Multilayer Perceptrons for Structural Health Monitoring of welded bridge joints based on Temperature, Traffic and Strain Monitoring Outcomes

A DISSERTATION PRESENTED
BY
DAVID E. HAVERON
TO
THE DEPARTMENT OF COMPUTER SCIENCE AND MATHEMATICS

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE
IN THE SUBJECT OF
BIG DATA

UNIVERSITY OF STIRLING
SCOTLAND, UNITED KINGDOM

SEPTEMBER 2017

Thesis advisor: Dr. Kevin Swingler David E. Haveron

Multilayer Perceptrons for Structural Health Monitoring of welded bridge joints based on Temperature, Traffic and Strain Monitoring Outcomes

Abstract

Cities are becoming increasingly connected to the digital world in pursuit of becoming 'smart' and more efficient. In the field of civil engineering, technological advances have opened up new, alternative approaches not only to assess and monitor the deterioration of civil infrastructures but potentially predict structural failure. While the traditional approaches for structural assessment of civil infrastructure have been resource intensive and somewhat inefficient, new approaches aim to explore the potential of data-based approaches which could lead to better allocation of resources and potentially timely detection of abnormal structural behaviour.

Motivated by this, the research presented in this thesis aims to investigate the application of artificial neural networks, specifically the multilayer perceptron (MLP), to characterise the normal correlation pattern between monitored environmental conditions (daily-averaged pavement temperatures), operational loads (daily-aggregated heavy traffic counts) and a strain-based performance indicator, using monitoring outcomes from the Great Belt Bridge in Denmark.

The research herein aims to discuss and evaluate the technologies available to investigate the outlined problem, a brief mathematical introduction to the relevant machine learning algorithms, then follows a methodology to review the business problem, explore and prepare the data, model the data and evaluate the best performing models.

In summary, a random discrete hyperparameter search was completed to identify nine unique models, representing nine strain-gauges. After training, these models were evaluated against a set of performance criteria however, on average, the multilayer perceptron models were unable to reproduce (or improve upon) the performance of the multiple linear regression models developed by Farreras Alcover[1].

Contents

Listing of figures

I would like to sincerely thank and acknowledge the guidance provided by my dissertation supervisor Dr. Kevin Swingler. In addition, I would like to thank Dr. Isaac Farreras Alcover and COWI A/S for the support and the permission to use data from the Great Belt Bridge (Denmark). Furthermore I would like to thank my Grandmother (Marjory Thomas, age 96) for encouraging me to keep my standards high in pursuit of my dreams. My mother Lesley and father Derek for being pillars of strength throughout my education and finally my best friend and sister Michelle for supporting & encouraging me to pursue that which I love.

"People don't buy what you do; they buy why you do it. And what you do simply proves what you believe."

- Simon Sinek, Start with Why: How Great Leaders Inspire Everyone to Take Action

1 Introduction

1.1 Background and Context

A book published by Professor Klaus Schwab[7] in the World Economic Forum describes that the World is on the brink of a new technological revolution: The Fourth Industrial Revolution, or Industry 4.0. The First Industrial revolution (mechanical production, railroads and steam power) started in the 1700s and was inspired by the idea that steam could be used to power a variety of mechanical systems. In the 1800s, the Second Industrial Revolution brought about electrical power and the assembly line. Its arrival inspired a period of mass production through systems which could be electrified. The Third Industrial Revolution in the 1900s brought to the world automated production, electronics and computers - the combination of which dramatically increased the productivity and efficiency of numerous industries.

Now, it is believed we are entering a period of change in the world brought on by a wave of new technological advances: the Internet of Things (IoT), machine learning, cognitive computing and artificial intelligence. There are a host of new technologies which could drive this potential period of change - however some of the biggest changes have been seen through the exponential growth in processing and storage capacity, rapid advances in communication and data transfer speeds and a dramatic reduction in manufacturing costs of sensors. In light of these advances, the path is being

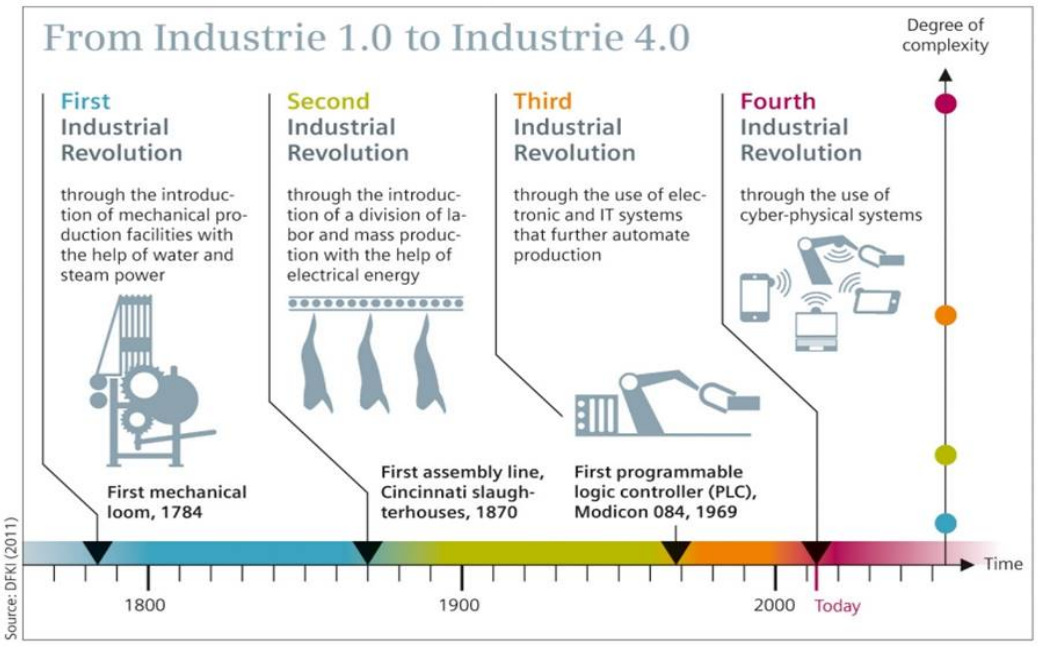

Figure 1.1.1: From Past to Present: Industry 4.0. Source: Online, DFKI 2011 [2]

paved for emerging technology in the fields of artificial intelligence, robotics, IoT, autonomous vehicles, 3-D printing, nanotechnology, biotechnology, materials science and quantum computing.

