Tesla automobile to suffer a crash is 4.31 million miles when Autopilot (the company's self-driving technology) is activated.

The immediate conclusion that could be drawn from data above is that Autopilot makes cars between 3 and 9 times safer, since this is the ratio of distances that are driven on average in each case before an accident occurs. One should obviously consider these numbers with a grain of salt. There is an undeniable benefit for Tesla as a private company to advertise their cars and exclusive features being much safer than the competence. It is not clear if it is fair and rigorous to compare accident propensity of Tesla's vehicles, usually driven by at least upper middle class people, with

general NHTSA statistics. Moreover, Autopilot is mostly used for long trips on highways, where the probability of an accident tends to be lower than on urban settings (more precise data on this in section 4.2.2). Anyway, I am referring to these statistics as an indicator of the enormous potential of autonomous cars to improve road safety.

My aim with this dissertation is gaining insight into what situations induce a greater danger of accident. That is why I will be using several machine learning models and I will compare not only their performance but also the information they provide us with about the variables used as input. Even if the future of mobility is about self-driving vehicles, knowing precisely which elements contribute to accidents will help create the software of those new generation cars so that risk factors are taken into account.

2.2 Literature review

For all the reasons previously explained, problems associated with traffic have been under study for a long time. Despite this, the focus has not always been on accidents, but on the flow of traffic. This might respond to a necessity from public authorities to organize the road network efficiently. In consequence, this literature review section will present examples of works that have targeted broad predictions regarding road traffic, no matter if they are specifically aimed at accidents.

Given that traffic flow prediction approaches have usually been divided into parametric or non-parametric [47], the same categories will be kept here. Studies tend to show that parametric models deliver powerful statistical results that are easier to interpret. Nevertheless, non-parametric approaches handle complexity in a more versatile way and usually offer better prediction outcomes [43].

2.2.1 Parametric approach

The parametric algorithms are defined by a specific model whose structure is clearly defined in advance and only the values of a set of parameters need to be estimated [23]. These models assume a previous knowledge of the way the process of interest works. When it comes to traffic, the most common choice is relying on time series.

Since a relatively long time ago, there have been several instances of authors using this tool to model accidents in Great Britain. To cite a few, Scott used monthly data from 1970 to 1978 to predict data for the following three-year period [45]. He made distinctions between accidents where two vehicles were involved and accidents with only one vehicle. For the former he used regression, with reasonably consistent results. For the latter trends seemed more complicated so he decided to switch to a Box-Jenkins time series method, but his results were not particularly promising. Broughton [9] used a much longer time span (1949-1989) to try to predict the number of casualties that would occur on the year 2000. The methodology was based on the observation that the decline in casualties and accidents per hundred million vehicle driven-kilometres over the four decades under study was remarkably regular, so he tried to fit a logarithmic curve. His main conclusion was the existence of significant uncertainty about forecasting casualties 11 years ahead in time. More recent works pointing in the same direction include Quddus' comparison of ARIMA and INAR models to study accidents in Great Britain in the 1950-2005 period [40]. He found that INAR(1) Poisson performed better than ARIMA to fit the observed data to the model.

Models that employ time series are not outdated despite the recent popularity of non-parametric approaches. For instance, Deretic et al. constructed a Box–Jenkins seasonal autoregressive integrated moving average (SARIMA) model to investigate the pattern in the time series describing the monthly number of traffic accidents in the city of Belgrade, Serbia [15]. They were able to show a very important seasonal component, since accidents increased significantly during the last quarter of the year. Also, there were few large outliers, as only 5 out of the 48 values under study (every month for the years

2016-2019) laid outside their confidence intervals. My predictions in this dissertation will refer to the same time period, but with more granular data from Great Britain.

The main drawback of the studies already cited is that the models they used are rather simple and only consider statistical trends across time. Their outcomes are predictions on total number of accidents, casualties or deaths for a given location using past data. This is not the authors' fault or a result of lack of computing power at the time of publishing, but a feature of the models themselves. However, not all parametric models have to be time series based. Fitiantri et al. [17] used a GLM to model the material loss as a result of traffic accidents in Indonesia with four predictors: carelessness, indiscipline, alcohol use and speed excess. They were then able to assign a weight to each of these four factors. Although their context and objectives differ from mine, I will also make use of a GLM for the problem to be tackled of this dissertation. Its most important characteristic is that I will be able to extract information from the coefficients fitted to each variable. The details on the model will be fully explained in section 3.2.

2.2.2 Non-parametric approach

Conversely to the case of the parametric approach, non-parametric algorithms do not assume a particular structure for their models. Therefore, both the exact model structure and its parameters need to be estimated solely from the data at hand. This makes them more flexible but also more complex from a computational standpoint.

One of the most common algorithms that belong to this category is KNN. Its definition, which can be found in section 3.1, is not very complicated and therefore it will be used as a baseline solution for the problem. There are a few examples of this model being used for traffic prediction that are worth mentioning. Habtemichael et al. [25] worked on an enhanced KNN model for short-term traffic forecasting based on the weighted Euclidean distance as similarity measure, which is the same distance used for my algorithm. Furthermore, they added some extra features to minimize the influence of dominant neighbors. Also, Zheng and Su [55] dealt with the same kind of forecasting problem applying a customized KNN algorithm that reduced the effect of extreme values filtering them with the aid of a principal component algorithm.

Artificial neural networks (ANN) are probably the best tool when the accuracy of the predictions is the most important result. They are able to incorporate the non-linear effects that may arise in traffic dynamics. Furthermore, ANN models have been much improved due to the recent popularity of deep learning. A simple search on Google Trends produces the graph below.

Figure 2: Interest of the term "Deep Learning" in Google searches over time for the period from January 2004 to August 2022.

Figure 2 shows the explosion of interest in this technology midway through the 2010 decade. It has produced a revolution in machine learning and traffic prediction is not an exception. We can name

several studies where ANN models have been used to predict traffic flow, such as the ones by Kumar et al. [32] or Zhu et al. [56]. Proper deep learning approaches started around 2015, obtaining better results. The most usually cited example of this is the work of LV et al [34], where autoencoders are used in the feature space.

Models of a similar kind have also been applied to traffic accidents. Qian et al. [39] used an artificial neural network to predict the number of deaths in car accidents in China. Their results were superior to the ones obtained with time series. An analogous study was later carried out for Switzerland roads [19]. Additionally, it is worth mentioning that Sameen et al. [44] applied a recurrent neural network (a special kind of ANN) to predict the severity of accidents. This dissertation will also used severity as the target variable, since I find that it is a logical way of quantifying the impact of an accident. The exact treatment of this variable will be explained in section 4.3.

Finally, I would like to highlight a master's thesis I read while planning mine. Eva Elisabeth Oudemans explored similar topics to the ones I have in her dissertation "Predicting traffic accident injury severity in Great Britain using machine learning techniques" in 2016 [38]. Despite her interest in severity, she considered a binary classification problem where the models only produced as outcome if a given accident was fatal or not. I have attempted to expand her work by considering a wider range of possible values using regression instead of classification algorithms.

3 Predictive Models

This section is devoted to describe the models that will be applied when performing the accidents predictions. The focus will not only be on the mathematical formulation of each of these models, but also on highlighting their main advantages and disadvantages for the task ahead.

3.1 KNN regressor

The main concept underlying the idea of a K-Nearest Neighbours (KNN) algorithm is that there is a certain *distance* separating any given pair of instances. Before clarifying what this means for the purpose of the problem at hand, we shall state a formal definition of metric [41], the mathematical term for the intuitive idea of distance.

Definition 3.1 *A metric on a set X is a specification of a distance $d(x, y)$ between any two points $x, y \in X$, in other words a map $d : X \times X \longrightarrow \mathbb{R}$, required to satisfy the following axioms for all $x, y, z \in X$:*

1. *Positive definiteness: $d(x, y) \geq 0$ and we have that $d(x, y) = 0$ if and only if $x = y$.*
2. *Symmetry: $d(x, y) = d(y, x)$.*
3. *Triangle inequality: $d(x, y) \leq d(x,z) + d(z, y)$.*

This can be easily used to work out the distance between two instances on a dataset. The feature space is represented by X and you can choose any distance function d satisfying the properties above. A very usual one is the Euclidean distance for vectors in $\mathbb{R}^n$, where $d(x, y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$. It is easy to see that this distance function follows the definition above.

KNN algorithms first use was as classifiers. Once we have a dataset containing instances and their true labels, we can predict the label of a new instance by considering the distance of these new instance to all the labeled ones than we have. Then, we will simply select the k closest ones and assign the predicted label as the most common label among this group of "close instances" [46]. Since there might be cases where two or more labels are tied, extra rules need to be put in place to make sure the classifier returns a valid result. Examples of this are giving priority to the closest label among the ones tied or extend the classification to $k + 1$, $k + 2$... until the tie is broken. If k is set to 1, the resulting space for a binary classification problem looks as follows[2]:

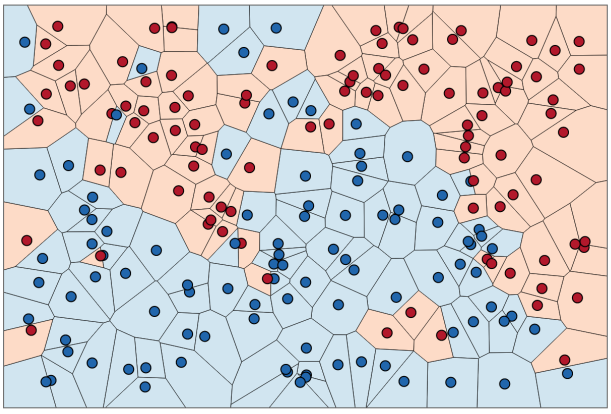


Figure 3: Feature space for a binary classification problem solved with 1NN.

There is a very clear intuition behind this algorithm: if instances are near in the feature space, their labels are likely to be the same. Moreover, the same principles can be applied to obtain a regressor instead of a classifier. In this case, the prediction for an instance will be either the simple average of the outcomes for its k-nearest neighbours or a weighted average where the weight of each value will be the inverse of the distance to the new instance whose regression we are trying to work out.

[2]Image taken from https://kevinzakka.github.io/2016/07/13/k-nearest-neighbor/

3.2 GLM

Before digging into the way Generalized Linear Models (GLMs) work, it is necessary to step back and look at a simple yet useful concept that lies at the core of a GLM: the linear regression.

3.2.1 Linear regression

A simple linear regression model [37] is one such that a single variable x has a relationship with a response y that follows a straight line:

$$y = \beta_0 + \beta_1 x + \varepsilon, \tag{3.1}$$

where β_0 is the intercept, β_1 is the slope of the straight line and ε represents a random error. This is obviously an extremely simple approach, as the vast majority of outcomes of interest y —let's say, for instance, traffic accidents—, depend on more than one factor. For this reason, the natural extension of the aforementioned idea is the multiple linear regression [37], that takes the following form:

$$y = \beta_0 + \sum_{i=1}^{n} \beta_i x_i + \varepsilon, \tag{3.2}$$

where there are n regressor variables denoted by x_i $(i = 1, ..., n)$, each of which is multiplied by a (probably different) coefficient β_i that weighs the sum to arrive to the outcome y. If it is assumed that the error ε follows a Gaussian distribution with mean zero (and this is a *big* assumption, as it will be explained later in 3.2.2), then the expected value of the outcome y for a set of given variables $\bar{x}$ is:

$$E(y|\bar{x}) = \beta_0 + \sum_{i=1}^{n} \beta_i x_i. \tag{3.3}$$

Although this expression is quite similar to 3.2, it allows for a very clear interpretation of the coefficients β_i. For any given i_0, the parameter β_{i_0} measures the expected change in the outcome y per unit of change in x_{i_0} assuming that all the remaining regressor variables x_i $(i \neq i_0)$ are held constant. Therefore, if a certain process of normalisation is done over the regressor variables, the value of β_i is a measurement of the importance of feature x_i.

The immediate question that arises is how to work out the values of the coefficients β_i given a set of N pairs of variables and outcomes $(\bar{x}_j, y_j)$. Since our goal is finding those values so that our predictions are as close to the actual outcomes as possible, we will try to minimize the sum of the squared errors ε_j^2. This is important to make sure that errors are counted in the same way for both underestimating or overestimating y_j. The method [18] is usually referred to as OLS, that stands for Ordinary Least Squares. The function to be minimized is

$$f_{OLS}(\beta_0, \beta_1, ..., \beta_n) = \sum_{j=1}^{N} \varepsilon_j^2 = \sum_{j=1}^{N} \Big(y_j - E(y|\bar{x})\Big)^2 = \sum_{j=1}^{N} \Big(y_j - \beta_0 - \sum_{i=1}^{n} \beta_i x_{ji}\Big)^2, \tag{3.4}$$

where x_{ji} is the value of variable i for instance j. There are $n+1$ derivatives to work out:

$$\frac{\partial f_{OLS}}{\partial \beta_0} = -2 \sum_{j=1}^{N} \Big(y_j - \beta_0 - \sum_{i=1}^{n} \beta_i x_{ji}\Big) \tag{3.5}$$

$$\frac{\partial f_{OLS}}{\partial \beta_i} = -2 \sum_{j=1}^{N} \Big(y_j - \beta_0 - \sum_{i=1}^{n} \beta_i x_{ji}\Big) x_{ji} \quad \forall i \neq 0.$$

By making the derivatives equals to 0 and rearranging the terms we arrive to the system of equations:

$$\begin{aligned} n\beta_0 + \sum_{i=1}^{n}\left(\beta_i \sum_{j=1}^{N} x_{ji}\right) &= \sum_{j=1}^{N} y_j \\ \beta_0 \sum_{j=1}^{N} x_{j1} + \sum_{i=1}^{n}\left(\beta_i \sum_{j=1}^{N} x_{ji}x_{j1}\right) &= \sum_{j=1}^{N} y_j x_{j1} \\ \beta_0 \sum_{j=1}^{N} x_{jn} + \sum_{i=1}^{n}\left(\beta_i \sum_{j=1}^{N} x_{ji}x_{jn}\right) &= \sum_{j=1}^{N} y_j x_{jn}. \end{aligned} \quad \vdots \tag{3.6}$$

If we assume that the regressors are linearly independent, then there is a unique solution for the system of equations above [37] because the coefficient matrix that multiplies the unknowns has full rank. We can therefore obtain a set of solutions $(\hat{\beta}_0, \hat{\beta}_1, ..., \hat{\beta}_n)$.

3.2.2 Shortcomings of linear regression

Despite the popularity and undeniable usefulness of linear regression, it is not the best method when it comes to complex predictions. Given its simplicity, there are certain effects that cannot be captured by this type of model. The most important of them are outlined below:

1. Linear regression does not account for any interaction between two or more variables. Expression 3.2 treats each variable separately. Fortunately, there is an easy way to introduce the interaction of two variables in the model by artificially creating a new feature that is defined as the product of two previously defined variables.

2. There is an assumption on normality that might be violated. Normal distributions are the default for many statistical purposes because they are ubiquitous in nature, but one should always wonder if they apply to the problem at hand. If they don't, then other distributions should be considered and the model will become more complex: we will be using a GLM. Traffic accidents are events that do not happen very usually if you consider the huge number of cars that use the road network everyday. Indeed, the most serious among them happen the least often. These factors and others that will be explained further in section 4 justify the use of the Poisson distribution for this dissertation.

3. Non-linear behaviours are not captured by linear regression, which can be a problem. This issue is usually solved by considering Generalized Additive Models (GAMs), where a linear predictor involving a sum of smooth functions depending on the variables of interest is used [53]. The smooth functions are worked out with basis expansions. GAMs are quite complex because there is a great deal of freedom to choose the basis expansions and their use is beyond the scope of this dissertation, but they are worth mentioning as a natural extension of GLMs. Moreover, the one-hot encoding approach that will be used to deal with categorical data (more on this in section 4.3.1) limits the results than can be obtained with a GAM, which is better suited for numerical variables.

3.2.3 Non-Gaussian outcomes

As it was explained in the previous section, the motivation for the introduction of GLMs comes from the existence of certain phenomena that do not follow a normal distribution. The way to express this in a mathematical way [5] is introducing a slight modification to equation 3.3:

$$g(E(y|\bar{x})) = \beta_0 + \sum_{i=1}^{n} \beta_i x_i, \tag{3.7}$$

where g is the *link function*, a monotonic and differentiable mapping that connects the usual linear regression to the expected value of the outcome y. It is essential to point out that the expected value denoted by $E(y|\bar{x})$ is no longer the mean of a normal distribution, but instead it represents the expectation of a distribution belonging to the exponential family [7]. Since this family of functions will be key for GLMs, let's define it properly.

Definition 3.2 *The exponential family of distributions over y, given parameters θ, is defined to be the set of distributions of the form*

$$f(y|\theta) = h(y)a(\theta)e^{\theta u(y)}, \tag{3.8}$$

where θ is called the "natural parameter", functions h and u are functions of y and $a(\theta)$ is a coefficient to make sure that the integral is normalized to 1. Note that θ (and therefore $u(y)$) might be a vector if there are several parameters of interest.