The relevance of this is we are now beginning to see a world increasingly connected and as a result, data is now being produced at unprecedented rates. Some estimates believe more than 90% of the data we have today, has been generated in the last few years alone[8] and others believe this might not change anytime soon. A research paper by a collection of authors at the University of Lancaster titled: “Are there limits to growth in data traffic?: On time use, data generation and speed” describe the growth of data in the last decade to be ”dramatic” and further argue that this will be an ongoing pattern[9].

This exponential growth in data production could enable us to gain deeper insights into the world we live in and provide us with long-term benefits in efficiency and productivity. With the considerable reduction in manufacturing costs of sensors the design of sensory and data collection systems into structural and mechanical assets is becoming increasingly feasible and even advantageous. Collecting, processing and modelling this data opens up opportunities for companies to characterise the relationships among extreme environmental or operating conditions, complex loading and structural or material responses in order to develop more efficient

products and services.

As an example, Rolls-Royce (a British luxury-car and aero engine manufacturing business) are using the data they collect from their products to anticipate and avoid failures, increase reliability and lower maintenance costs[10]. Similarly, John Deere - a manufacturer of agricultural, construction, forestry machinery and diesel engines, employ the same strategy to more effectively manage their fleet and reduce downtime of the products they produce through predictive maintenance models built from the data they collect[11].

This logic is now being explored in applications in civil structures to advance structural health monitoring (SHM) techniques. Charles Farrar, the director of The Engineering Institute at Los Alamos National Laboratory, and Keith Worden, the Head of the Dynamics Research Group in the Department of Mechanical Engineering at the University of Sheffield, co-published literature "Structural Health Monitoring: A Machine Learning Perspective" which details the extensive findings of the authors' surveys of technical literature and their conclusions of performing numerous analytical and experimental structural health monitoring studies[12].

Looking at specific applications, Nikolaos Dervilis published research carried out at the University of Sheffield and published his PhD thesis: "A machine learning approach to Structural Health Monitoring with a view towards wind turbines"[13]. Dervilis evaluated the use of Gaussian processes and neural network regression techniques for SHM of Wind Turbines and argues the research of pattern recognition algorithms for SHM purposes, is in its infancy when compared to application in civil and aerospace engineering.

Conventionally, monitoring of structural health in bridges involves routine and periodic visual inspections by certified structural/civil engineers or qualified personnel. Due to the nature of this methodology and inherent risk associated with physical inspections of structural components, inspectors may require health & safety training and permits to carry out inspections in a safe manner. Assuming thorough structural inspections are required for bridges which span over 1,000m, one could quickly deduce this approach can be resource-intensive particularly with larger, more complex civil infrastructure. In addition, these traditional approaches to structural health management have some significant drawbacks. For example, physical inspections are subjective as they are dependent on the judgement or experience of the inspector. Further, the inspections are often periodic in nature and cannot be fully relied upon to effectively capture the early signs of structural failure in real-time.

1.2 Problem Statement

Structural asset owners are often limited to the resources allocated to maintain structural health of assets. This resource allocation should be done in an efficient and optimal way, in order to maximise the life of the structural asset being maintained. The Great Belt Bridge in Denmark (*Figure 3*) has maintained and managed using systematic inspections of critical structural or load bearing components in order to determine its physical condition, maintenance needs, and potential hazards. This approach may require teams of inspectors to source appropriate permissions/permits to inspect the bridge, procurement of the equipment necessary to carry out inspection works and selection of a contractor or engineering team who are suitably qualified to carry out such works.

The Great Belt Bridge is subjected to a variety of loading including traffic and construction/maintenance loads, stress relaxation of cables, settlement of supports, wave impact and water flow[14]. The combination of dynamic loading and assorted weather conditions (temperature, wind and snow) add to the complexity of the system, making physical-based models less reliable. In light of this, data-based models are being explored as a solution to compliment physical-based model approaches (finite element analysis models, for example) in pursuit of more efficient structural health monitoring approaches.

1.3 Motivation

There is an argument to be made for the effective monitoring, maintenance and repair of civil structures. Structural infrastructures naturally fall victim to a complex stochastic deterioration exacerbated by complex dynamic material loading and environmental conditions.

And, as governments or structural asset managers are bound to budgetary constraint in maintenance of these assets, there is incentive to optimise this asset management process. Improved management programs of civil infrastructures could offer increased service life estimates and reduced maintenance costs or down time, among many other potential benefits. With limited funds devoted to structural repair, it becomes clear there is value in deploying accurate structural health assessment and service life prediction techniques, in order to allocate the resources in an optimal way.

Motivated by this, alternative approaches using data-based models are suggested for investigation in pursuit of more efficient structural health monitoring techniques.

1.4 Scope and Objectives

The scope of the research aims to document the data-mining approach taken to train and test a multilayer perceptron which serves to function as an envisaged real time structural performance monitoring system on the Great Belt Bridge (Denmark). That is, should the chosen model be given historic (or real time) temperature and traffic loading data (an observation for $\Delta t = 1$ day), can the models calculate estimates for the strain-based performance indicator with better performance than the multiple regression models developed by Farreras Alcover[6]? Improved performance of multilayer regression models, over multiple regression models, could further support the argument for short term and long term monitoring campaigns for collection of structural data in pursuit of more efficient structural health monitoring techniques. In addition, development of satisfactory models could imply an opportunity for the same data-based approach to be applied in structural health monitoring of assorted civil assets.

1.5 Achievements

Data from the Great Belt Bridge in Denmark, specifically the temperature, traffic count and strain-based performance indicator, was cleaned and used to train and evaluate 1000 candidate multilayer perceptron models for each of the nine strain gauges located under the 'slow' lane of the Great Belt Bridge. Appropriate performance metrics were defined and used to evaluate model performances after which, a random discrete hyperparameter search was executed to identify favourable model parameters for each of the nine models. Of the models considered, a model exhibiting satisfactory performance was selected for each of the strain gauges. The results of the selected models were presented and their performance compared to the nine multiple regression models developed by Farreras Alcover[6]. Finally, a discussion was presented reviewing the performance and suitability of the models for the regression task presented.

1.6 Overview of Dissertation

The *Introduction* chapter begins with a description of the growth of data and discusses use cases where data is explored in industry and academics to reduce inefficiencies in a variety of mechanical and civil applications.

The *State Of The Art* chapter details the relevant theory behind structural health motoring and includes a description of previous work carried out in development of data-based models to characterise the normal correlation between explanatory variables, $B_{\Delta t}$ & $T_{\Delta t}$, and the response variable, $D_{\Delta t}$. In addition, the chapter includes a high-level overview of the multiple linear regression theory.

The *Multilayer Perceptron for Structural Health Monitoring* chapter includes a brief introduction to the mathematical theory behind multilayer perceptron algorithms for regression tasks.

The *Cross Industry Standard Process for Data Mining* penultimate chapter, outlines the CRISP data mining approach used to train, evaluate and select multilayer perceptron models used to characterisation the normal correlation between the explanatory variables, $B_{\Delta t}$ & $T_{\Delta t}$, and the response variable, $D_{\Delta t}$.

Finally, the *Conclusions* chapter summarises the research findings, evaluates the the research achievements against the objectives set and suggestions are made for future work.

"When I was in college, I wanted to be involved in things that would change the world. Now I am."

- Elon Musk, Engineer and Entrepreneur

2 State-of-The-Art

2.1 Categories of Structural Health Monitoring Techniques

Farreras et al. explain structural health monitoring can be categorised according to either global or local approaches[15]. Global approaches have been explored in efforts to effectively identify significant or substantial damage of structural features. The global approach aims to detect and locate damages by comparing the dynamic properties (for example, natural frequencies and mode shapes) of a structure obtained at different times, with those properties corresponding to normal conditions[15]. In contrast, local approaches have been explored to better identify known deterioration/damage mechanisms occurring at pre-determined critical locations, for example fatigue or corrosion, and aim to quantify (rather than locate) the extent of the deterioration/damage in a structure[15]. In addition, structural health monitoring can further be categorised into either physical-based or data-based models[15]. Physical-based approaches typically rely on an initial physical representation of a model structure (for example, a finite element model) and model parameters need be updated using data provided by sensors[15]. Data-based models, in contrast, are statistical models which are not directly related to the underlying physical properties of a structure[15].

2.2 Regression models for predictive tasks

Following the categories available for structural health monitoring, regression models used for prediction of assorted structural failure could be considered as a local, data-based approach to structural health monitoring. A brief overview of the theory behind regression models is therefore necessary and further described in this section & the *Multilayer Perceptron for Structural Health Monitoring* section.

In the context of predictive models, a dataset used for training machine learning algorithms can be said to contain m examples (or observations) with n features. These dimensional attributes define the feature matrix, X, of size (m x n). Each row vector can be referenced as $x^{(i)}$ where *(i)* refers to the *ith* observation (with *n+1* dimension including $x_0 = 1$ in the matrix X). Each column vector can be referenced as x_j where j is the j^{th} feature or dimension. Following this, every element $x_j^{(i)}$ (row i, column j) represent the j^{th} feature of i^{th} observation. The feature matrix X and the target variable column vector (or response column) can be seen in *Figures 2.3.1* and *2.3.2*, respectively.

$$\mathrm{X} = \begin{bmatrix} x_1^{(1)} & x_2^{(1)} & \dots & x_j^{(1)} \\ x_1^{(1)} & . & . & . \\ x_1^{(2)} & . & . & . \\ \vdots & . & . & . \\ x_1^{(i)} & . & . & x_j^{(i)} \end{bmatrix}$$

Figure 2.2.1: A feature matrix *X* has *n* features with *m* observations (size *m x n*) where $x_j^{(i)}$ is the element value of the j^{th} feature from the i^{th} observation.

$$\mathrm{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ y^{(3)} \\ \vdots \\ y^{(i)} \end{bmatrix}$$

Figure 2.2.2: The target response column vector *y* with continuous values

Regression models are used to predict target variables which are on a continuous numeric scale, when provided with inputs. The model is initially trained to determine and estimate the coefficients for a line (uni-variate) or hyper-plane (multi-variate) which best fits the training data. Once adequately trained to a satisfactory performance, it can be used for prediction tasks. Thus, when provided input feature values, the model produces a real-value prediction.