Many of the most usual distributions fit the mold of equation 3.8. Some of the most important ones are shown below. For each of them, we will find out the natural parameter going from their usual density function to the exponential family form.

- Gaussian distribution:

$$N(y|\mu,\sigma) = \frac{1}{\sqrt{2\pi\sigma^2}}e^{-\frac{1}{2}\left(\frac{y-\mu}{\sigma}\right)^2} = \frac{1}{\sqrt{2\pi}}\frac{1}{\sigma}e^{-\frac{y^2}{2\sigma^2}+\frac{\mu y}{\sigma^2}-\frac{\mu^2}{2\sigma^2}} = \tag{3.9}$$
$$= \frac{1}{\sqrt{2\pi}}\left(\frac{1}{\sigma}e^{-\frac{-\mu^2}{2\sigma^2}}\right)e^{-\frac{y^2}{2\sigma^2}+\frac{\mu y}{\sigma^2}} = \frac{1}{\sqrt{2\pi}}\left(\sqrt{-2\theta_2}\times e^{\frac{\theta_1^2}{4\theta_2^2}}\right)e^{\theta(y,y^2)} = f(y|\theta),$$

  where the equality is achieved by taking $\theta = (\frac{\mu}{\sigma^2}, \frac{-1}{2\sigma^2})$ in vector form, $h(y) = \frac{1}{\sqrt{2\pi}}$ and the normalizer $a(\theta) = \sqrt{-2\theta_2}\times e^{\frac{\theta_1^2}{4\theta_2^2}}$.

  This implies that linear regression can be retrieved from the more general GLM approach, given that the normal distribution can be considered as part of the exponential family.

- Bernoulli distribution:

$$Bern(y,\mu) = \mu^y(1-\mu)^{1-y} = e^{y\log(\mu)+(1-y)\log(1-\mu)} = (1-\mu)e^{\log(\frac{\mu}{1-\mu})y} = \tag{3.10}$$
$$= (1-\frac{1}{1+e^{-\theta}})e^{\theta y} = \frac{e^{-\theta}}{1+e^{-\theta}}e^{\theta y} = \frac{1}{1+e^{\theta}}e^{\theta y} = f(y|\theta),$$

  where the natural parameter is $\theta = \log(\frac{\mu}{1-\mu})$, which allows us to write μ using θ via the sigmoid function. In this case, $h(y) = 1$ and $a(\theta) = \frac{1}{1+e^{\theta}}$[3].

  The Bernoulli distribution is very useful for binary classifications or random events with two possible outcomes, such as the toss of a coin.

- Poisson distribution:

$$Pois(y,\lambda) = \frac{\lambda^y e^{-\lambda}}{y!} = \frac{1}{y!}e^{-\lambda}e^{y\log(\lambda)} = \frac{1}{y!}e^{-e^{\theta}}e^{\theta y} = f(y|\theta), \tag{3.11}$$

  which has $\theta = \log(\lambda)$ and the functions $h(y) = \frac{1}{y!}$ and $a(\theta) = e^{-e^{\theta}}$.

  This distribution will be extremely important since its shape fits very well the reported accidents count when we filter by severity. Further explanations on this in section 5.1.

[3] Interestingly, this expression is actually the sigmoid function of $-\theta$, so we have $\mu = \sigma(\theta)$ and $a(\theta) = \sigma(-\theta)$.

There are a lot more possible distributions (binomial, chi-squared, log-normal, gamma, beta...) that can be expressed in this way, but they will not be required for this dissertation. On the other hand, it is relevant to point out that the choice of any of these distributions has an effect on the link function g. After all, as we can see in 3.7, the role of the link function is transforming the expected value into something that the regression can fit. As it is explained in [1], we can extract the canonical link just from the term that multiplies y in the exponential family density function. This link function is called "canonical" because it is defined via $g(\mu) = \theta$, by writing the natural parameter in terms of the mean of the distribution. The canonical links of the distributions above would be $g(\mu) = \mu$ (identity) for the normal, $g(\mu) = \log(\frac{\mu}{1-\mu})$ (logit function) for the Bernoulli and $g(\lambda) = \log(\lambda)$ for the Poisson. Other link functions are possible as long as they respect the support that comes from each distribution. The table below shows some of the most commonly used link functions.

Distribution	Link name	Link expression
Normal	Identity	$g(\mu) = \mu$
Bernoulli	Logit	$g(\mu) = \log(\frac{\mu}{1-\mu})$
Binomial	Cloglog	$g(\mu) = \log(-\log(1-\mu))$
Poisson	Logarithm	$g(\mu) = \log(\mu)$
Gamma	Inverse	$g(\mu) = \mu^{-1}$
Inverse Gaussian	Inverse squared	$g(\mu) = \mu^{-2}$

Table 1: Link functions for a range of distributions.

3.2.4 Fitting a GLM

Once a distribution and link function have been selected, we have to find a way to fit the model to the experimental data. This consists in finding the values of β_i that connect the features to the expected outcome. The best method to do so is the Maximum Likelihood Extimation (MLE) [5], that uses the log likelihood to make the most of the structure of the exponential family and convert the product of likelihoods into a sum. If we have a collection of N instances $(\bar{x}_j, y_j)$ as in section 3.2.1, the log likelihood formula is as follows:

$$\log(L(\beta_0, \beta_1, ..., \beta_n)) = \log(\prod_{j=1}^{N} L_j(\beta_0, \beta_1, ..., \beta_n)) = \sum_{j=1}^{N} \log(L_j(\beta_0, \beta_1, ..., \beta_n)) = \tag{3.12}$$
$$= \sum_{j=1}^{N} \log(f(y_j|\theta_j)) = \sum_{j=1}^{N} \Big[\log(h(y_j)) + \log(a(\theta_j)) + \theta_j u(y_j) \Big]$$

Then, this expression will be derived with respect to each of the coefficients β_i using the chain rule. For notation purposes, we will use $\mathcal{L}_j := \log(L_j(\beta_0, \beta_1, ..., \beta_n))$ and $\eta_j := g(\mu_j) = \beta_0 + \sum_{i=1}^{n} \beta_i x_{ji}$. We can now write the derivatives as

$$\frac{\partial \mathcal{L}_j}{\partial \beta_i} = \frac{\partial \mathcal{L}_j}{\partial \theta_j} \frac{\partial \theta_j}{\partial \mu_j} \frac{\partial \mu_j}{\partial \eta_j} \frac{\partial \eta_j}{\partial \beta_i} = \Big(u(y_j) + \frac{a'(\theta_j)}{a(\theta_j)} \Big) \frac{\partial \theta_j}{\partial \mu_j} (g'(\mu_j))^{-1} x_{ji}, \tag{3.13}$$

where it is assumed for the sake of simplicity that $x_{j0} = 0 \quad \forall j \in \{1, ..., N\}$. If we incorporate this to the expression for the whole log likelihood we come up with a set of equations over β_j that need to be solved:

$$\frac{\partial \log(L(\beta_0, \beta_1, ..., \beta_n))}{\partial \beta_i} = \sum_{j=1}^{N} \Big[\Big(u(y_j) + \frac{a'(\theta_j)}{a(\theta_j)} \Big) \frac{\partial \theta_j}{\partial \mu_j} (g'(\mu_j))^{-1} x_{ji} \Big] = 0 \quad \forall i \in \{1, 2, ..., n\} \tag{3.14}$$

Obviously, the equations above depend on the particular exponential family and on the link function that are chosen in each case. The solutions are complicated because of the large number of

parameters and thus an analytical expression of their solutions cannot be provided. In general, numerical methods such as gradient descent are used to arrive to the final values for $\hat{\beta}_i$. An important note to be made is that for these methods to work it is often necessary to use the negative log-likelihood so that a minimum (and not a maximum) is found.

Let's see how the process works out for a particular example: the Poisson distribution. This distribution is interesting both because the solution is quite simple and also due to the fact that it will be applied to the modelling of section 5.1. We have to take into account that the link function is the logarithm, $u(y) = y$ and $a(\theta)$ takes the form of $e^{-e^{\theta}}$, so its derivative is $a'(\theta) = e^{-e^{\theta}} \frac{d(-e^{\theta})}{d\theta} = -a(\theta)e^{\theta}$. Therefore, we can rewrite 3.14 for this case:

$$\frac{\partial \log(L(\beta_0, \beta_1, ..., \beta_n))}{\partial \beta_i} = \sum_{j=1}^{N} \Big[\Big(y_j + \frac{-a(\theta_j)e^{\theta_j}}{a(\theta_j)} \Big) \frac{\partial \log(\lambda_j)}{\partial \lambda_j} \Big(\frac{\partial \log(\lambda_j)}{\partial \lambda_j} \Big)^{-1} x_{ji} \Big] = \tag{3.15}$$

$$= \sum_{j=1}^{N} \Big[(y_j - e^{\theta_j}) x_{ji} \Big] = \sum_{j=1}^{N} \Big[(y_j - \lambda_j) x_{ji} \Big] = 0 \quad \forall i \in \{1, 2, ..., n\}$$

Although β_i is not explicitly in the above equation, we have that $\log(\lambda_j) = g(\lambda_j) = \beta_0 + \sum_{i=1}^{n} \beta_i x_{ji}$, so when we find the λ_j's that satisfy 3.15 we will have that $\lambda_j = e^{\beta_0 + \sum_{i=1}^{n} \beta_i x_{ji}}$, which is equivalent to $\beta_0 + \sum_{i=1}^{n} \beta_i x_{ji} = \log(\lambda_j)$. It is important to note that these equations might not have analytical solutions in all cases. Therefore, numerical methods will often be needed.

3.2.5 Interpretation of GLM coefficients

The primary advantage of GLMs is the easy interpretation that can be obtained from the coefficients $(\hat{\beta}_0, \hat{\beta}_1, ..., \hat{\beta}_n)$ once they are found. This subsection will focus on the case of Poisson-based GLMs because that is the type of model used for the predictions in 5.1.2. The output of this model for an input vector $\bar{x}$ is

$$y_{pred}(\bar{x}) = e^{\hat{\beta}_0 + \sum_{i=1}^{n} \hat{\beta}_i x_i} = e^{\hat{\beta}_0} \prod_{i=1}^{n} e^{\hat{\beta}_i x_i}. \tag{3.16}$$

Given the properties of the exponential function, the following interpretations can be attained according to the data type:

- For a numerical or binary variable x_{i_0}, it will have no effect on the outcome if $\hat{\beta}_{i_0} = 0$, as the exponential of 0 is 1 and then the prediction is unchanged. The same reasoning explains that a positive change in x_{i_0} will make the prediction higher if the coefficient is positive and lower if the coefficient is negative. The exact repercussion on the prediction per unit of change in the variable will be a factor $e^{\hat{\beta}_{i_0}}$.

- For categorical mutually exclusive variables x_A, x_B such that one must be 1 and the other must be 0, the variable whose coefficient is greater will have a positive impact on the output. The difference ratio can be derived from equation 3.16 assuming all others variables remain constant as $\frac{y_{pred}(x_A=1, X_B=0)}{y_{pred}(x_A=0, X_B=1)} = \frac{e^{\hat{\beta}_A}}{e^{\hat{\beta}_B}} = e^{\hat{\beta}_A - \hat{\beta}_B}$.

Offering a straightforward interpretation makes the GLM a very interesting model. We can get from it both qualitative insight (if a variable makes the prediction higher or lower) and quantitative information (the actual amount of change it produces).

3.3 Neural network

Artificial Neural Networks (ANNs), usually simply referred to as neural networks, are models that can be described as "interconnected assemblies of simple processing elements, units or nodes, whose functionality is loosely based on the animal neuron. The processing ability of the network is stored in the interunit connection strengths, or weights, obtained by a process of adaptation to, or learning from, a set of training patterns"[24]. Before explaining these terms, it is worth pointing out that this idea is much older than one could think, since the principles of neural networks were first proposed in 1943 [35]. It is not the purpose of this dissertation to present a comprehensive history of ANNs, but this section will cover the key elements that configure a neural network.

3.3.1 Structure of an ANN

The most basic component of any ANN is the node, also called neuron. Its name comes from the fact that, just like the animal cell, it receives many inputs from other nodes but produces a single output. The mathematical expression that creates the output is called activation function [22]. Let's suppose that neuron i receives m inputs x_j. Then:

$$z_i = \sum_{j=1}^{m} x_j \, \omega_{ij} \quad \longrightarrow \quad y_i = f(z_i) \tag{3.17}$$

where ω_{ij} is the weight associated to the input x_j, f indicates the activation function and y_i is the final output of the neuron. There are many possible activation functions with different properties. Some of the most common ones are listed below:

- Threshold logic unit (TLU): $f(z) = H(z - T)$, where H is the Heaviside step function and T is a threshold value. It simply indicates if the linear combination z that works as input reaches a certain threshold. The TLU was the first activation function, introduced by McCulloch and Pitts in their already cited article [35].
- Sigmoid function: $f(z) = \frac{1}{1+e^{-z}}$. It allows for a continuous output that can be interpreted as a probability, since it stays between 0 and 1.
- Hiperbolic tangent function: $f(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$. Its S shape is similar to the sigmoid, but can produce negative outcomes because its range goes from -1 to +1. It is symmetric with respect to the origin.
- Rectified linear activation unit (ReLU): $f(z) = \max(z, 0)$. It returns the value of z if z is positive and 0 otherwise. Since this function is not smooth at $z = 0$, an approximation is often used in the form $f(z) = \log(1 + e^z)$. This is called the softplus function and it is easy to check that for big values of z it converges to $\log(1) = 0$ if z is negative and to $\log(e^z) = z$ if z is positive. Hence, it produces output values that are very close to the ones that can be obtained with the ReLU activation function.

The neurons work together and are organized in a network such as the one shown in Figure 5[4]. There are three distinct types of nodes in such a network depending on their location within the structure. The nodes in yellow are called input layer, since they represent the features that are given as information to perform the prediction. There is no activation function here, just transmission to the next layer. Nodes in blue and green are the hidden layers where the inner workings of the model take place. They may have different activation functions from one another. Finally, the node in red is the output layer, as its outcome is the final prediction of the network.

One may decide how many hidden layers there are, the number of neurons in each layer and the activation functions. However, once this architecture is chosen, those will be fixed elements.

[4]Image taken from https://towardsdatascience.com/applied-deep-learning-part-1-artificial-neural-networks-d7834f67a4f6

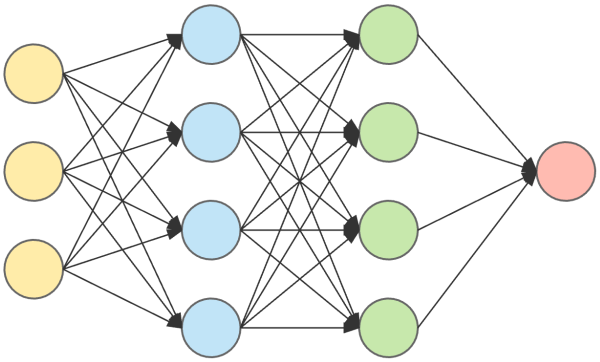


Figure 4: Neural network structure.

Thereupon, the flexibility of an ANN that allows for accurate predictions is present in the arrows that connect nodes. Arrows are in fact visualizations of the weights w_{ij} from equation 3.17. The role of these weights is assigning the relative importance of the output of a neuron when it becomes an input for the next one. Training a neural network is the process of working out the optimal values for the weights. Next section will be devoted to explaining how this is done.

3.3.2 Training: backpropagation

In order to adequate the outcomes of the neural network to the predictions that are needed, one must choose an error function $E = E(y, \hat{y})$ that depends on the neural network's prediction $\hat{y}$ and on the actual values y that should have been predicted. The goal of the training process is minimising this function over the training set, whose instances are labeled, by changing the values given to the weights. However, the vast majority of neural networks are too big to obtain the optimal values for the weights analytically. Therefore, numerical iterative methods are better suited for it. They will require the gradient of the error function with respect to each individual weight, which is worked out using a method called backpropagation [26] that is based on applying the chain rule as follows:

$$\frac{\partial E}{\partial \omega_{ij}} = \frac{\partial E}{\partial f(z_i)} \frac{\partial f(z_i)}{\partial z_i} \frac{\partial z_i}{\partial \omega_{ij}}. \tag{3.18}$$

The derivatives of the equation above can be worked out with slightly different results depending on the layer where the neuron i is located. The middle term will always be the same, since it is the derivative of the activation function evaluated with the current weights. For the last term, it filters the input of the neuron that comes with the weight whose gradient is being evaluated. Consequently, if the neuron is in the first layer, it is just the feature with weight w_{ij}, so x_j. If the neuron is in a different layer, the inputs come from a previous neuron. Then, in this case $\frac{\partial z_i}{\partial \omega_{ij}} = f(z_j) = y_j$. Finally, the first term is very simple if neuron i belongs to the output layer because then $f(z_i)$ is just the output of the network $\hat{y}$ and $\frac{\partial E}{\partial f(z_i)} = \frac{\partial E}{\partial \hat{y}}$. In the contrary, if it doesn't, a recursive formula can be found to calculate it as a function of the derivatives of the error for the next layer L:

$$\frac{\partial E}{\partial f(z_i)} = \sum_{l \in L} \left(\frac{\partial E}{\partial f(z_l)} \frac{\partial f(z_l)}{\partial z_l} \frac{\partial z_l}{\partial f(z_i)} \right) = \sum_{l \in L} \left(\frac{\partial E}{\partial f(z_l)} f'(z_l) \omega_{li} \right) \tag{3.19}$$

At some point of the recursion the next layer L will be the output layer, so the numerical evaluation will always be possible. Having seen that the derivative of E with respect to ω_{ij} can be obtained, we can discuss algorithms that use it to find the optimal weights. The most popular one is probably gradient descent [6], which goes by the expression

$$\vec{\omega}(t+1) = \vec{\omega}(t) - \alpha \nabla \vec{\omega}(t), \tag{3.20}$$

where α is the learning rate and t indicates the number of updates. This algorithm works by moving the weights towards the direction of greater descent, as it takes the negative of the gradient

to choose the direction at each step. There is, however, a crucial flaw to this method. If the error function is not convex, there might be points where the gradient is zero but the global optimum has not been attained. The image below[5] shows how this can happen for local minima and saddle points.

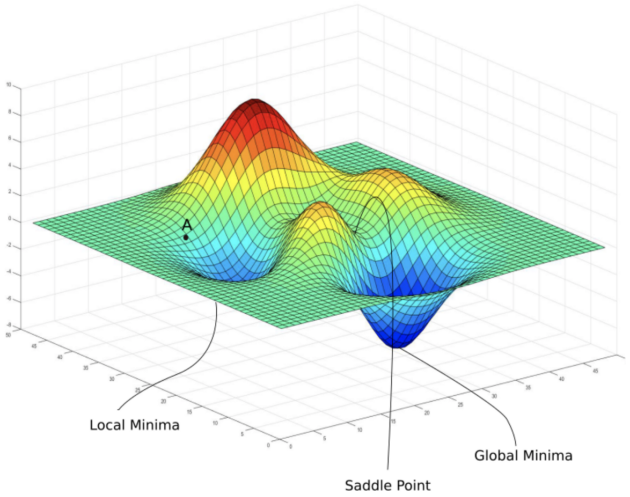


Figure 5: Examples of local minimum and saddle point.

Since checking the convexity of functions in large dimension may be difficult, other methods have been developed to avoid this problem. They usually involve combining several gradients at each step. A good example of such idea is the Adam optimizer. In this case, the update rule [31] is

$$\vec{\omega}(t+1) = \vec{\omega}(t) - \alpha\vec{m}(t), \tag{3.21}$$

with $\vec{m}(t)$ being a moving average parameter of the form $\vec{m}(t) = \gamma\vec{m}(t-1) + (1-\gamma)\nabla\vec{\omega}(t)$. The value of $\vec{m}(0)$ is usually initialized to 0 and γ is just a constant that can be chosen to give more or less importance to the new gradients.

Finally, let's discuss briefly how many updates of the weights are performed. Sometimes, a stopping rule is set so that the weights are not updated if the change in the error is very little for a certain number of steps. As discussed above, this can actually be counterproductive near local minima or saddle points. Therefore, very often numerical methods update the weights according to two factors chosen by the programmer:

1. Number of epochs: how many times the network goes through the training set.

2. Batch size: the number of training instances that are considered to work out the error each time.

Thus, if a training set has 100 instances and the batch size is 25, each epoch will have 4 weights updates. Hence, if the number of epochs is fixed at 10 the weights will be updated $4 \times 10 = 40$ times.

All things considered, ANNs usually achieve very high predictive performance. However, this comes at the cost of losing intuitions of the reasons that lead to those predictions, since the fitting process is computationally complex and there are a huge number of interconnections between the neurons which are arduous to follow.

[5] Image taken from https://towardsdatascience.com/applied-deep-learning-part-1-artificial-neural-networks-d7834f67a4f6

3.4 Which is the best model?

The question chosen as heading for this section does not have an ultimate answer. One should bear in mind that predictive models do not possess intrinsic value. A neural network might be more complex and computationally demanding because of the number of layers it has, a KNN regressor could be considered witty if it applies a peculiar definition of distance. However, none of these things make the model necessarily better. After all, models are just mere tools implemented in order to achieve a goal. Therefore, how "good" or "bad" they are depends largely on the task they will be used for and, even more importantly, how success for that task will be defined and measured.

The ordinary way of defining success of a predictive model is through *performance*, which is usually summarized as a number. For a classification task, this number can be the accuracy (percentage of correct predictions out of the total number of predictions made) or some other more specific evaluation metrics depending on the purpose of the predictions, such as true or false positive rate, sensitivity, specificity, recall, F-measure... A proper definition of these concepts can be found in [29], chapter on Performance Metrics with a Single-Class Focus. In the case of regression, there are also several options to quantify how far from the actual values the predictions are. The main metric that will be used in this dissertation is the mean absolute error (MAE), which is defined as follows.

$$MAE = \frac{1}{N}\sum_{j=1}^{N}|\hat{y}_j(\bar{x}_j) - y_j|, \tag{3.22}$$

where we consider a set of N predictions to be made using variables $\bar{x}_j$ and the absolute differences between the predictions $\hat{y}_j(\bar{x}_j)$ and the actual values y_j are averaged. Another metric that will be considered later is the mean squared error (MSE). Using the same notation, we have:

$$MSE = \frac{1}{N}\sum_{j=1}^{N}(\hat{y}_j(\bar{x}_j) - y_j)^2. \tag{3.23}$$

Note that in both cases the deviation from the actual prediction is counted towards the average as positive, no matter it is above or below the actual value.

There are some cases, though, where pure performance is not the only element that matters. This might sound strange: is it not the whole point of a predictive algorithm make predictions as accurate as possible? Let's consider two scenarios to illustrate that things can get complicated. Firstly, imagine that a logistics company stores different kinds of construction materials in a massive warehouse. The vast size of the warehouse makes access to some of its parts difficult, since it is often required to move certain items in order to reach others. In this situation, the logistics company might think of creating a machine learning model that predicts which materials will be bought from them and organize the warehouse accordingly. Therefore, they will be interested in maximizing the performance of the model as much as possible to improve their ability to assist their clients quickly.

Some other applications of machine learning involve more human-related decisions with ethical implications. Let's now consider a surgeon who has to operate on a difficult brain tumor. He can use the help of a machine learning model that, based on medical images of the patient, determines the best locations to perform the incisions on the skull. Even if past history shows that the model outperforms most doctors at this task, it is not perfect and sometimes the indications it provides lead to permanent injury or death of the patient. What should the surgeon do? It is fair to say that, given that the model has been better than human doctors in the past, you could take the risk of following its instructions. However, if the surgery goes wrong, who is responsible for the failure? The doctor, who was just doing what the model said? The programmer who designed the algorithm? Maybe more importantly, can we learn from a mistaken prediction to make sure it does not happen again? These questions will be solved if the model's output is not only a solution for the task (in this example, the

surgery instructions) but also a certain explanation of how it arrived to that solution. This way, the surgeon will be able to take an informed decision instead of just doing blindly what the computer says and mistakes will be easier to catch, understand and avoid.

This concept of being able to get from the model an explanation on how the results were obtained is usually called *interpretability*. A popular definition of interpretability is the one proposed by Doshi-Velez and Kim [16]: ability to explain or to present the model's output in understandable terms to a human. Several categories for interpretability are often established depending on the level of domain of expertise of the human who interprets the model. Some authors consider that there is another slightly different term that can be used to talk about this: explainability [33]. It is not associated to the meaningful insights or intuitions a human can extract from the output, but rather with the extent to which the internal logic and mechanics inside the machine learning system can be deeply understood. Linardatos et al. (again, [33]) point out that "an interpretable model does not necessarily translate to one that humans are able to understand the internal logic of or its underlying processes. Therefore, regarding machine learning systems, interpretability does not axiomatically entail explainability, or vice versa." A common —and, in my opinion, fair— critique to these terms is that they are not mathematically formal nor rigorous enough [4].

My dissertation will use only the term "interpretability" in a general way to compare how much information about the decision process can be obtained from each of the models under study and how that should be taken into account when deciding to use one or the other. This approach is aligned with the ideas expressed by the European Comission on the White Paper on Artificial Intelligence [13]. This document shows concern about the potential dangers of opaque (also known as black box) decisions taken by AI algorithms, which might replicate existing biases, intrude in citizens' private lives or even be used on purpose for criminal activities. All in all, a certain balance between performance and interpretability has to be found. This thesis will tackle what that balance should be for traffic accidents prediction.

Obviously, the best possible solution to the performance vs. interpretability question would be not choosing: obtaining great performance with a very clear interpretation of the workings of the algorithm. This problem is far from solved to this day, and there are ongoing efforts [42] to make deep neural networks' decisions understandable for humans. It is out of the scope of this thesis to analyse how this should be done, but it will for sure be a very exciting area of research in the foreseeable future.

4 Data analysis and feature engineering

The purpose of this section is to perform a deep inspection of the data available. Very often, when dealing with a data science problem, the main emphasis is placed on the machine learning models implemented. There are several motives for this to happen, as selecting and tuning the models is usually intellectually stimulating and they produce the final results. However, I consider the the previous analysis just as important. Many of the ideas about what hypothesis to test with those models arise after careful scrutiny of plain data. In addition, the way in which features are organized is not neutral: it will have an impact on the performance of the predictive algorithms and then there must be a solid rationale behind the decisions taken in this regard. Because of all of these reasons, the following epigraphs will contain detailed descriptions of all the steps that convert the raw data into the final training and testing sets which will be used in section 5.

4.1 Data sources

The UK government offers online a very large amount of datasets with different informations of public interest. Traffic is not an exception for this rule. The datasets used to tackle the problem of interest will be the ones that report accidents[6]. Those selected corresponded to years 2016, 2017, 2018 and 2019. 2020 was not included because the behaviour of traffic that year was very different to normal as a consequence of the Covid-19 pandemic. 2021 was not included because, although traffic at that time was probably very close to normal, the data for that year is still provisional and incomplete. There are more files available containing data from years further in the past, but the format is not exactly the same and consistency of the years selected is considered a great advantage. Finally, there is a very interesting dataset called "Road Safety Data - Accidents 1979 - 2020". Analysing it would have been interesting, but cars and traffic have changed a lot in the last forty years, so probably the conclusions would have not been as important for the future.

One of the features provided by the accidents datasets is weather at the time of the event. However, section 4.3.2 will require consulting weather at different past times than the ones given there. In those cases, the information will be extracted from climate-data.org for rainfall days per month and from weather-atlas.com for snowfall days per month. Both websites include an internal search engine to select the cities of interest.

In the end, all the datasets used are public and can be accessed by anyone with an Internet connection. This is a great advantage of the project because my results might be replicated and tested without any special requirement or permission from a third party.

4.2 Descriptive analysis of UK accidents

As stated above, the first step when dealing with big amounts of data is performing some exploratory analysis to gain insight into the information at hand.

4.2.1 Datasets structure

Given that accidents datasets from 2016, 2017, 2018 and 2019 share the exact same layout, they were combined into a single dataset. This initial dataset contained $506,774$ instances of accidents with 35 features for each of them. Including here a detailed description of each feature would be long and not very interesting for the task ahead. If the reader wants to know more about these features and their value range, appendix A contains a large table with further information. The only consideration to be made about the data is that the values "NULL" and -1, which do not appear in the table of the appendix, are used across the dataset to indicate missing information.

[6] They can be downloaded from https://www.data.gov.uk/dataset/cb7ae6f0-4be6-4935-9277-47e5ce24a11f/road-safety-data by selecting the CSV files named in the "Road Safety Data - Accidents YYYY" fashion, where "YYYY" is the year.

As this setting contains a substantial amount of data, the initial goal is to reduce the number of features so that only variables of interest remain in the dataset. After some consideration, the variables kept were: 'location_easting_osgr', 'location_northing_osgr', 'date', 'day_of_week', 'time', 'first_road_class', 'road_type', 'speed_limit', 'weather_conditions', 'urban_or_rural_area' and 'accident_severity'. Although table 5 in the appendix offers the basic information about these variables, some comments about them are appropriate here. The coordinates kept for location purposes are the ones corresponding to the OSGR system, a location system put in place by the Ordnance Survey of Great Britain (OSGB). The main advantage of this system is that all numbers are positive, which is not the case for the usual longitude since Greenwich meridian goes through the UK. Moreover, for some reason decimal places in latitude and longitude were not well interpreted by Excel on my computer, so sticking to OSGR allowed me to avoid wasting time on formatting. When required, there is a Python library that can convert OSGR coordinates to latitude and longitude smoothly. It is also worth mentioning that several changes were introduced in the features:

- Columns indicating date and time, that are strings, were combined into a feature called "datetime" in a Python DateTime format.
- There were many categories for weather, but they were simplified into only three: fine, rain and snow. This division does not account for the presence or absence of high winds and includes fog inside the rain category.
- "Rural area", that was coded as 2, was changed to 0. This makes the feature adequate for the one-hot encoding approach that will be applied to the whole dataset in section 4.3.1.
- The column 'accident_severity' will be taken in the following as the target to be predicted. Section 5 includes an explanation on the numerical encoding used for this purpose.

Furthermore, let's explain briefly the rationale to exclude the rest of the variables. Some of them (reference number of event, road number, number of casualties, whether the police arrived fast or not...) are just not useful to predict if an accident will occur. Others (light conditions, special conditions, road surface conditions...) are "normal" most of the time and/or overlap with the weather feature that is already considered. Finally, the dataset also contains several variables that refer to information about a secondary road in the case of the accident taking place within 20 metres of an intersection. Although they could be useful for this specific type of accidents, they do not apply to many others, so ignoring them seemed the best option to keep the feature space relatively small and coherent for every accident.

All in all, this phase of the feature selection process was aimed at taking those variables that are conceptually simple and easy to handle. Once selected, yet another perk of the process was that they did not have many missing values in comparison to some of the discarded ones. Therefore, a complete case analysis will be considered from now on. Filtering for instances where all the information was available reduced the number of instances of the dataset to $473,982$. Since this number is close to half a million and represents over 93% of the accidents initially at our disposal, it was considered enough to undertake the problem and no imputation method was performed over the instances with missing values, that were just discarded.

4.2.2 Some statistics

Let's now dive into the numbers of the features that we have just chosen to get an idea of the way our accidents are distributed. In order to do so, visualizations can be extremely useful to get a more intuitive idea of the stories numbers tell. The reader should keep in mind that the statistics that are about to be displayed come from aggregating all the accidents in the period 2016-2019. Therefore, one must be careful when jumping to conclusions. Certain trends of interest will be highlighted and several of them will justify some aspects of the preprocessing (section 4.3) and splitting (section 4.4) that will take place afterwards. However, full conclusions will not be achieved and discussed in detail until modelling results are obtained.

The first characteristic to point out about the data on accidents that is available is how imbalanced the distribution is when it comes to severity. This is a logical consequence of the fact that fatal accidents do not happen as often as minor ones. To be precise, we only have $6,409$ examples of fatal accidents, so about 1.35% of the total number of reported accidents. Serious accidents are more common, we have $85,499$ of them, which is slightly over 18%. Finally, there are $382,074$ slight accidents, where no significant injury occurred to anyone involved. They account for a bit over 80% of the dataset. A visual representation of these figures is provided with the histogram of Figure 6.

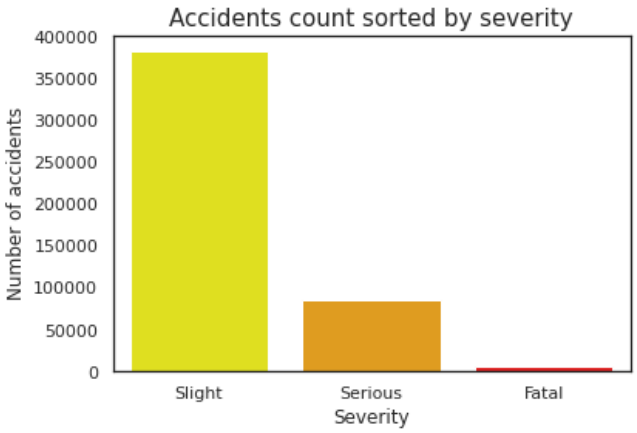
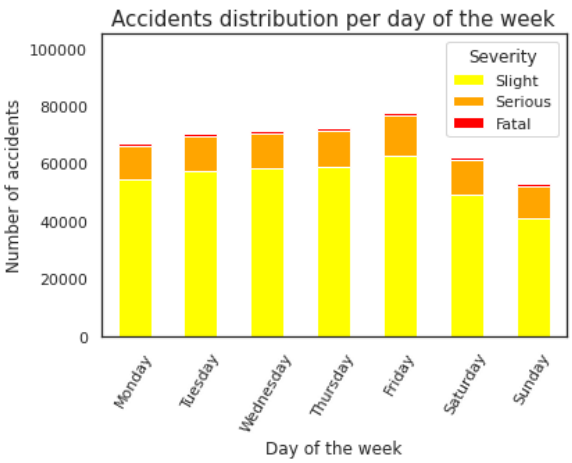


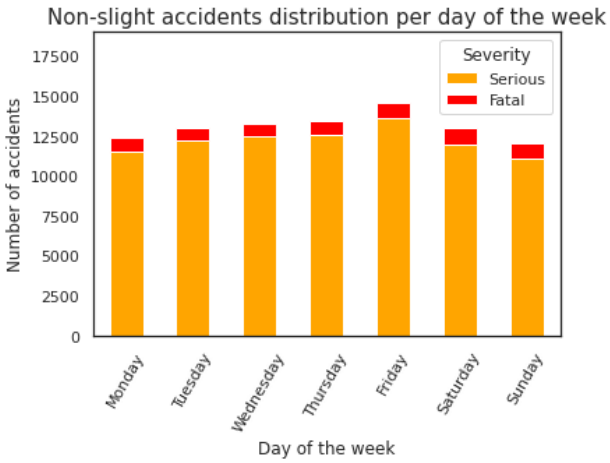
Figure 6: Histogram representing number of accidents by severity.