Uni-variate regression (or simple linear regression) is used to uncover the relationship between a single feature and a target variable. However, if the dataset provided contains multiple input features, these may all contribute to the target variable value, and a more complex multi-variate 'multiple' (or polynomial) regression needs to be implemented to best characterise the relationships in the dataset. This is done through multiplying features with each other to make new features, for example given features x_1, x_2, new candidate features could include x_1x_2 or x_1x_1. The general form of multi-variate regression is represented in *equation (2.1)* where, $h_\theta(x)$ is the function used to calculate the predicted target value. Here, θ_i represents a given weight (also called parameters/coefficients) and x_i represents the feature or input value:

$$h_\theta(x) = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + ... \tag{2.1}$$

A more general form to represent many weights with many input values is introduced by setting x_0 to 1 in *equation (2.1)*. This approach sums the product of the weight and the input value over the entire dataset or and can be expressed in the vectorised version, *equation (2.2)*.

$$h_\theta(x) = \sum_{i=0}^{n} \theta_i x_i = \theta^T x \tag{2.2}$$

The weights are selected (or iteratively updated) such that the sum of the squared error (between actual observed and predicted values) on the training data, is minimised (using the ordinary least mean squares method LMS, or an alternative optimisation algorithm). The 'cost' function described by *equation (2.3)* can be used to quantify the model error and it used to update the weights θ_i in the direction such that the cost function value is minimised, and, the difference between a predicted value of $h_\theta(x_{(i)})$ and the true observed value y_i is reduced for all the observations in the training dataset.

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^i) - y_i)^2 \tag{2.3}$$

This is often done through an optimisation algorithm, with gradient decent being a common choice. The (batch) gradient decent update rule is represented by *equation (2.4)*, where the

weights θ_j are initially set to some random value, and simultaneously updated (for all θ_j). Here, θ_j represents a given weight and α is the learning rate (the step size taken in the direction of convergence). The learning rate is then multiplied by the partial derivative of the cost function with respect to the weight being updated (the partial derivative simply represents the rate of change with respect to that weight).

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta) \tag{2.4}$$

In addition, regularisation can be used with regression models, with the effect of inducing smoothness and enforcing a reduction in the complexity of the model (where necessary), to avoid over-fitting the model to the training data (these extended equations incorporating regularisation, have been omitted from this report).

Once this model has been trained to a satisfactory standard after identifying an optimal set of model parameters, the model is represented by a set of weights (or coefficients). These weights are then multiplied by their respective variable value and combined, to yield a final target variable value.

To illustrate a more definitive or real example, consider a predictive model which could be used to estimate a sale value for a property. Concretely, given inputs such as the property size (m^2), house age (no. of years), whether the property has a pool and undercover parking (binary values, 0 or 1), predict a real-value estimate of the properties value, generated from a model which has been appropriately trained on the property sales data from a specific area or post code. As with theoretical example of house value prediction, many other applications are being explored using predictive regression models.

2.3 Previous Work

There is increasing interest and incentive to better understand data-based approaches to structural health monitoring and a host of research has been completed exploring assorted themes of modern structural health monitoring - [16],[17],[18],[19],[20] among others. One of particular relevance, is research by Dr. Farreras Alcover who completed a PhD dissertation entitled "Data-based Models for Assessment and Life Prediction of Monitored Civil Infrastructure Assets", which detailed the development and investigation of data-based models (specifically polynomial regression models) as a novel approach for structural health monitoring. Included in Farreras Alcovers research was an assessment and probabilistic life prediction in the area of fatigue reliability of welded joints in orthotropic steel decks. Farreras Alcover concluded the

research noting an increase in pavement temperatures contributed to higher strain levels at monitored welded joints due to the temperature dependence of the pavement stiffness[1]. This research also showed loads induced by heavy vehicles (vehicles above 10m long and 2.8m high) were the main contributors S-N fatigue damages[1]. Farreras Alcovers used both a Weighted Least Squares (WLS) approach to determine the regression model parameters, in addition to a Bayesian regression approach to parameter estimation, and found the Bayesian regression approach to produce similar results to the Weighted Least Squares technique. More specifically, Farreras Alcover suggested monitoring of pavement temperatures and traffic conditions could lead to accurate estimates of the cumulative strain-based damage of the welded joints (via the developed regression models). Further, an argument was made for the potential benefit of short-term complete monitoring campaigns (pavement temperature, traffic counts and strain) for development of regression models, in addition to long-term monitoring campaigns targeting pavement temperature and traffic counts, which could potentially be used to obtain estimates of the strain-based performance indicator $D_{\Delta t}$ (proportional to S-N fatigue damage) at monitored welded joints.

The research carried out by Farreras Alcover, was conducted using the monitoring outcomes from the The Great Belt Bridge in Denmark (*Figure 2.2.1*). The bridge spans over 1,624 m in length and requires regular maintenance checks on critical structural components over the length of the bridge.

Figure 2.3.1: The Great Belt Bridge, Denmark. Source: Online, Storebaelt website [3]

"I've found in my experience that cleaning the data is 80% of the hard work."

- DJ Patil

3

Multilayer Perceptron for Structural Health Monitoring

3.1 The Multilayer Perceptron

The perceptron was designed to loosely mimic the learning functions of neuron in a biological brain (*Figure 3.1.1*). The term neural network is used when many perceptrons are combined in assorted arrangements, one such arrangement is the multilayer perceptron (MLP) which can be used for regression or classification tasks.

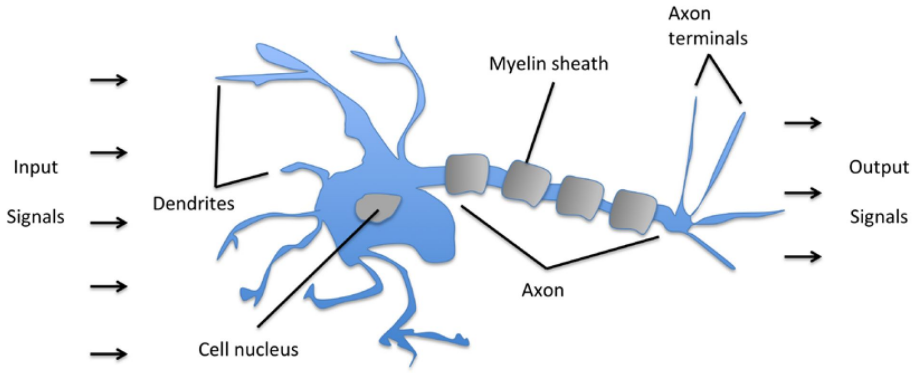

Figure 3.1.1: The basic structure of a neuron. Source: Sebastian Raschka - Python Machine Learning[4]

Perceptrons can be considered computational units that receive weighted input signals and produce an output signal using an activation function. The MLP network topology consists of a collection of perceptrons arranged into layers. *Figure 3.1.2* shows a single layer MLP with input nodes, one hidden layer and an output node.