The red bar representing fatal accidents is nearly impossible to see. We will come back to this later, as this distribution will have a deep impact on the predictions. With so few examples of fatal accidents, our algorithms will be biased towards predicting low values for severity. We will have to take this into consideration when evaluating the performance of the models.

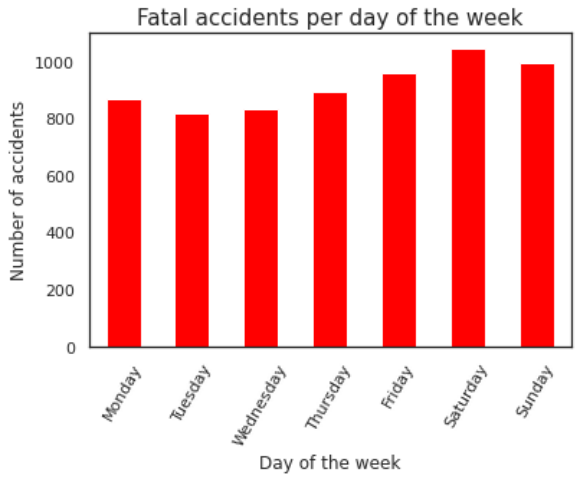
We will examine now how severity changes depending on the day of the week considered. There are three histograms shown on Figure 7 that will be quite useful in this endeavour. The first one of them, labeled (a) and located in the top left position, displays all types of accidents with their severity indicated by colour. As before, yellow is used for slight accidents, orange indicates serious ones and red is reserved for fatal accidents. We can see that the total number of accidents remains stable during weekdays, peaking on Friday. However, then it decreases quite abruptly on Saturday and Sunday. This disposition of accidents seems to follow a simple pattern: since people drive less often at weekends because they do not usually have to go to their workplace, accidents are lower than during the week. Nonetheless, if we zoom in and restrict the same graph only to serious and fatal accidents, the situation changes. A look at the second histogram, located in the top right position, shows that in this case all days of the week have a similar amount of accidents. Friday remains the most accident-prone day, but the other days have a similar occurrence of accidents. Finally, the third histogram completely confirms the tendency we have guessed from the first one. When only fatal accidents are plotted, it is actually during the weekend when the most events of this kind take place and Saturday is the "most dangerous" day.



(a) All kinds of accidents



(b) Serious and fatal accidents



(c) Only fatal accidents

Figure 7: Distribution of accidents on record by days of the week.

In order to dig into this finding, let's plot the ratio of fatal accidents with respect to the total number of accidents each day. This number represents and answer to the question "If there is an accident on this day of the week, what are the odds of it being fatal?" Figure 8 answers this question. About 12 in every thousand accidents have someone killed during weekdays, but that number skyrockets to nearly 17 on Saturdays and 19 on Sundays. It is difficult to draw an unambiguous conclusion from these graphs, but the main takeaway we can have is that accidents patterns differ quite a lot between weekdays and the weekend.

The week is not the only time unit we are interested in to categorize accidents. The time of the day when they occur can also be very helpful. A density plot is presented on Figure 9 to visualize how each different type of accident is distributed across the day. It is pertinent to point out that the normalization has been done for every type of accident separately, which implies that direct comparisons cannot be done: the fact that at 12 pm the three plots merge does not mean that there are the same amount of slight, serious and fatal accidents at that time of the day. Since the total number of slight accidents is greater, the absolute number of slight accidents will be bigger at that time. The conclusion that we can actually derive from that 12 pm intersection that the graph presents is that the relative importance of that time with respect to the other hours of the day is equal for the three severity cases under study.

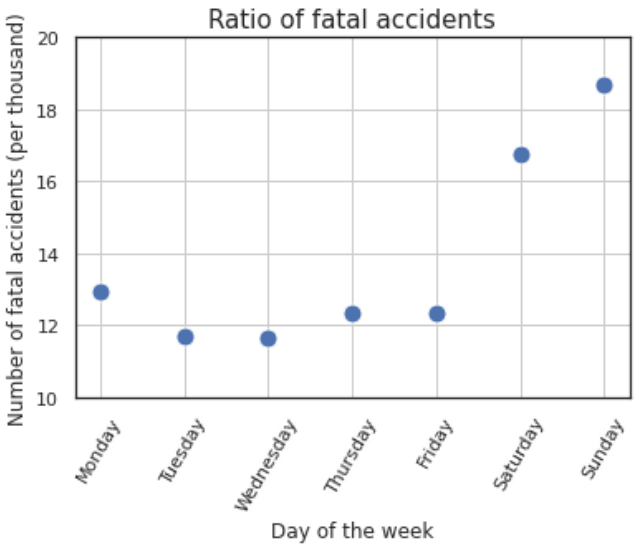


Figure 8: Number of fatal accidents for every 1000 accidents of any severity.

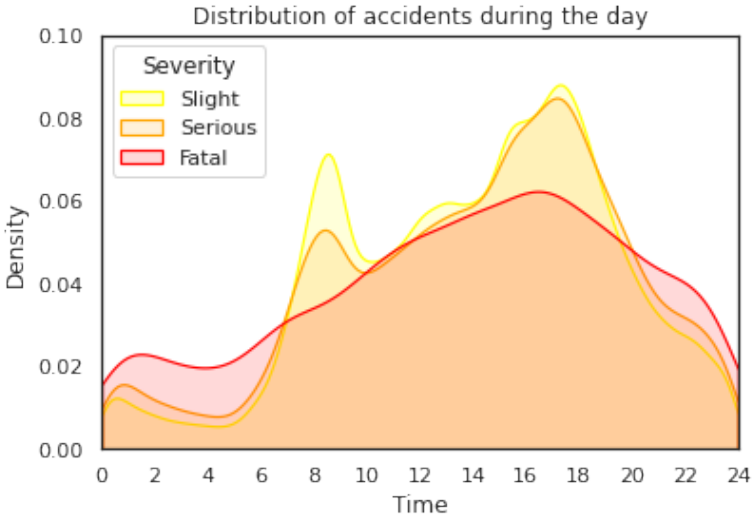
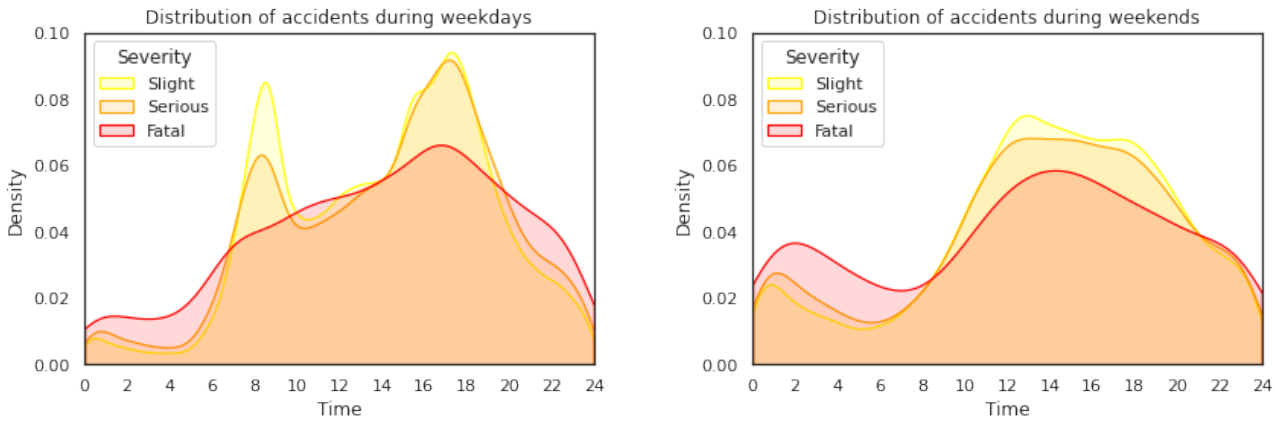


Figure 9: Distribution of accidents along the day.

The density distribution graph shows quite different patterns depending on the severity of the accident. While slight and serious accidents (depicted in yellow and orange as in previous plots) follow similar trends, fatal ones (in red) are on their own. We can see that fatal accidents are spread out in a close to uniform fashion along the day. There is a concentration of them from 2 pm to 7 pm, but it is not very extreme. What is rather surprising is that the density remains fairly high during the night despite traffic being quite lower at that time. In contrast, slight and serious accidents have two very clear peaks: one in the early morning and another one in the evening. These times coincide with the most common commuting hours. In fact, they are basically the same times proposed by Wemegah et al. [52] to model traffic. In order to test whether the peaks actually correspond to people commuting, the same density graph will be produced filtering for weekdays and weekends, as one could expect that commuting is not important for the latter, when most people do not work. These visualizations can be seen one next to the other on Figure 10.

The graphs are very clear and show a very different distribution of accidents for weekdays and weekends. Although this had been anticipated by Figure 8, the extent of the dissimilarity can be fully observed now. Weekdays have extremely high peaks in the morning and the evening. The morning peak completely vanishes during weekends and the evening one flattens to the point where it reassem-



(a) Accidents on weekdays

(b) Accidents on weekends

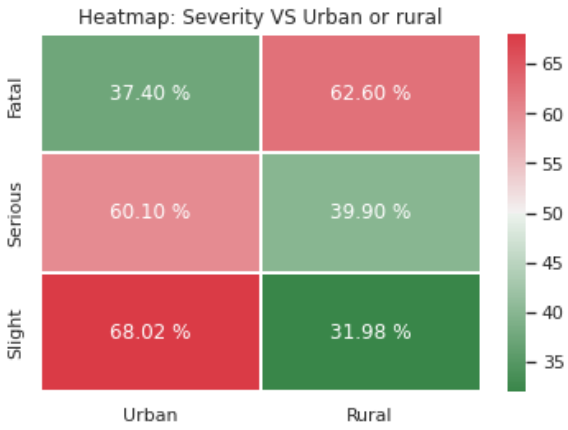
Figure 10: Distribution of accidents along the day grouping by weekdays (from Monday to Friday) and weekends (Saturday and Sunday).

bles a plateau instead of an actual peak, so the commuting hypothesis is confirmed. Another important feature to point out is that night accidents of no matter which level severity grow significantly over the weekend, which seems reasonable given that people go out more often those nights.

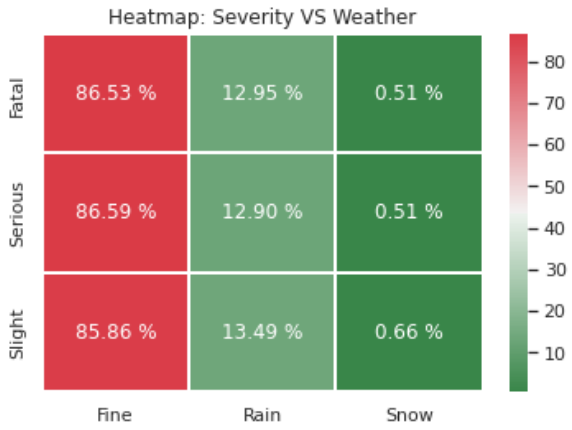
Taking into account the Wemegah article previously cited and the density graphs above, three categories will be established to include the information on time of the day in the feature space: peak hours (from 6 am to 9 am and from 4 pm to 7 pm), night hours (from midnight until 6 am) and off peak day hours (any other time, so from 9 am to 4 pm and from 7 pm to midnight). The main advantage of this approach is that it allows to convert times, which are always a tricky data type to consider, into a categorical variable with three possible values.

The last visualization tool that will be used in this section to explore accidents' features is the heatmap. It consists in a grid divided into different quadrants that looks at the relationship between two variables. Here, severity will be studied with respect to different available features. Each quadrant is colorized according to a scale so that it is easier to perceive at first sight where the important information is. The darker red the colour is, the more accidents are concentrated in that quadrant. It is crucial to note that all heatmaps displayed below are normalized across the horizontal axis. This means that the percentage shown in white numbers inside each quadrant refers to the accidents with the same degree of severity. For instance, in Figure 11 the graph in the left shows that 37.40% of fatal accidents happen in urban environments. The normalization was performed in this way because, as we have seen in Figure 6, the amount of accidents for each category is quite different and normalizing across the vertical axis would have made comparisons very difficult and colours less meaningful.

Figure 11 is indeed a perfect example of how some heatmaps are more informative than others. On the left we can see that urban and rural areas do not have the same accident patterns. The higher the severity of an accident, the more likely it is to have taken place in a rural environment. More than 60% of fatal accidents happen in rural areas whereas the proportion is below 32% in the case of slight accidents. However, the graph on the right does not provide much information. There are very few accidents during snowy weather, they account for less than 1% of the total number of registered collisions. In case of rain, we have that the figure hovers around 13% for all severity levels. All the rest, which are the bigger share, took place with fine weather conditions. Since we do not have direct information on the amount of time it rains or snows on a general basis, we cannot extract solid conclusions from the heatmap. Anyway, weather will be addressed again when looking at interpretability.



(a) Relationship between severity and location



(b) Relationship between severity and weather

Figure 11: Heatmaps of accidents severity distribution with respect to location and weather.

Another crucial element to consider is the speed limit of the road where accidents occur. That is why the heatmap on Figure 12 relates the severity of the accidents and the speed limits in miles per hour (mph).

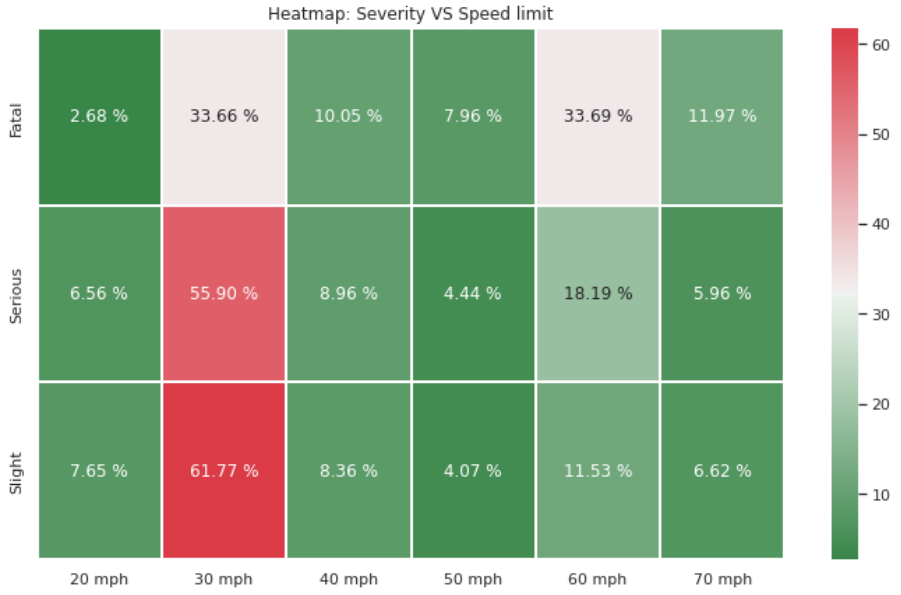


Figure 12: Heatmap of accidents severity distribution with respect to speed limit.

The graph above is too homogeneous to be clearly understandable. Most sections are coloured in green and they look very similar to each other. The only exception for this is the column that corresponds to 30 mph, which concentrates a large proportion of accidents. However, as this is the most common speed limit for roads inside cities, that colour basically shows that more accidents happen where there are more cars, which is not very surprising. In order to have a more insightful visualization of the data, the speed limits will be grouped in two classes: those strictly below 50 mph and those at or above 50 mph. This binary classification compresses the graph horizontally so that it becomes Figure 13.