On the left-hand side of *Figure 3.1.2*, each of the green input nodes receives a feature value for a given instance in the data and passes the value through the node as an output (no processing done in this first layer). These values are forward propagated through the network, multiplied by the appropriate weight and passed onto the next layer. For each of the blue nodes (specifically a_1, a_2 & a_3), the input values (product of the feature value & path weight) are added together and passed through an activation function to yield the activation value for that node (the node a_0 is often used as a bias unit and set to 1). This process is mathematically described in *equations 3.1*, *3.2* & *3.3*.

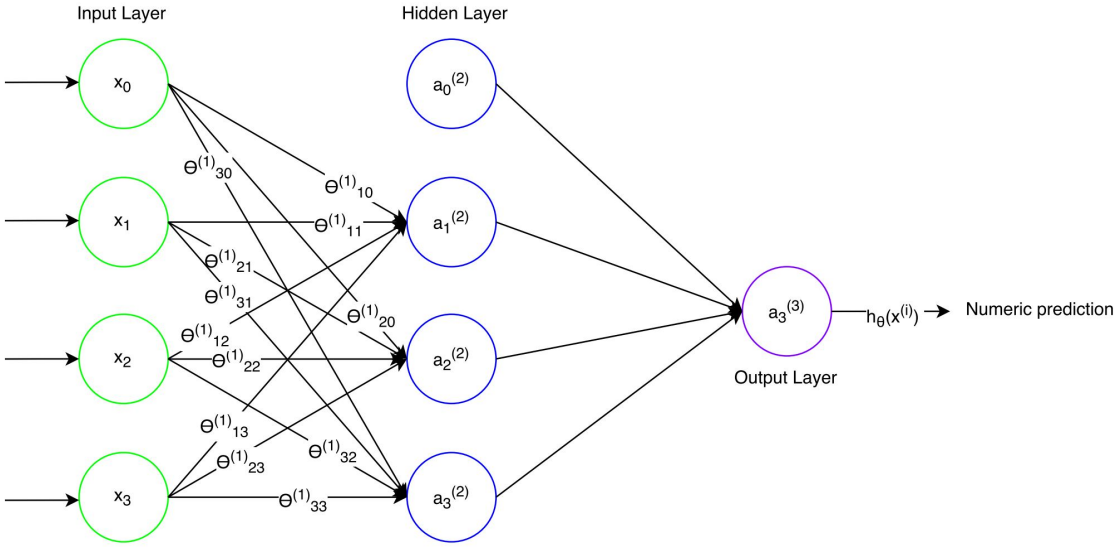

Figure 3.1.2: An example of simple MLP architecture for regression

$$a_1 = g(\Theta^{(1)}_{10} x_o + \Theta^{(1)}_{11} x_1 + \Theta^{(1)}_{12} x_2 + \Theta^{(1)}_{13} x_3) \tag{3.1}$$

$$a_2 = g(\Theta^{(1)}_{20} x_o + \Theta^{(1)}_{21} x_1 + \Theta^{(1)}_{22} x_2 + \Theta^{(1)}_{23} x_3) \tag{3.2}$$

$$a_3 = g(\Theta^{(1)}_{30} x_o + \Theta^{(1)}_{31} x_1 + \Theta^{(1)}_{32} x_2 + \Theta^{(1)}_{33} x_3) \tag{3.3}$$

As with the optimisation of the multivariate linear regression cost function, optimisation of the MLP cost function is also necessary to improve predictive performance. This is done with back error propagation or 'back-propagation' where the 'error' for a set of given weights is calculated and worked back through individual nodes in the network. This optimisation is achieved by updating the weights in the direction which minimises the individual node errors and hence the total error of the network. This is illustrated using *equations 3.4* where the partial derivative term is defined by *equation 3.5* (assuming no regularisation is applied). In these equations, l is any given layer being computed and j is the unit number:.

$$\Theta_j := \Theta_j - \alpha \frac{\partial}{\partial \Theta_j} J(\Theta) \tag{3.4}$$

$$\frac{\partial J\left(\Theta\right)}{\partial \Theta_{ij}^{l}} = a_j^l \delta_i^{l+1} \tag{3.5}$$

After the final node value is determined, the back-propagation is carried out using the total error of the network (δ_j^L). This error is defined as the difference between this predicted output (the final activation value, a_j^L) of the network and the true, observed value (y_j) from the dataset, where L is the last layer, l is any given layer being computed and j is the unit number:

$$\delta_j^L = a_j^L - y_j = h_\Theta(x^{(i)})_j - y_{observed} \tag{3.6}$$

Knowing the total error of the network δ_j^L the individual error associated with each activation node, δ_j^l (the error of node j in layer l) , can be calculated, and when passing it back through the activation function (that is, the derivative of the activation function), the error associated with that path weight, can be obtained:

$$\delta_j^l = \left(\Theta^l\right)^T \delta^{l+1} .* g'\left(z^l\right) \tag{3.7}$$

, where the values for z^l & $g'z^l$ can be determined by *equations 3.7* & *3.8*:

$$z^l = \Theta^{l-1} . a^{l-1} \tag{3.8}$$

$$g'\left(z^l\right) = a^l .* \left(1 - a^l\right) \tag{3.9}$$

Each path weight can then in turn be updated in the direction which minimises the total error of the network. The weights for a MLP can be updated for each data point (stochastic gradient decent), or once for a complete pass of the data set (batch gradient decent).