In this case, a very strong tendency in the distribution of accidents can be seen. The information was already present in the previous graph, but a better choice of speed limits allows for an easier interpretation. We can see that slight and serious accidents happen much more often in roads where

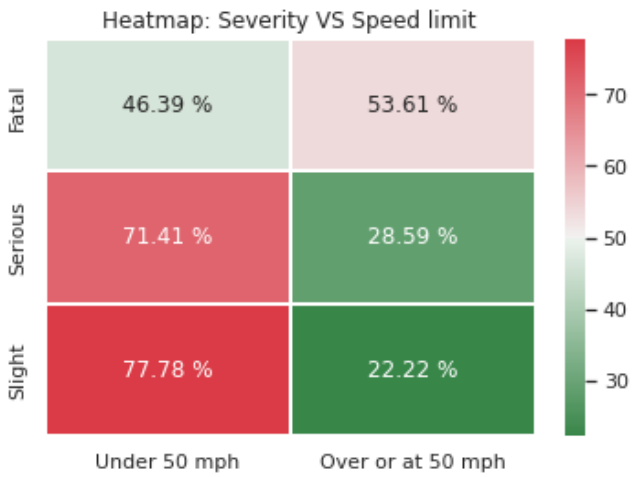


Figure 13: Heatmap of accidents severity distribution with respect to speed limit, only two categories.

the speed limit is below 50 mph. However, that is not the case for fatal collisions, which are more or less equally split, even with a slim advantage for higher speed roads. In fact, this graph is similar to the one already commented in Figure 11 about the difference between urban and rural areas. Speed limit makes an even greater distinction, since the share of non-fatal accidents on roads whose speed limit is below 50 mph is more than 70%. Thereupon, I took the decision to split the prediction problem in two depending on the speed limit of the road. This will be done by considering separate datasets in the data preparation that is detailed in the next section. Then, each model will be fitted twice: one time for low speed roads and one time for high speed roads.

4.3 Preprocessing data

The way data is formatted as input is important for it to be better suited for the task. This subsection explains how the datasets were encoded.

4.3.1 Simplification and one-hot encoding

A fact that stands out immediately when looking at the data available is that most of the features are categorical: location (urban or rural), day of the week, road class, road type, weather conditions and month. In order to keep certain homogeneity, it would be interesting to make the remaining variables (speed limit and time of the day) also categorical. This will be done with a different approach in each case.

The speed limit of a road is a number but the amount of possible values for it in the UK is limited to just six. Consequently, the feature will be made categorical by considering the figures as categories. On top of that, as it was anticipated in section 4.2.2, the clearly different dynamics of accidents with respect to the speed limit of the road considered encourages to divide the dataset between those accidents taking place where the speed limit is 20, 30 or 40 mph and those with associated speed limits of 50, 60 or 70 mph. For time of the day the situation is slightly more tricky, since the range of values is much wider. Although a viable option would be taking just the hour and discarding the minute, this would leave 24 categories, which seems too many for a single feature. Instead, following the findings of plotting the accident density per time of the day (Figure 9) and Wemegah et al. [52], each time will be assigned to one and only one of the categories “peak”, “off peak” and “night”.

Furthermore, a new variable will be added with the objective of studying the effect of a very specific yet potentially dangerous traffic situation. From my personal experience, when the sun is low at the horizon (right after sunrise or before sunset), it might blind you while driving. This can lead to great difficulty to see and react to the surrounding traffic. To track this, a binary variable called “Sunglare” will be equals to 1 if an accident took place within the 30-minute period after sunrise or

before sunset, and equals to 0 for any other time of the day. The main problem to define this variable is that the sunrise and sunset time is not fixed: it varies throughout the year and also depends on the location. The Python library Astral will be used to work out the exact time when these events occur. Details on the implementation of the function that gets the sunrise time can be found in appendix D. A completely analogous function was designed for sunset time and then the two were applied to each row of the dataset.

Lastly, and given that after the transformations outlined above all the variables are categorical (sunglare is binary, but that is not a problem as 1 can be seen as category Yes and 0 as No), it was decided to proceed with a one-hot encoding approach aided by the Pandas function *get_dummies*. This means that each variable will be split out into as many variables as categories it has. For example, the variable day of the week will become seven variables: day of the week_Monday, day of the week_Tuesday, day of the week_Wednesday... For each instance, all of these variables will be 0 except for the one that corresponds to the actual category. The main advantage of this layout for the dataset is that no variable is privileged as they all share the same structure. We have not mentioned here the accident severity, which is the variable we will actually be trying to predict using the others. The way severity will be encoded is actually very related to the modelling framework, so it is more appropriate to discuss it later.

4.3.2 Simulation of auxiliary data

The nature of the problem this dissertation tries to solve means that some crucial events are not present in the data. After all, we have examples of slight, serious and fatal accidents, but we lack information about something equally important: non-accident events. We need them to make sure there is not something present in all accidents that is not noticed precisely because of that. If we imagined, for the sake of simplicity, that all accidents happened on Mondays, our prediction models would not be able to tell that this is extraordinary if all the data they use as input comes from accidents (namely Mondays!). However, if we introduced in the model events depicting general traffic where severity is null (because there is no accident), then the models would realise that the day of the week is a driving factor behind the occurrence of accidents, since traffic during the rest of the days of the week behaves differently. Thus, the role of this section is explaining how auxiliary events where no accident occurred were simulated.

There is an important issue to address before designing the simulation process. In real traffic conditions, the proportion of accidents with respect to all car trips is minimal. However, we should not replicate this proportion in the data. If we do, we will encounter an extreme imbalance problem with the vast majority of instances representing non accidents. Therefore, a predictor that naively classifies any input as a non-accident would be right more than 99.9% of the time. In order to solve this issue, what will be done is simulating as many non-accident events as the total number of accidents in each case. This will reflect the fact that non-accidents are common, since the severity category with the highest frequency will be null, but at the same time it will not be an exaggerated difference so that accidents have a big enough weight for the model. A different proportion might have been used, as several numbers would have done the trick for this purpose. Ultimately, the actual number is not that important because the main goal of this dissertation is not doing the best possible prediction for the datasets created, but comparing the performance and interpretability of different models for the problem.

We can now proceed to describe how each of the non-accident events were generated. Firstly, a random datetime point in the period under study was created using the Faker library. From this datetime object we can extract the variables day of the week, month and peak/off peak/night feature. Secondly, we need a location for the event. One could think of taking random spatial coordinates, but that is actually quite problematic. Those coordinates will likely not fall on a road and, even if they could be assigned to the closest road, finding the speed limit or the road type would not be trivial. In order to create this information, the process will be based on selecting a random index from the

dataset containing accidents and retrieve the location information from one of them. By doing this, the non-accident will take the coordinates, road type, road class, speed limit and urban or rural area feature from a real accident event that happened at a different time there. The coordinates, along with the datetime object, are fed to the sunrise and sunset times functions to check whether the conditions for sunglare apply. Although this method may lead to certain locations being over-represented, the great dispersion of the accidents will help to avoid effect being too problematic. Lastly, we need to assign a weather to this event, which will be more complicated.

Obtaining historical weather is not particularly difficult. A quick Google search offers several websites that keep track of the total amount of rain per month over the main regions of the UK. Nevertheless, it is much more complicated to find weather information with the proper granularity: for very specific locations at specific times of the past. Given that the non-accident events being simulated here have an exact location, granularity in the weather is very important. It will be created by hand using the data sources commented in 4.1. In order to do so, the first step is to partition Great Britain into 18 distinct regions. This was done using OSGR coordinates creating square regions of approximately 100 km of side, which corresponds to 100,000 units in the OSGR. Note that the irregular shape of Great Britain makes it impossible to have square regions all along. The exact geographical boundaries that define these regions are shown in table 2.

Now, for every non-accident event that needs to be given weather conditions, two factors will be taken into account. On the one hand, the month of the event, since weather varies greatly throughout the year. Also, the region where the event is located will be considered, since the weather differs from some places to others. Finally, the simulation will include a stochastic element because the weather will be seen as a probability of either rain or snow following the expression:

$$Prob_{Rain}(m,r) = \frac{1}{ND(m)} \sum_{c \in Cities(r)} K_c N_{Rain}(m,r). \tag{4.1}$$

In the equation above m stands for month and r indicates region. $ND(m)$ is the number of days of month m and $N_{Rain}(m,r)$ refers to the average number of rainy days in region r during month m, which can be found in appendix B in table 6. The sum is performed over the cities that are used as reference for each region. These cities, as well as the coefficients K_c that are used in each case, are displayed in table 2. A completely analogous process will be used to work out the probability of snow. The only difference is that the number of snow days will be used for the calculations, they can be found in the same appendix in table 7. All other coefficients remain unchanged.

Region	Geographical limits				Cities for weather (coefficients)
	South	North	West	East	
1. North Scotland	748608	-	-	-	Inverness (0.5), Aberdeen (0.5)
2. South Scotland	648608	748608	-	-	Edinburgh (0.5), Glasgow (0.5)
3. North England	548608	648608	-	-	Glasgow (0.2), Moffat (0.5), Newcastle upon Tyne (0.3)
4. Middlesbrough area	448608	548608	-	-	Newcastle upon Tyne (0.2), Middlesbrough (0.4), Penrith (0.4)
5. Manchester area	348608	448608	-	433863	Manchester (0.6), Liverpool (0.2), Sheffield (0.2)
6. Leeds area	348608	448608	433863	-	Leeds (0.5), Sheffield (0.5)
7. North Wales	248608	348608	-	333863	Wrexham (0.8), Birmingham (0.2)
8. Birmingham area	248608	348608	333863	433863	Birmingham (1)
9. Nottingham area	248608	348608	433863	533863	Nottingham (0.7), Peterborough (0.3)
10. Norwich area	248608	348608	533863	-	Peterborough (0.2), Norwich (0.8)
11. South Wales	148608	248608	-	333863	Cardiff (0.6), Swansea (0.4)
12. Bristol area	148608	248608	333863	433863	Bristol (1)
13. West London area	148608	248608	433863	533863	London (0.8), Reading (0.2)
14. East London area	148608	248608	533863	-	London (0.7), Colchester (0.3)
15. Cornwall	-	148608	-	333863	Exeter (1)
16. Bournemouth area	-	148608	333863	433863	Bournemouth (0.8), Bristol (0.2)
17. Southampton area	-	148608	433863	533863	Southampton (0.6), Brighton (0.4)
18. Dover area	-	148608	533863	-	Dover (0.4), Brighton (0.6)

Table 2: Region division information for weather simulation.

A methodological note about these coefficients is appropriate. They have been selected so that they add up to 1 for every region, thus the interpretation to be made is that they represent the weight of the city they are assigned to with respect to the whole region. Then, if the city is located near the centre of the region and there are no other cities nearby, its coefficient will be high (or even 1, as it is the case for Bristol). On the contrary, in the event of a region where several cities are not located in a privileged position, the individual coefficient for each of them will be relatively low (a good example of this is region number 4, Middlesbrough area). Using this procedure to estimate the weather implies that the cities whose weather data are considered become over-represented. However, since accidents take place more often in those major cities and their surroundings, the effect will not be too exaggerated. On top of that, as it was previously mentioned, we will be obtaining better granularity, which would be very difficult otherwise.

4.4 Splitting into training, validation and test set

Something to note when analysing the dataset that contains the accidents is that their spatial distribution across the country is quite homogeneous. There is not a place with a particular high or low share of accidents once you factor in that big cities tend to accumulate them because of their greater population and traffic. This feature brought about the possibility of selecting a certain location and use it as training set. Normally, a procedure of this kind would not be advisable because there is a risk of overfitting the model to the characteristics of a specific subset of the data. However, in this case there are also potential perks related to government's traffic safety plans that will be discussed in more detail in the conclusions.

In the end, it was decided to use Manchester area (number 5 in table 2) as training set because it included both urban and rural areas and the metropolitan area of Liverpool, so it could be fairly representative of the whole country. This implies that the training set contains 18818 cases of high speed events, which accounts for 8.34% of the total number of events, and 87968 cases of low speed events, so 12.18% of them. A validation set is also required to select certain key hyperparameters for the models. It was created by taking from the whole dataset for every type (high or low speed) the same number of examples that are in the training set. Finally, the whole dataset was considered as test set. It has one order of magnitude more instances than the training set, which contains 225582 high speed and 722382 low speed events.

5 Modelling results

Once the available information has been studied and formatted in a way that makes it easier to use, the actual problem-solving can begin. The aim of this section is providing a series of well-justified answers to the questions posed in 1.1.

In order to apply any of the models previously discussed, it is key to clearly define the output we will be targeting. This output will be the severity of the accident, that has four different categories: non-accident, slight, serious and fatal. Although the problem might be considered from a classification approach (as done in [38]), I believe going with a regression formulation makes more sense. After all, the severity degrees create a sort of scale where non-accident and fatal are the extremes. It is quite logical to claim that predicting a fatal accident as serious is a mistake, but not as bad as predicting it as a non-accident. Classification metrics do not catch this idea, but we will be able to take it into consideration with a regression.

Fitting a regression will require assigning numerical values to the accident severity categories. The choice for this is relatively free, but it must satisfy two conditions: representing the danger each type of accident involves and avoiding the existence of trivial solutions that provide good performance. The scale proposed in this thesis gives value 0 to non-accidents, 1 to slight accidents, 5 to serious accidents and 10 to fatal accidents. A higher number will therefore imply a greater severity. Slight accidents were given a value relatively close to 0 because, although they might disrupt traffic, they usually do so for a short period of time and their impact on people's health is low or nonexistent. In contrast, serious and fatal accidents have devastating consequences on people. Moreover, this scale does not encourage trivial solutions. If we predicted either no accident (0) or slight accident (1) for every event, the MAE would be 0.875 in the case of low speed accidents and 1.070 for high speed accidents. Both these figures are quite high when compared to the errors that will be obtained from the algorithms.

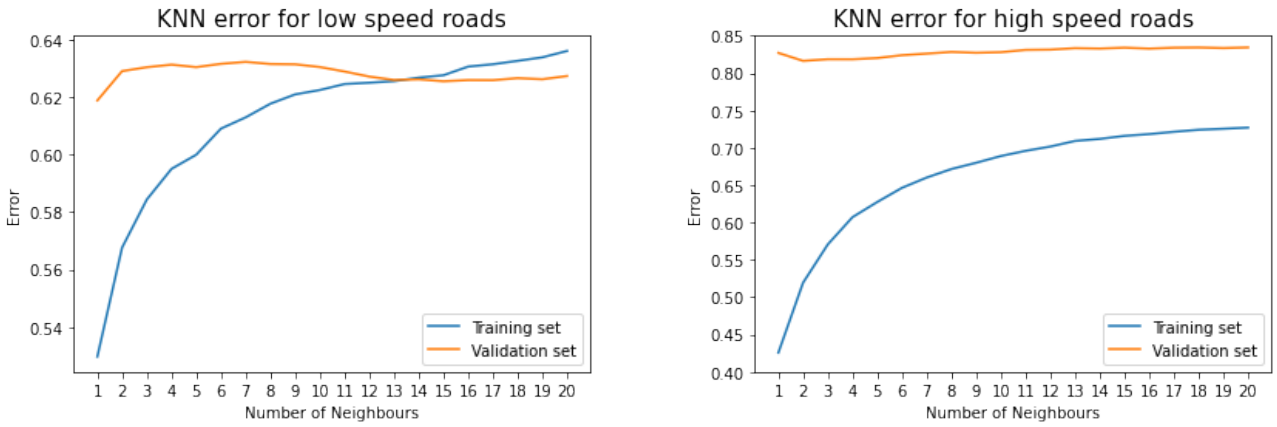
5.1 Applying the different models

In the following, the three models proposed are applied in the same order in which they were explained in section 3. Each of them will be trained on Manchester area data and then applied to the whole Great Britain. A distinction will be made for low speed and high speed roads, so two final results will be given for every case. In this section only the mean absolute error will be considered.

5.1.1 How many neighbours are optimal?

K-nearest neighbours is the model that will be used as baseline. Results are expected to be relatively good because it is a non-parametric approach. Since the distance metric has been fixed to Euclidean distance for simplicity, the only decision to be made is the value of k, in other words how many surrounding examples will be taken into consideration to work out the regression value of a prediction. As a tool to choose k, Figure 14 was plotted. It shows the error of the prediction performed on the training set and the validation set for a set of values of k ranging from 1 to 20.

The training set error grows with the number of neighbours, but the performance of the algorithm on the validation set is quite stable for both low speed and high speed roads. There are arguments to take $k = 1$, as this case shows the smallest overall error, but then the risk of overfitting is important. Therefore, and using as main reference the validation error on low speed roads, the algorithm will be implemented with $k = 15$.

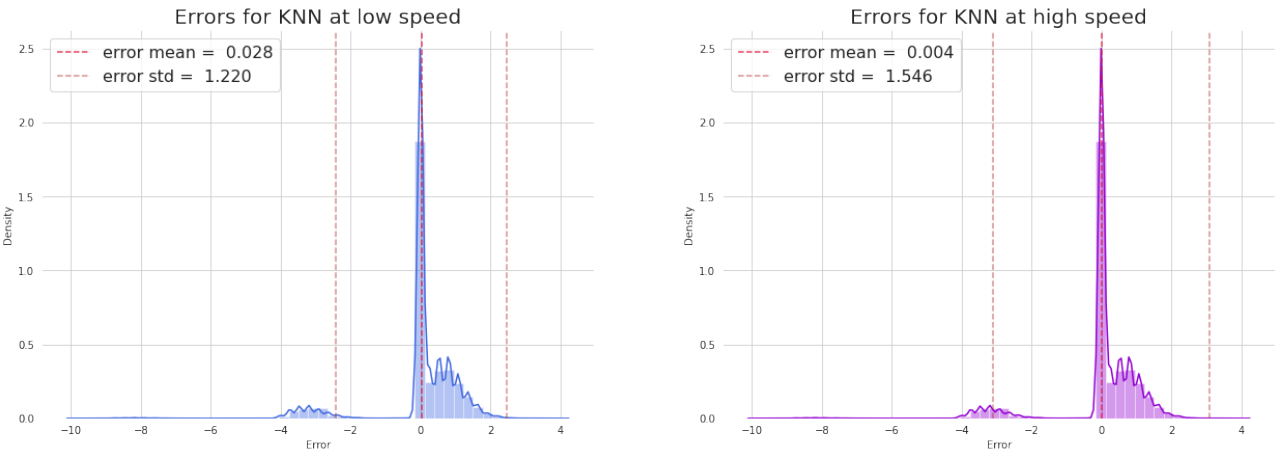


(a) Low speed roads

(b) High speed roads

Figure 14: KNN predictions error on training and validation sets for different numbers of neighbours.

Applying the KNN model to the prediction problem with $k = 15$ produces a MAE of 0.62920 for low speed events and 0.83787 for high speed events. Figure 15 shows how the errors are distributed in each case. Here the absolute value is not considered, just the difference between the predicted severity and the actual one.



(a) Low speed roads

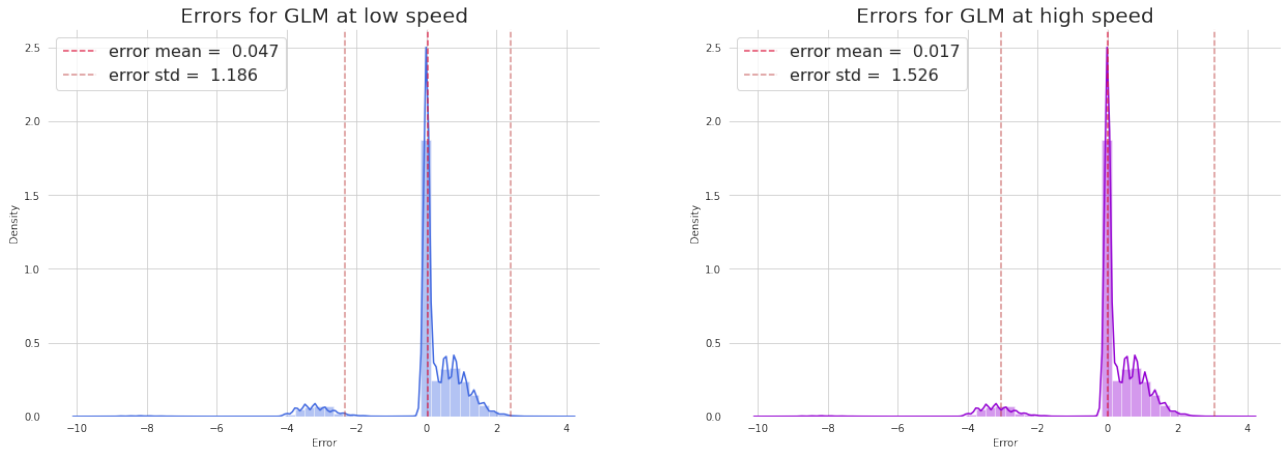
(b) High speed roads

Figure 15: Error distribution for predictions performed with KNN.

Something to point out from the figure above is that the error distributions are quite similar, so they can be commented on together. As the legends of the graphs show, the mean of the error is very close to 0 and this is also the most common error, meaning that the prediction has been exact (or very close to exact) quite often. There are two other noticeable bumps in the distribution around the central peak. The one on the right represents those cases where there has been an overestimation of severity by up to 2 points, so for example a non accident classified as a slight one. On the left there is a smaller but still clear bump in the area of -3 or -4, which corresponds to instances whose severity has been underestimated. All in all, the relative height of these mistakes is little so the performance can be considered good. It shows an improvement of about 25% with respect to the naive approach where all predictions are 0 or 1, and this is quite remarkable given how imbalanced the distribution of accidents is.

5.1.2 Insights from the linear approach

For the generalized linear model, it is necessary to pick a distribution and a link function. Given the accident count shown in Figure 6 and the number of non-accidents that have been simulated, the most reasonable approach is going with a Poisson distribution because it fits the shape of the accidents distribution. The link function will be the canonical logarithm because when other link functions were tried they did not respect the prediction boundaries and that led to substantial errors. Using this Poisson-based GLM on our problem generates a MAE of 0.64560 for low speed events and 0.85037 for high speed events. The distribution of the errors can be seen on Figure 16.



(a) Low speed roads

(b) High speed roads

Figure 16: Error distribution for predictions performed with GLM.

The conclusions that can be extracted from the graph above are very similar to the ones already commented for the analogous visualization when predicting with KNN. The errors are centered around 0 and the position of the bumps is basically the same. Achieving a performance that is comparable to KNN with a parametric algorithm is a noteworthy result. On top of that, the main advantage of GLM is that we can analyze the coefficients that were assigned to every variable. Two tables can be found in appendix C where for each feature there is a coefficient, the standard error of that coefficient and a 95% confidence interval for it. The last item of the table "const" is not a variable but the constant β_0 that works as the intercept for the regression.

There is a clear pattern that both tables share: weekend days are related to a higher severity than weekdays. If we take the average coefficient for Saturday and Sunday and compare it to the average for Monday to Thursday, there are interesting results to show. For low speed roads, the severity increase is above 5% because $e^{-2.3259+2.3795} = 1.055$. For high speed roads, this gap grows to more than 11% because the analogous calculation results in $e^{0.7459-0.6393} = 1.112$. Something else that stands out is the greater importance of the speed limit in high speed roads, whose coefficients for the variables related to speed limit are all larger than 4 in table 9, than in low speed roads, since those same coefficients hover around 2.5 in table 8. Although a similar reasoning could be applied to road class and road type, one should be cautious because certain classes such as roundabout or 1-way roads are not as represented as others. It is also worth pointing out that certain factors do not seem to play a significant role for accident severity: all months and times of the day have similar coefficients and urban or rural area and sunglare are close to zero. This last variable was the one introduced in 4.3.1 and, unfortunately, the hypothesis that sunglare can increase accidents does not hold, at least for the data available. Finally, some remarks about the role of weather are required. The results for both datasets indicate that, the better the weather, the higher the predicted severity will be. This does not seem very intuitive. For the case of snow, the conclusion may be deemed not significant since there are only 455 instances of accidents with snow weather, which accounts for less than 0.5% of the training set. However, the finding is consistent for fine weather with respect to rain.

5.1.3 Neurons at work

The artificial neural network developed for the task combines layers of different sizes with activation functions of type ReLU, softplus and sigmoid. It was trained across 500 epochs using a batch size of 128 with the default Adam optimizer implemented in the Keras library [12]. The code that includes full details on implementation can be found in appendix D. The convergence of the error occurs very fast for both low and high speed roads, as Figure 17 shows (note that the horizontal axis uses log scale).

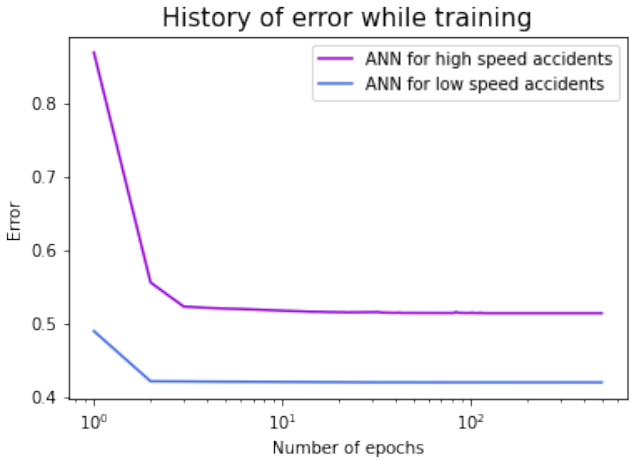
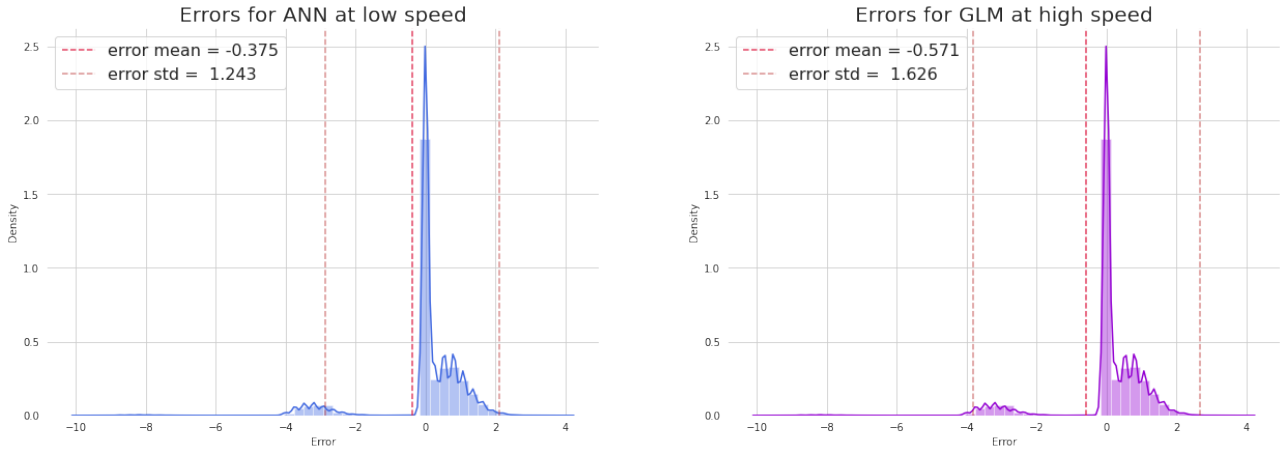


Figure 17: Error of ANN during training process.

The training error displayed above is higher in high speed accidents because the severity is those cases tends to be greater as already discussed. Applying the ANN to the prediction problem results in a MAE of 0.37545 for low speed events and 0.57182 for high speed events. Figure 18 shows how the errors are distributed in each case.



(a) Low speed roads

(b) High speed roads

Figure 18: Error distribution for predictions performed with ANN.

The legend reveals a major difference between the predictions of this ANN in comparison to the ones performed by the KNN and GLM models. In this case, the error mean is negative, which implies that the model is underestimating the severity consistently. However, this has en overall positive net effect when it comes to the MAE due to the fact that the neural network reduces the amount of positive errors committed by the other algorithms.

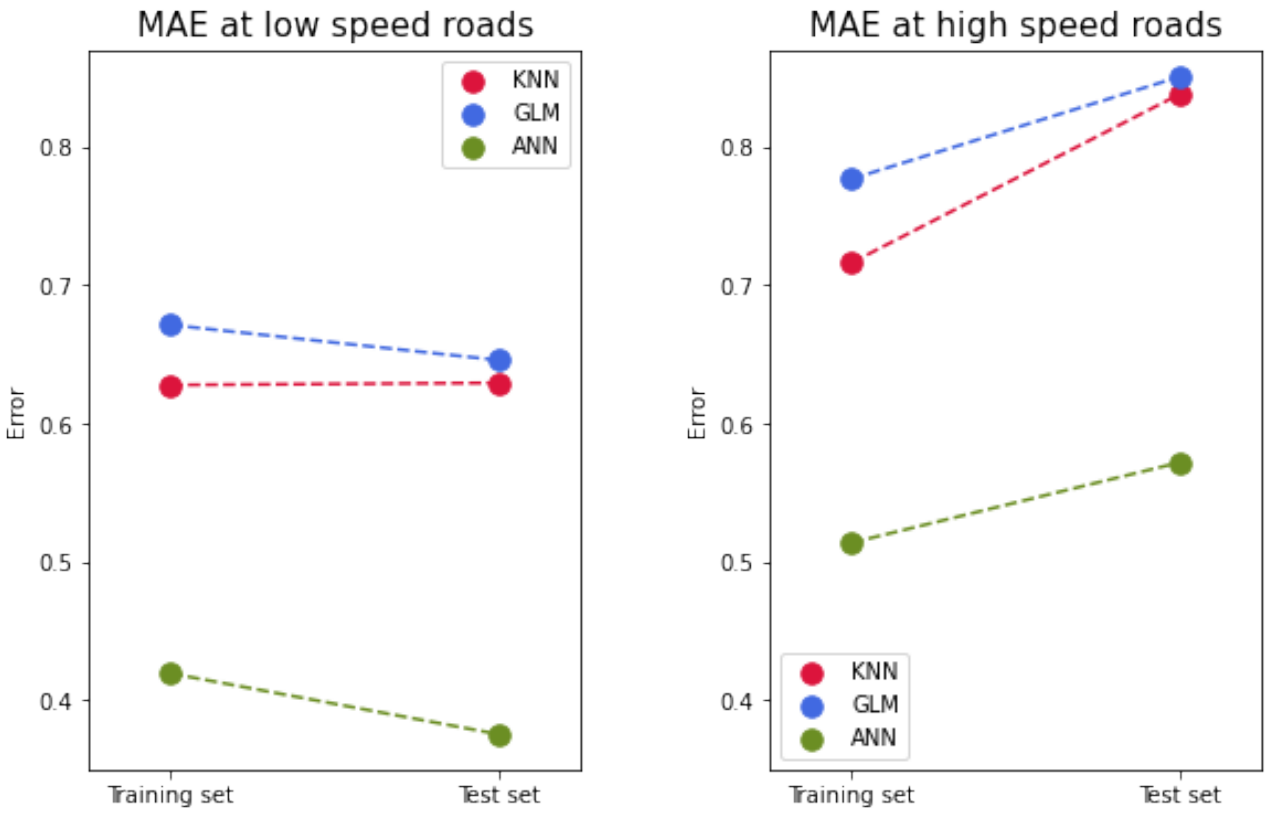
5.2 Summary of results

This section aims to give an overview of the performance achieved by all models considered. The mean absolute errors on both training and test sets are shown in table 3.

	KNN		GLM		Neural network	
	Training set	**Test set**	**Training set**	**Test set**	**Training set**	**Test set**
Low speed	0.62768	0.62920	0.67105	0.64560	0.41945	0.37545
High speed	0.71590	0.83787	0.77645	0.85037	0.51376	0.57182

Table 3: Errors when considering MAE

High speed roads have a greater error in all cases. As raw numbers make comparisons among models more difficult, results from the table above have been represented visually in Figure 19, where the vertical axis has been fixed to help see the differences between low and high speed roads.



(a) Low speed roads

(b) High speed roads

Figure 19: Prediction mean absolute errors for all models considered.

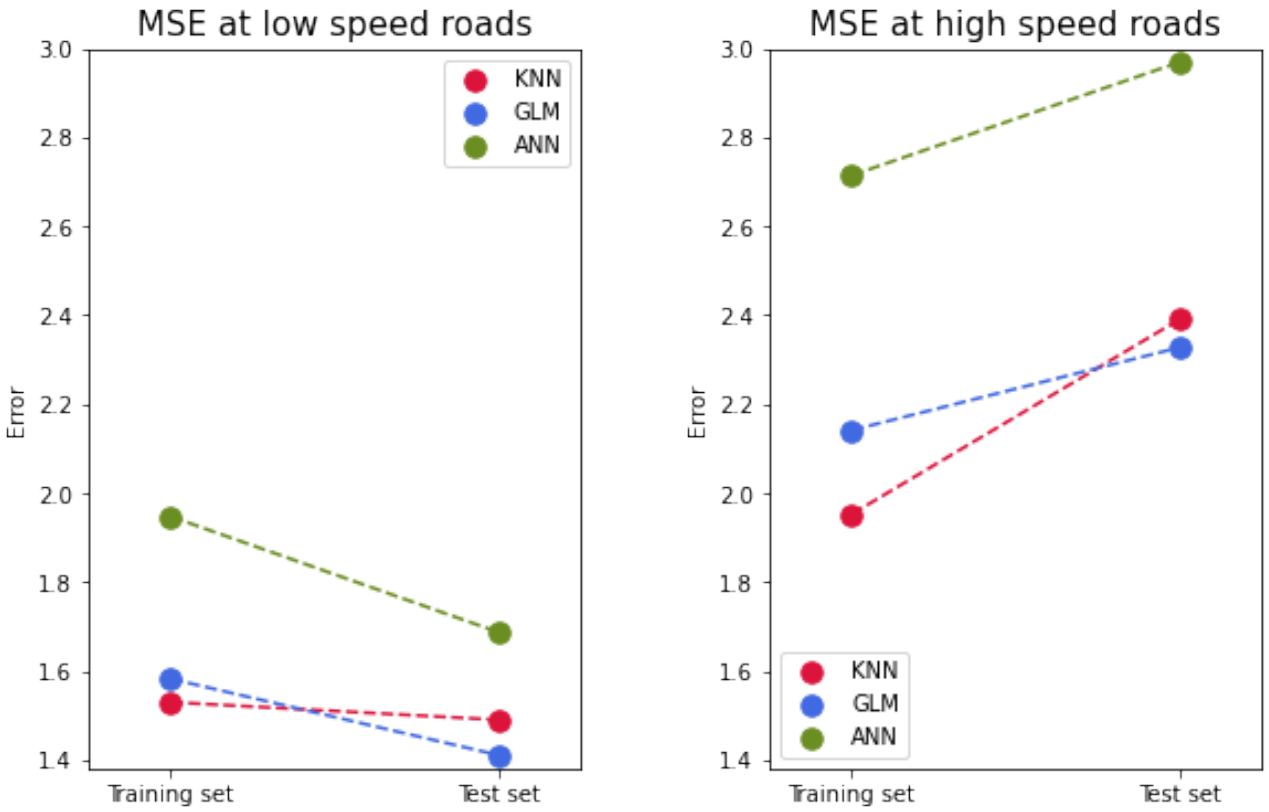
It is obvious from the graphs above that the neural network outperforms the KNN and GLM models. Its errors are lower at any speed and for training and test sets. Although KNN does slightly better than GLM, their performance is quite similar, especially in the test set, where the GLM nearly closes the gap between the two. Furthermore, an interesting disparity between low and high speed roads is how the training set performance compares to the test set. The slope of the dotted line joining training and test set performances is greater for high speed roads, which implies that the learning process is not ideal, as the error grows. The slope is even negative for some algorithms in low speed roads. Although it is difficult to tell with certainty, the reason for this might be in the numbers discussed in section 4.4, since the training set contained a larger share of instances in the case of low speed roads.