This outlines the basic theory of multilayer perceptrons for regression tasks, not including regularisation or the effects of other parameters. MLP's are known to be effective in learning and representing complex relationships between a feature matrix and an output variable. And as the output depends on the choice of activation function, the MLP output format can be chosen to correspond to the required regression task (linear activation functions are applied) or binary/multi-class classification task (in which cases sigmoid or similar functions are applied).

Although the MLP can perform well on both prediction and classification tasks, they have been known to require a high degree of optimisation and tuning for even the most basic network topology . As the complexity of the MLP grows, so does the number of parameters to be tuned and computational power required, to effectively cover the hyperparameter search space and hence choose an appropriate set of model parameters.

Despite these drawbacks, MLP's have a noteworthy ability to extract patterns and detect trends other algorithms perhaps cannot. A study by Gaudart, Giusiano and Huiart investigated the performance of MLPs when compared to linear regression models for epidemiological data and found comparable qualities in both the algorithms used[21]. Following this, it sparks an interest to investigate the performance of the MLP in achieving this regression task.

By 2020, the average cost per sensors is expect to drop to just $0.38

- Data: Goldman Sachs, BI Intelligence Estimates

4

Cross Industry Standard Process for Data Mining

CRISP-DM or Cross Industry Standard Process for Data Mining has been a leading industry standard and methodology to effectively manage data mining projects in a structured way[22]. The methodology is defined by a set of phases namely: Business understanding, Data Understanding, Data Preparation, Modeling, Evaluation and Deployment (*Figure 4.0.1*). The CRISP-DM methodology has been followed to address the research problem defined in the *Abstract* section at the beginning of the document.

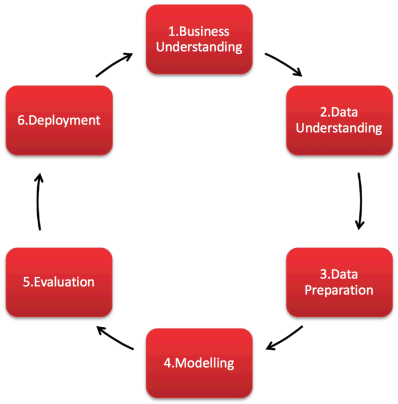

Figure 4.0.1: Cross Industry Standard Process for Data Mining. Source: Online, Smart Vision - Europe [5]

4.1 Business Understanding

4.1.1 The SHM system of the Great Belt Bridge

The Great Belt Bridge is located in Denmark and connects the Danish islands of Zealand and Funen. This suspension bridge was opened in 1998 and spans approximately 1624m in length with a maximum hanger width of 177m. The cross section (*Figure 4.1.1*) is formed by a closed steel box girder supporting an orthotropic deck suspended by longitudinal troughs and cross beams spaced every 4m. Data has been periodically collected on the bridges pavement temperatures, traffic loading conditions and strain observations at selected welded joints on the Orthotropic Steel Deck (OSD).

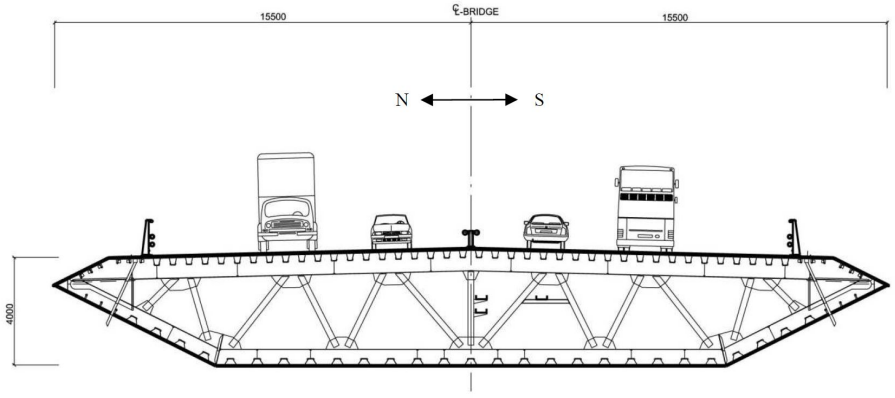

Figure 4.1.1: Cross-section of the Great Belt Bridge, Denmark. Source: PhD Thesis, Farreras Alcover[6]

1. **Pavement temperature monitoring system**

   The temperature monitoring sensors were installed at two different positions along the length of the bridge: one at the approximate 'centre' of the bridge span and one at the eastern end of the bridge. At each of these locations, there are further two separate temperature sensors installed at different cross sectional positions: S9901 & S9902 are installed in the northern lane and S9903, S9904 are installed in the southern lane (*Figure 4.1.2*). These sensors are embedded 10mm into the pavement and record pavement temperature readings at a time interval of $t = 5min$. The longitudinal location and traffic lane orientation of each of the four sensors is shown in *Table 4.1.1*. The daily-aggregated mean temperature is represented by $T_{\Delta t}$.