Let's now consider a different metric to evaluate performance. We will using mean squared error, which is not that far from mean absolute error as one can tell from the definitions provided in equations in section 3.4. This new metric produces the following errors.

	KNN		GLM		Neural network	
	Training set	**Test set**	**Training set**	**Test set**	**Training set**	**Test set**
Low speed	1.52931	1.48954	1.58178	1.40983	1.94687	1.68611
High speed	1.95230	2.39144	2.14118	2.32772	2.71442	2.96967

Table 4: Errors when considering MSE

As it was to be expected, errors are greater because MSE squares the difference between the prediction and the actual value. This means that every time an error is greater than one, which happens quite often given what can be seen on Figures 15, 16 and 18, its contribution will become bigger. However, the most remarkable change with respect to taking MAE as metric is the comparison of our models. Figure 20 shows this very clearly.



(a) Low speed roads

(b) High speed roads

Figure 20: Prediction mean squared errors for all models considered.

The ranking order of models has reversed. When considering MSE, the neural network's performance worsens dramatically and has the highest error of the three. Although KNN and GLM remain close, this time it is the GLM that gets the best result over the test set. The fact that GLM is the most reliable model when changing the metric indicates that it is able to catch deeper principles that drive accidents' severity. This idea will be explored in more detail in the next section.

5.3 A discussion around interpretability

The idea of interpretability was introduced in this paper in section 3.4. Having developed and applied three different models to the traffic accidents prediction problem, one should wonder if the concern about interpretability is relevant here. In other words, is this situation more similar to the warehouse logistics case or the life-or-death surgery example? We could agree that traffic accidents have a large effect on health and involve ethics, so they belong to the second group. In addition, a factor to bear in mind is that these predictions will have an impact on the way government handles traffic. They might lead to road closures or incentives to certain regions for new infrastructures, disfavouring others. Integrating insights obtained from algorithms into daily lives requires interpretability to increase social acceptance of decisions taken using machine learning models.

Among the three models considered in this dissertation, the Poisson-based GLM is the plain winner for interpretability. After all, KNN uses a mathematical partition of the future space that simply looks for the k-closest instances to a given point, so it is inherently local and there are no global weights or structures explicitly learned [36]. ANN is the result of multiple iterations of a numerical optimization algorithm over a big amount of neurons, so the role each assigned weight plays in the overall prediction is not easy to follow. Several methods such as ROAR (Remove and retrain) [27] have been proposed, but they are not straightforward, usually requiring a lot of computation time, and have mixed results. In contrast, we can obtain information from the parametric GLM model very easily.

Section 5.1.2 has been dedicated to explaining what relationships between variables and severity can be derived from the GLM. For the ones that are more clear, i.e. day of the week, the effect has been quantified. Even the least intuitive takeaways, such as the fact that fine weather seems to be more dangerous than rain, have literature backup. For example, Karlaftis and Yannis [54] used 21 years of daily rainfall data for Athens, Greece, and found that high amount of precipitation may reduce the number of accidents. The reason for this kind of unexpected pattern that some other authors have proposed is that drivers change their habits to adapt to the weather conditions and can therefore compensate outside negative factors [30]. Nevertheless, there is also evidence suggesting that rain causes more accidents both involving vehicles ([10],[11]) and pedestrians ([21]).

All in all, the GLM provides with the most information about the prediction problem. Not only does it have a greater interpretability power, but also gives the best results when the performance metric is changed, as section 5.2 has shown. This is vital for open ended problems such as the one this dissertation has tackled. It allows for decisions to be taken by humans using machine learning as a tool, and not the other way around, which fits perfectly with the core values of the White Paper on Artificial Intelligence [13]. Public policies need transparency.

6 Conclusions

The aim of this dissertation was to use three different models (KNN, GLM and ANN) to predict traffic accidents which took place in Great Britain in the 2016-2019 period and compare their results for the task. The choice of models was not accidental, since one of them is parametric (GLM) and the other two are not, with KNN functioning as a simple baseline and ANN as a more complex solution. It is important to point out that the results obtained from them include not only the performance of the algorithms when looking at the errors on the predictions but also the degree of interpretability they can offer. As it has been previously discussed, this is particularly relevant given the nature of the problem.

After some necessary remarks regarding a revision of literature review and mathematical descriptions of the models to be used, the largest volume of work was devoted to studying the problem of accidents prediction. This required a careful scrutiny of the data available to best adapt the tools to the information at hand, which was carried out primarily by converting data types to categorical and applying one-hot encoding for homogeneity reasons. Furthermore, the analysis of data provided the first insights about how vehicle collisions occurred. For example, clear patterns in accident rates were observed between peak and off peak traffic hours, which is consistent with evidence found in the literature. Also, there was a division between two distinct types of accidents depending on the speed of the road where the event had taken place. Consequently, the problem was bisected to better take into account the traffic dynamics.

When it comes to measuring the performance of the models that were applied to solve the problem, two indicators were considered: mean absolute error and mean squared error. The former was the main metric throughout the thesis, since it was used when training and adjusting the algorithms; the latter was proposed to check for consistency. MAE showed similar results for low and high speed roads, with ANN greatly outperforming both KNN and GLM, which had very similar results as KNN edged out GLM by a small margin. One could argue that this outcome was to be expected given the usual superiority of neural networks for prediction exercises. However, when MSE was measured the same ANN had worse results than the other algorithms and GLM was even able to beat KNN's performance in the test set for both low and high speed roads. This occurs due to the underestimation of severity that the neural network does, which leads to substantial errors associated with giving some of the serious or fatal accidents a low severity score. When those errors are squared, they have a significant impact on the overall performance.

The most important takeaway from the performance results is that ANN is the best model for the well-defined task of accidents prediction, but at the same time it is probably overfitted to it. Even a slight change in the metric used leads to a notable decline in effectiveness when comparing it to the other models. In contrast, GLM is the most consistent algorithm, since its predictions remain good even when metrics change. It is also worth mentioning that GLM is able to adapt to the test set better than any other algorithm. Indeed, in the case of low speed roads its performance on the test set is better than in the training set for both metrics. This is distinctively remarkable because the training set included only accidents taking place in the Manchester region, whereas the test set was the whole Great Britain. The fact that training with data exclusively taken within a restricted geographical area extends so seamlessly to the whole country opens the door for very exciting future developments. For instance, pilot projects of new road signs or innovative traffic control strategies could be tried in a relatively small region. Then, the data on the success or need for improvement of the projects could be extrapolated with confidence to the UK.

The other main domain of interest of this thesis is linked to interpretability, where using a GLM has many more incentives than the other approaches given that it is possible to interpret from the coefficients which variables have the biggest impact on the prediction. One of the strongest findings achieved was that, despite the fact that a relatively small fraction of accidents take place on weekends, they are related to greater severity than those happening on weekdays. Moreover, speed limit plays

a significantly larger role when it is high (50, 60 or 70 mph) than when it is low (20, 30 or 40 mph). Nevertheless, other features such as urban or rural areas, time of the day or the presence of sunglare do not show any significant effect on severity. Although this result might actually be true for the case of sunglare, which was an original hypothesis that has been tested unsuccessful, it goes against the intuition for time of the day, as peak times are reportedly more dangerous. Additionally, weather seems to be a more complicated factor to evaluate and there are contradictory examples in the literature. This dissertation aligns with those which postulate that rainy days actually decrease the severity of accidents. The reasons for this might be connected to the lower amount of traffic those days or to drivers adapting to the new circumstances, although no data on the matter was available to support the claim.

In fact, future developments could be done based on this work by augmenting the quality and quantity of the data available. Here only those circumstances surrounding an accident were considered, but no direct information on the vehicles involved was used. In the beginning of the project, NTIS datasets were considered to obtain statistics on traffic flow per road, but in the end they were discarded. Linking NTIS data to individual accidents would have led to added complications as the reference IDs used by NTIS changes depending on the date, and additionally, the majority of the accidents recorded did not occur in the same locations as where NTIS sensors were available. Other interesting information to include could be predictive maintenance data of each vehicle (age, engine capacity, measurements such as tyres air pressure...) in order to provide an individual risk assessment. Furthermore, the method employed to predict weather of non-accidents events, which was designed to gain granularity, derived results that should be taken cautiously given its simplicity. In that sense, access to better sources of historical weather would be helpful.

Lastly, and as a final addendum to the interpretability discussions, more research towards the application of traffic results to government action would be extremely interesting. If the most decisive causes for severe accidents can be identified, they could be avoided to a certain extent. A public authority with precise knowledge of the effect of traffic or day of the week on accidents may have significant incentives to reduce the number of vehicles that use a specific road on a specific day. Re-routing cars might increase travel time on a small scale, but the overall effect will be beneficial if consequently the number of accidents drops. A focus on interpretable machine learning is paramount to fully justify evidence-based decisions that can generate improvements for us all.

References

[1] Generalized linear models. advanced methods for data analysis (36-402/36-608). https://www.stat.cmu.edu/~ryantibs/advmethods/notes/glm.pdf, 2014. Consulted on 2022-07-21.

[2] Sopra steria - about us. https://www.soprasteria.com/about-us, 2022. Consulted on 2022-08-23.

[3] Sopra steria - our offices. https://www.soprasteria.com/about-us/our-offices, 2022. Consulted on 2022-08-23.

[4] A. Adadi and M. Berrada. Peeking inside the black-box: A survey on explainable artificial intelligence (xai). *IEEE Access*, 6:52138–52160, 2018.

[5] A. Agresti. *Foundations of Linear and Generalized Linear Models*. John Wiley Sons, Inc., 2015.

[6] A. Beck. *First-Order Methods in Optimization*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2017.

[7] C. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[8] Brake. Road crashes: a preventable epidemic around the world. https://www.brake.org.uk/get-involved/take-action/mybrake/knowledge-centre/global-road-safety, 2018.

[9] J. Broughton. Forecasting road accident casualties in great britain. *Accident Analysis Prevention*, 23(5):353–362, 1991. Special Issue: Theoretical models for traffic safety.

[10] C. Caliendo, M. Guida, and A. Parisi. A crash-prediction model for multilane roads. *Accident Analysis Prevention*, 39(4):657–670, 2007.

[11] L.-Y. Chang and W.-C. Chen. Data mining of tree-based models to analyze freeway accident frequency. *Journal of safety research*, 36(4):365–375, 2005.

[12] F. Chollet et al. Keras adam optimizer. https://keras.io/api/optimizers/adam/, 2015.

[13] E. Comission. White paper: On artificial intelligence - a european approach to excellence and trust. Technical report, European Union, 2020.

[14] N. S. . T. Council and U. S. D. of Transportation. Ensuring american leadership in automated vehicle technologies. Technical report, U.S. Government, 2020.

[15] N. Deretić, D. Stanimirović, M. A. Awadh, N. Vujanović, and A. Djukić. SARIMA Modelling Approach for Forecasting of Traffic Accidents. *Sustainability*, 14(8):1–18, April 2022.

[16] F. Doshi-Velez and B. Kim. Towards a rigorous science of interpretable machine learning, 2017.

[17] H. Fitrianti, P. Pasaribu, and P. Betaubun. Modeling factor as the cause of traffic accident losses using multiple linear regression approach and generalized linear models. *IOP Conference Series: Earth and Environmental Science*, 235:012030, 02 2019.

[18] J. Fox. *Applied regression analysis and generalized linear models*. SAGE Publications, Inc., 3rd edition, 2016.

[19] B. García de Soto, A. Bumbacher, M. Deublein, and B. T. Adey. Predicting road traffic accidents using artificial neural network models. *Infrastructure Asset Management*, 5(4):132–144, 2018.

[20] U. N. General Assembly. Improving global road safety. https://www.un.org/pga/74/wp-content/uploads/sites/99/2020/08/Draft-Resolution-Road-Safety.pdf, 2020. Consulted on 2022-07-23.

[21] D. J. Graham and S. Glaister. Spatial variation in road pedestrian casualties: the role of urban scale, density and land-use mix. *Urban Studies*, 40(8):1591–1607, 2003.

[22] D. Graupe. *Principles of Artificial Neural Networks*. World Scientific Publishing Co. Pte. Ltd., 2013.

[23] G. A. Gravvanis, A. I. Salamanis, and C. K. Filelis-Papadopoulos. *Machine Learning Paradigms - Advances in Data Analytics*, volume 149 of *Intelligent Systems Reference Library*. Springer, 2019.

[24] K. Gurney. *An introduction to neural networks*. UCL Press Ltd., 1997.

[25] F. G. Habtemichael and M. Cetin. Short-term traffic flow rate forecasting based on identifying similar traffic patterns. *Transportation Research Part C: Emerging Technologies*, 66:61–78, 2016. Advanced Network Traffic Management: From dynamic state estimation to traffic control.

[26] J. Heaton. *Introduction to the math of neural networks*. Heaton Research, Inc., 2012.

[27] S. Hooker, D. Erhan, P.-J. Kindermans, and B. Kim. A benchmark for interpretability methods in deep neural networks, 2018.

[28] T. Inc. Tesla vehicle safety report. https://www.tesla.com/es_ES/VehicleSafetyReport?redirect=no, 2022. Consulted on 2022-07-21.

[29] N. Japkowicz and M. Shah. *Evaluating Learning Algorithms: A Classification Perspective*. Cambridge University Press, 2011.

[30] A. J. Khattak, P. Kantor, and F. M. Council. Role of adverse weather in key crash types on limited-access: Roadways implications for advanced weather systems. *Transportation Research Record*, 1621(1):10–19, 1998.

[31] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization, 2014.

[32] K. Kumar, M. Parida, and V. Katiyar. Short term traffic flow prediction for a non urban highway using artificial neural network. *Procedia - Social and Behavioral Sciences*, 104:755–764, 2013. 2nd Conference of Transportation Research Group of India (2nd CTRG).

[33] P. Linardatos, V. Papastefanopoulos, and S. Kotsiantis. Explainable ai: A review of machine learning interpretability methods. *Entropy*, 23(1), 2021.

[34] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang. Traffic flow prediction with big data: A deep learning approach. *IEEE Transactions on Intelligent Transportation Systems*, 16(2):865–873, 2015.

[35] W. S. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5, 1943.

[36] C. Molnar. *Interpretable Machine Learning*. 2nd edition, 2022.

[37] D. C. Montgomery, E. A. Peck, and G. G. Vining. *Introduction to linear regression analysis*. John Wiley Sons, Inc., 5th edition, 2012.

[38] E. E. Oudemans. Predicting traffic accident injury severity in great britain using machine learning techniques. Master's thesis, Econometrics – Specialisation Big Data, 2016.

[39] Y. Qian, X. Zhang, G. Fei, Q. Sun, X. Li, L. Stallones, and H. Xiang. Forecasting deaths of road traffic injuries in china using an artificial neural network. *Traffic Injury Prevention*, 21(6):407–412, 2020. PMID: 32500738.

[40] M. A. Quddus. Time series count data models: An empirical application to traffic accidents. *Accident Analysis Prevention*, 40(5):1732–1741, 2008.

[41] M. Reid and S. Balazs. *Geometry and Topology*. Cambridge University Press, 2005.

[42] T. Räuker, A. Ho, S. Casper, and D. Hadfield-Menell. Toward transparent ai: A survey on interpreting the inner structures of deep neural networks, 2022.

[43] J. Salotti, S. Fenet, R. Billot, N.-E. El Faouzi, and C. Solnon. Comparison of traffic forecasting methods in urban and suburban context. pages 846–853, 11 2018.

[44] M. I. Sameen and B. Pradhan. Severity prediction of traffic accidents with recurrent neural networks. *Applied Sciences*, 7(6), 2017.

[45] P. Scott. Modelling time-series of british road accident data. *Accident Analysis Prevention*, 18(2):109–117, 1986. Special Issue Accident Modelling.

[46] S. Shalev-Shwartz and S. Ben-David. *Understanding Machine Learning. From theory to algorithms*. Cambridge University Press, 2014.

[47] B. Smith, B. Williams, and R. Oswald. Comparison of parametric and nonparametric models for traffic flow forecasting. *Transportation Research Part C: Emerging Technologies*, 10:303–321, 08 2002.

[48] Statista. Number of licensed cars in the united kingdom (uk) from 2000 and 2020. https://www.statista.com/statistics/299972/average-age-of-cars-on-the-road-in-the-united-kingdom/, 2022. Consulted on 2022-07-23.

[49] S. Steria. H1 2022 results - robust revenue growth and higher profitability. https://www.soprasteria.co.uk/docs/librariesprovider2/cartes-de-voeux/sopra-steria-revenue-for-1st-half-2022.pdf?sfvrsn=300aa1dc_1, 2022. Consulted on 2022-08-23.

[50] C. Sénquiz-Díaz. Transport infrastructure quality and logistics performance in exports. *ECONOMICS*, 9(1):107–124, 2021.

[51] A. Theofilatos and G. Yannis. A review of the effect of traffic and weather characteristics on road safety. *Accident Analysis Prevention*, 72:244–256, 2014.

[52] T. D. Wemegah, S. Zhu, and C. Atombo. Modeling the effect of days and road type on peak period travels using structural equation modeling and big data from radio frequency identification for private cars and taxis. *European Transport Research Review*, 10, 2018.

[53] S. N. Wood. *Generalized Additive Models. An Introduction with R*. CRC Press, 2nd edition, 2017.

[54] G. Yannis and M. G. Karlaftis. Weather effects on daily traffic accidents and fatalities: a time series count data approach. In *Proceedings of the 89th Annual Meeting of the Transportation Research Board*, volume 10, page 14, 2010.

[55] Z. Zheng and D. Su. Short-term traffic volume forecasting: A k-nearest neighbor approach enhanced by constrained linearly sewing principle component algorithm. *Transportation Research Part C: Emerging Technologies*, 43:143–157, 2014. Special Issue on Short-term Traffic Flow Forecasting.

[56] J. Z. Zhu, J. X. Cao, and Y. Zhu. Traffic volume forecasting based on radial basis function neural network with the consideration of traffic flows at the adjacent intersections. *Transportation Research Part C: Emerging Technologies*, 47:139–154, 2014.

Appendices

A Structure of accidents datasets

Field name	Range of values	Comment
accident_index	12-digit number	Unique for each accident
accident_year	1979-2021	In our case, from 2016 to 2019
accident_reference	8-digit number	Might be repeated across years
location_easting_osgr	6-digit number	Location using OSGR system
location_northing_osgr	6-digit number	Location using OSGR system
longitude	3 decimal places	- - -
latitude	3 decimal places	- - -
police_force	1-99	Code for police department in charge
accident_severity	1-3	Indicates fatal, serious or slight
number_of_vehicles	Natural number	Number of vehicles involved
number_of_casualties	Natural number	Number of casualties in the accident
date	DD/MM/YYYY	- - -
day_of_week	1-7	1 for Sunday, 2 for Monday,..., 7 for Saturday
time	HH:MM	- - -
local_authority_district	1-999	- - -
local_authority_ons_district	Letter+8digits	Heathrow airport has its own code (EHEATHROW)
local_authority_highway	Letter+8digits	Heathrow airport has its own code (EHEATHROW)
first_road_class	1-5	1 for motorway, 2 for A(M), 3 for A, 4 for B, 5 for C
first_road_number	0-999	0 is used if the road is unnumbered
road_type	1-12	Each number represents a type (roundabout, dual carriageway...)
speed_limit	20-70	Only multiples of 10
junction_detail	0-9	Type of junction (0 if further than 20 metres from one)
junction_control	0-9	Control in place (0 if further than 20 metres from junction)
second_road_class	0-6	Same code as first_road_class (0 if further than 20 metres from junction)
second_road_number	0-99	Same code as first_road_number (0 if further than 20 metres from junction)
pedestrian_crossing_human_control	0-2	Control in place (0 if further than 50 metres from crossing)
pedestrian_crossing_physical_facilities	0-2	Control in place (0 if further than 50 metres from crossing)
light_conditions	1-7	1 for daylight, other numbers for darkness with different artificial lightning possibilities
weather_conditions	1-7	Fine,rain or snow with or without wind and fog
road_surface_conditions	1-7	Dry or different obstacles
special_conditions_at_site	0-7	Extraordinary events (0 for none)
carriageway_hazards	0-7	Extraordinary obstacles on carriageway (0 for none)
urban_or_rural_area	1-2	1 for urban, 2 for rural
did_police_officer_attend_scene_of_accident	1-2	1 for yes, 2 for no
trunk_road_flag	1-2	1 for trunk, 2 for non-trunk

Table 5: Features available in accidents datasets.

B Weather data

	January	February	March	April	May	June	July	August	September	October	November	December
Aberdeen	11	10	10	9	8	9	9	10	10	12	12	11
Birmingham	9	8	8	9	9	8	9	9	8	8	9	9
Bournemouth	9	8	8	8	8	8	8	8	7	9	10	9
Brighton	11	9	7	7	7	6	7	7	8	11	12	11
Bristol	9	8	8	9	9	8	9	8	7	9	10	9
Cardiff	11	9	9	9	10	8	9	10	9	10	11	10
Colchester	10	8	8	8	8	9	9	8	8	8	9	9
Dover	9	8	7	8	8	7	8	8	8	9	9	9
Edinburgh	11	9	10	10	10	10	11	10	9	10	10	10
Exeter	9	8	8	9	9	8	9	9	8	9	9	9
Glasgow	13	11	11	11	11	11	12	13	11	12	12	11
Inverness	13	11	12	12	12	12	12	12	11	12	11	11
Leeds	10	9	9	9	9	10	10	10	8	9	10	10
Liverpool	12	11	11	10	10	10	13	13	11	12	14	13
London	8	7	8	9	8	7	8	8	7	8	8	8
Manchester	12	10	10	10	10	11	12	12	10	11	11	11
Middlesbrough	11	9	8	9	9	9	9	9	9	9	10	10
Moffat	12	11	11	11	11	11	13	13	11	12	12	12
Newcastle upon Tyne	10	9	8	9	9	9	9	10	8	9	10	9
Norwich	9	9	9	8	9	9	10	9	8	9	9	9
Nottingham	9	8	8	9	8	8	9	8	7	8	9	9
Penrith	12	11	11	11	11	11	13	13	10	11	12	12
Peterborough	8	7	7	9	8	8	9	8	7	8	8	8
Reading	8	7	8	9	8	7	9	8	7	8	8	8
Sheffield	10	9	9	10	10	10	10	10	8	10	10	10
Southampton	8	8	8	9	8	8	8	8	7	9	9	8
Swansea	13	11	10	10	10	10	11	12	11	12	13	13
Wrexham	10	9	10	10	10	9	10	10	8	10	10	10

Table 6: Average number of rain days by month for a selection of UK cities.

	January	February	March	April	May	June	July	August	September	October	November	December
Aberdeen	5.2	4.9	2.0	0.9	0.4	0.0	0.0	0.0	0.0	0.2	1.0	2.6
Birmingham	3.6	3.8	1.7	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.3	1.9
Bournemouth	0.8	1.7	0.3	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.3	0.8
Brighton	1.7	1.7	0.6	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.3	0.9
Bristol	1.0	2.5	0.8	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.7
Cardiff	0.8	1.0	0.3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.3	0.7
Colchester	2.9	3.7	0.8	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.5	1.5
Dover	1.9	2.2	0.5	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.5	1.2
Edinburgh	2.0	2.8	2.0	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.5	1.7
Exeter	1.3	1.3	0.3	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.3	0.7
Glasgow	3.8	3.9	2.2	0.3	0.0	0.0	0.0	0.0	0.0	0.0	0.3	2.5
Inverness	3.9	4.5	2.9	1.7	0.7	0.0	0.0	0.0	0.0	0.2	1.4	2.8
Leeds	2.9	3.9	1.5	0.3	0.0	0.0	0.0	0.0	0.0	0.0	0.1	1.9
Liverpool	6.0	5.0	4.0	2.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	4.0
London	2.2	2.5	1.2	0.3	0.0	0.0	0.0	0.0	0.0	0.0	0.3	1.2
Manchester	2.7	2.7	1.3	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.2	1.3
Middlesbrough	3.8	4.1	2.2	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.3	1.6
Moffat	4.5	4.9	2.6	0.6	0.1	0.0	0.0	0.0	0.0	0.0	1.3	2.7
Newcastle upon Tyne	3.4	3.8	2.1	0.7	0.2	0.0	0.0	0.0	0.0	0.0	0.5	1.7
Norwich	3.0	4.4	1.5	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.5	2.0
Nottingham	2.7	3.8	1.3	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.3	1.5
Penrith	6.8	5.7	4.4	2.2	1.2	0.3	0.0	0.0	0.1	0.7	4.0	5.8
Peterborough	2.8	2.9	0.8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.1	1.5
Reading	1.8	2.9	1.0	0.3	0.0	0.0	0.0	0.0	0.0	0.0	0.2	1.4
Sheffield	3.1	3.5	1.6	0.3	0.0	0.0	0.0	0.0	0.0	0.0	0.2	1.6
Southampton	1.5	1.2	0.3	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.3	1.1
Swansea	0.2	0.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Wrexham	3.4	2.8	1.9	0.3	0.0	0.0	0.0	0.0	0.0	0.0	0.4	1.4

Table 7: Average number of snow days by month for a selection of UK cities.

C GLM coefficients

Variable	Coef	Std error	95% confidence interval	
urban_or_rural_area	-0.0247	0.011	-0.046	-0.003
day_of_week_Sunday	-2.3266	0.108	-2.539	-2.114
day_of_week_Monday	-2.3862	0.109	-2.600	-2.172
day_of_week_Tuesday	-2.3851	0.109	-2.599	-2.171
day_of_week_Wednesday	-2.3784	0.109	-2.593	-2.164
day_of_week_Thursday	-2.3682	0.109	-2.582	-2.154
day_of_week_Friday	-2.3593	0.109	-2.573	-2.145
day_of_week_Saturday	-2.3252	0.109	-2.539	-2.111
road_class_Motorway	-0.9748	0.157	-1.283	-0.667
road_class_A(M)	-1.7630	0.186	-2.127	-1.399
road_class_A	4.7174	0.171	4.381	5.053
road_class_B	4.7545	0.172	4.418	5.091
road_class_C	4.6622	0.172	4.325	5.000
road_class_Unclassified	4.7021	0.171	4.366	5.038
road_type_Roundabout	4.1100	0.165	3.787	4.433
road_type_OneWayStreet	4.2031	0.166	3.878	4.528
road_type_DualCarriageway	4.3059	0.165	3.983	4.629
road_type_SingleCarriageway	4.3770	0.164	4.056	4.698
road_type_SlipRoad	4.0931	0.175	3.749	4.437
speed_limit_20.0	2.5403	0.155	2.237	2.844
speed_limit_30.0	2.5193	0.154	2.217	2.821
speed_limit_40.0	2.6289	0.154	2.326	2.931
weather_conditions_Fine	-1.0589	0.174	-1.400	-0.718
weather_conditions_Rain	-1.0933	0.174	-1.435	-0.752
weather_conditions_Snow	-1.4786	0.173	-1.818	-1.139
hour_Peak	-0.7557	0.147	-1.045	-0.467
hour_OffPeak	-0.7262	0.147	-1.015	-0.437
hour_Night	-0.5143	0.147	-0.803	-0.226
month_1	-1.6825	0.110	-1.899	-1.466
month_2	-1.6659	0.111	-1.883	-1.449
month_3	-1.6644	0.110	-1.881	-1.448
month_4	-1.6291	0.110	-1.846	-1.413
month_5	-1.6638	0.110	-1.880	-1.447
month_6	-1.6692	0.110	-1.886	-1.453
month_7	-1.6615	0.110	-1.878	-1.445
month_8	-1.6492	0.110	-1.866	-1.433
month_9	-1.6715	0.110	-1.888	-1.455
month_10	-1.6252	0.110	-1.841	-1.409
month_11	-1.6520	0.110	-1.868	-1.436
month_12	-1.6474	0.110	-1.864	-1.431
Sunglare	-0.0252	0.018	-0.060	0.009
const	-5.1555	0.773	-6.670	-3.641

Table 8: GLM coefficients fitted for low speed roads.

Variable	Coef	Std error	95% confidence interval	
urban_or_rural_area	-0.0115	0.022	-0.055	0.032
day_of_week_Sunday	0.7623	0.092	0.582	0.942
day_of_week_Monday	0.6363	0.093	0.455	0.818
day_of_week_Tuesday	0.6515	0.094	0.468	0.835
day_of_week_Wednesday	0.6264	0.094	0.442	0.810
day_of_week_Thursday	0.6431	0.094	0.459	0.827
day_of_week_Friday	0.6514	0.094	0.468	0.835
day_of_week_Saturday	0.7294	0.094	0.545	0.913
road_class_Motorway	1.3621	0.123	1.122	1.603
road_class_A(M)	0.7800	0.142	0.501	1.059
road_class_A	1.4868	0.122	1.247	1.727
road_class_B	1.4425	0.125	1.197	1.688
road_class_C	1.1384	0.127	0.890	1.387
road_class_Unclassified	1.3856	0.125	1.140	1.631
road_type_Roundabout	5.5538	0.181	5.198	5.909
road_type_OneWayStreet	2.4070	0.285	1.849	2.965
road_type_DualCarriageway	6.0176	0.181	5.664	6.372
road_type_SingleCarriageway	6.2715	0.180	5.919	6.624
road_type_SlipRoad	5.6433	0.185	5.281	6.005
speed_limit_50.0	4.0806	0.171	3.746	4.415
speed_limit_60.0	4.1304	0.171	3.795	4.466
speed_limit_70.0	4.0837	0.172	3.747	4.421
weather_conditions_Fine	2.0785	0.164	1.756	2.401
weather_conditions_Rain	1.9545	0.166	1.630	2.279
weather_conditions_Snow	-0.4883	0.157	-0.797	-0.180
hour_Peak	-0.1522	0.141	-0.429	0.124
hour_OffPeak	-0.0554	0.141	-0.332	0.222
hour_Night	0.0632	0.139	-0.209	0.335
month_1	-1.3752	0.107	-1.584	-1.166
month_2	-1.3587	0.107	-1.569	-1.148
month_3	-1.2504	0.107	-1.460	-1.041
month_4	-1.2695	0.107	-1.479	-1.060
month_5	-1.3322	0.107	-1.542	-1.123
month_6	-1.3098	0.107	-1.520	-1.100
month_7	-1.2602	0.107	-1.469	-1.051
month_8	-1.2830	0.107	-1.492	-1.074
month_9	-1.3438	0.107	-1.553	-1.134
month_10	-1.2788	0.107	-1.488	-1.070
month_11	-1.4076	0.107	-1.617	-1.198
month_12	-1.2488	0.107	-1.458	-1.039
Sunglare	-0.1448	0.038	-0.219	-0.071
const	-12.2520	0.725	-13.672	-10.832

Table 9: GLM coefficients fitted for high speed roads.

D Relevant code

Listing 1: Function for sunrise time calculation.

```
def SunriseTime(row):

  (latitude, longitude) = osgb.grid_to_ll(row['location_easting_osgr'],
  row['location_northing_osgr'])
  date_required=row['datetime'].strftime("%Y-%m-%d")

  tf = TimezoneFinder()
  time_zone = tf.timezone_at(lng=longitude, lat=latitude)

  location = Location()
  location.name = 'name'
  location.region = 'region'
  location.latitude = latitude
  location.longitude = longitude
  location.timezone = time_zone
  location.elevation = 0

  dt0=date.fromisoformat(date_required)
  dt=datetime.datetime.combine(dt0, datetime.time.min)
  dt = pytz.timezone(time_zone).localize(dt)
  sun = location.sun(dt)
  a1=sun['sunrise'].time().strftime("%H:%M:%S")

  return a1
```

Listing 2: Neural network design.

```
mymodel = Sequential()
mymodel.add(Dense(12, input_shape=(41,), activation='relu'))
mymodel.add(Dense(20, activation='relu'))
mymodel.add(Dense(16, activation='softplus'))
mymodel.add(Dense(8, activation='relu'))
mymodel.add(Dense(1, activation='sigmoid'))

mymodel.compile(loss='mean_absolute_error', optimizer='adam', metrics=['mean_absolute_error'])

ModelLow=mymodel.fit(X_low_train, Y_low_train, epochs=500, batch_size=128)
ModelHigh=mymodel.fit(X_high_train, Y_high_train, epochs=500, batch_size=128)
```