Temperature sensor	Longitudinal location	Traffic lane
S9901	Main Span	Slow-North
S9902	Main Span	Slow-South
S9903	Approach Span	Slow-North
S9904	Approach Span	Slow-South

Table 4.1.1: Position & orientation of traffic sensors. Source: PhD Thesis, Farreras Alcover[6]

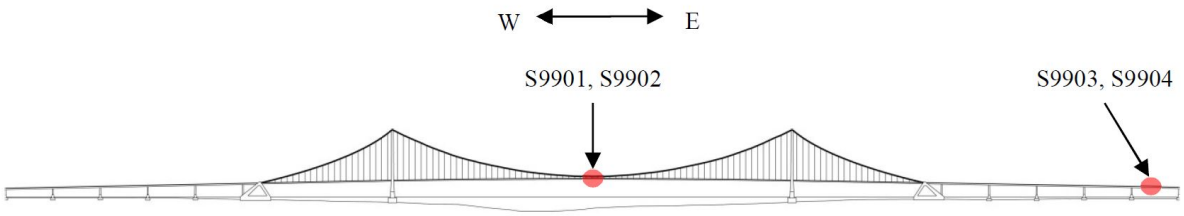

Figure 4.1.2: Longitudinal position of the temperature sensors SS9901-4. Source: PhD Thesis, Farreras Alcover[6]

2. **Traffic classification and monitoring**

The bridge has proved a critical infrastructure asset to the area with traffic flow reaching an approximate 10 million vehicles since 2006 (bi-directional traffic flow)[15]. The toll system makes hourly classifications of vehicles according to a set of dimensional criteria, classifying each vehicle into a category of vehicle class. *Table 4.1.2* illustrates the dimensional characteristics of the different vehicle categories. The daily-aggregated heavy traffic count is represented by $B_{\Delta t}$.

Vehicle class	Max. length [m]	Min. length [m]	Max. height [m]	Min. height [m]	Approx. vehicle type
1	3	0	No limit	0	Motorcycle
2	6	3	No limit	0	Car
3	20	6	2.8	0	Car with trailer
4	10	6	No limit	2.8	Van
5	20	10	No limit	2.8	Truck
6	No limit	20	No limit	No limit	Modular truck

Table 4.1.2: Characteristics of vehicle categories. Source: PhD Thesis, Farreras Alcover[6]

3. **A strain-based performance indicator**

The Great Belt Bridge has been instrumented with a configuration of 15 uni-axial strain sensors as per *Figure 4.1.3*. This configuration aims to effectively capture stresses experienced in the welds. Strain gauge (SG) numbers 1, 3, 4, 6, 7, 9, 10, 12, 13 and 15 record transverse nominal strains at the trough-to-deck weld. SG numbers 2, 5, 8, 11 and 14 are used to capture longitudinal nominal strains at trough splice welds.

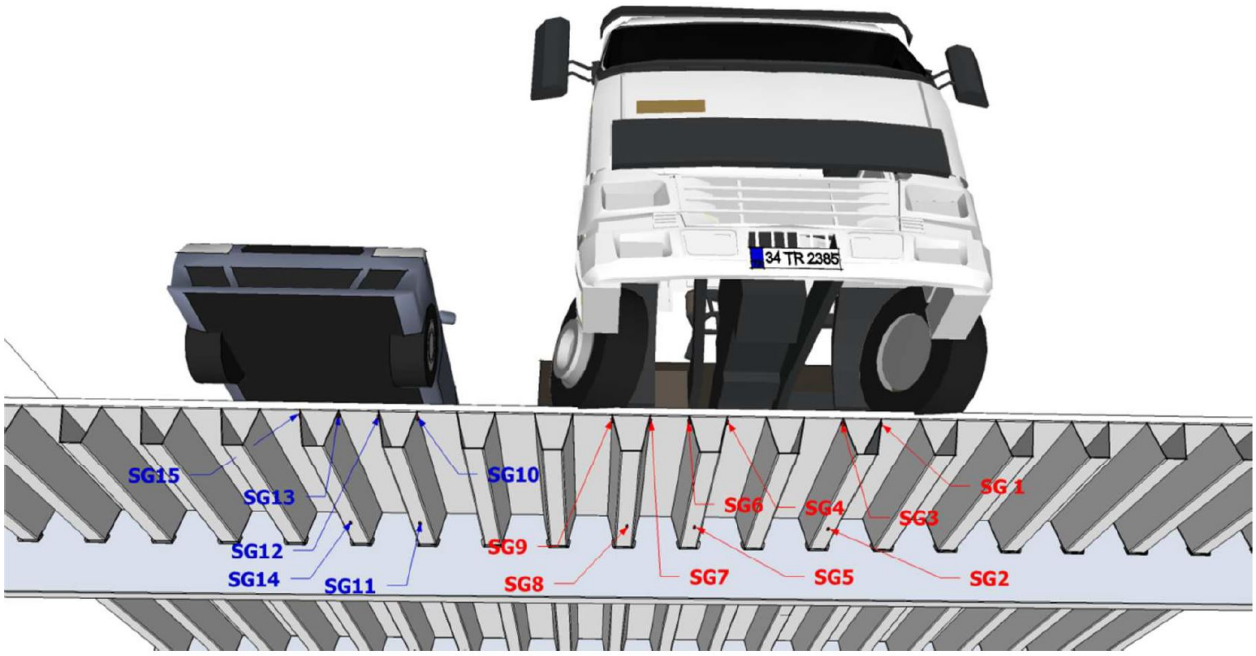

Figure 4.1.3: Configuration of strain sensors relative to traffic lanes. Source: PhD Thesis, Farreras Alcover[6]

Farreras Alcover [6] defined the strain-based performance indicator $D_{\Delta t}(t)$ to characterise the performance of a welded joint in terms of applied stresses. *Equation 4.1* illustrates how this indicator value is calculated, where $\Delta\sigma_i$ is the i^{th} stress range out of the total number of stress cycles (N_c) within the time interval $[t,t+\Delta t)$.

$$D_{\Delta t}(t) = \sum_{i=1}^{N_c} \Delta\sigma_i^3 \tag{4.1}$$

4.2 Data Understanding

The data files provided for this research are described, for reproducibility purposes:

1. Temperature data:

    - *File 1* - 'TEMPERATURE.xlsx' included 5 rows at the head of the file, describing meta-data about the file. The remaining part of the file contains a column with a timestamp against the measurement reading and a column with the reading measurement (temperature, °C). (Size - 786705R x 2C, including 5 rows of meta-data at the head of the file.)
    - *File 2* - 'Station 9902_2012.xlsx' contained the temperature data from the year 2012. The first column contains the temperature reading (°C) and the remaining columns define the year, month, day and minute the measurement was taken. (Size - 102521R x 6C.)

2. Traffic frequency count data:

    - *File 3* - 'Timetrafiksiden1998.xlsx' contained the traffic data from 1998 through to 2011. The first four columns indicate a timestamp. The remaining columns indicate assorted frequency counts for classes 1 - 10 for both Easterly and Westerly directions. The final columns indicate aggregated counts. (Size - 116698R x 27C, including header with variable names.)
    - *File 4* - 'Trafik_2012.xlsx' contained the traffic data from 2012. The first four columns indicate a timestamp. The remaining columns indicate assorted frequency counts for classes 1 - 10 for both Easterly and Westerly directions. The final columns indicate aggregated counts. (Size - 8783R x 27C, including header with variable names.)

3. Strain-based performance indicator data:

    - *File 5* - 'DD2011.mat' was provided as a Matlab data file, however the '.xlsx' format was available upon request. The 'DD2011.xlsx' file consisted of an unlabelled matrix of data - it was confirmed each observation (row of data) represents a strain gauge from SG1 - SG15, and each column represents a day of the year. Each data point represents a calculated value for the strain-based performance indicator, derived from aggregated stress values (*Equation 4.1*). (Size - 15R x 365C.)

- *File 6* - 'DD2012.mat' and 'DD2012.xlsx' represent the strain performance indicator data for the year 2012. **It was observed the year 2012 was a leap year and it was further confirmed an additional column was to be added at the end of the matrix to reflect strain gauge readings for 366 days of the year - this has been documented in the Data Preparation section.* (Size - 15R x 365C.)

4.3 Data Preparation

4.3.1 Big Data Technologies: Python Scripting Language, Jupyter Notebook, Pandas, Numpy & H_2O libraries

Data-based models yield best performance when the data they are trained on, is adequately prepared. However, as data seldom comes in this format, varying degrees of preprocessing is necessary to facilitate development of a model with satisfactory performance. This is done using a high-level programming language - the typical choices including Python, C, Scala, R or Java (among others). Once adequate preparation has been completed, the machine learning and optimisation algorithms need to be written (or imported from a pre-written 'library') and executed in an appropriate environment to support the computational requirements.

The work carried out in this project has been achieved using a selection of software technologies. For choice of a high-level programming language with which to clean the data and train/test the model performance, Python, R and Java were considered as candidates. Python and R offer similar functionality and are well supported by large programming communities, through which, many advanced functional libraries have been built. Python is considered a more general-purpose coding environment, where R is typically used when more of a pure statistical approach is required. Python too has been known to have a syntax many find easier to interpret than alternative more 'verbose' high-level languages such as Java or Scala. The growing interest in Python has resulted in a variety of available resources containing previous work and projects (Github, for example). This ecosystem has also given rise to a vast support community (Stackoverflow - a support repository, for example) for enthusiasts or academics who experience technical problems in achieving or executing tasks in Python or its supported libraries. However, depending on the specific requirements or objectives, the execution speed and computational efficiency may be considerations in choice of programming language. Java is known to outperform Python and R with respect to these, however considering the benefits of, and personal familiarity with, Python (over R or Java) for this specific application, a decision was made to implement Python as the high-level programming language to carry out this research.

The code has been written in Python 3.6 - Python and many other open source software projects have become an increasingly attractive choice for developers and scientists as the open source community collectively contribute to the development and maintenance of many new libraries which provide advanced functionality within the programming environment. Examples of these libraries include the Pandas and Numpy libraries. Pandas provides functionality for efficient handling and manipulating of 'Pandas' dataframes (pandas v0.20.1 was used). Numpy has become a core tool in the scientific computing community and provides high-performance multidimensional array objects, and optimised tools for processing these arrays (numpy v1.12.1 used).

Furthermore the H_2O library was considered as a candidate toolkit for execution of this task (H_2O version 3.12.0.1 used). This platform is comparable to the popular sklearn (Scikit-learn) library which provides the algorithms required develop and run many machine learning tasks in Python. H_2O is written in Java and is an open source (Apache 2.0) software for big data analysis and allows data scientist to fit thousands of potential models in search of discovering patterns in data. The software provides libraries of advanced machine learning algorithms, powerful data compression and the capacity to execute scalable in-memory processing for big data on one or many nodes. In addition, some of the greatest challenges of developing models which not only perform well on cross validation data (training/validation) but models which are able to generalise well to unseen data. For neural networks, this process of determining the optimal set of model parameters is called hyper-parameters tuning. Neural networks have many model hyper-parameters to define and evaluate, and this makes the discovery of an appropriate model to serve a particular function, difficult. The application of optimisation and heuristics to these algorithms promotes more efficient model training and data processing. Among other added benefits of using H_2O, include a graphical user interface (GUI) called Flow. H_2O Flow could be compared to the data mining software and data mining interface Weka, however has additional benefits worth mentioning.

After satisfactory training, testing and selection of a particular model (either using the H_2O Flow GUI, or explicitly developing the model in Python or another supported language), H_2O contains additional libraries to easily export the defined model (with its specific hyper-parameters) as a Plain Old Java Object (POJO). This may reduce the development resources required during the handover of the value created by the data scientist, to the developer or data engineer responsible for deploying a model commercially (this is often achieved using Java). This allows the data engineer to focus on effective deployment and implementation challenges, without spending additional time interpreting and explicitly defining the model as a Java Object.

For purposes of this task, however, it was decided not to implement the research in Weka or Flow, as the advantages of explicitly developing this code in Python using H_2O allows the user to carry out the pipeline of tasks from the data preparation, modelling, evaluation and deployment, in a single environment. For these reasons, H_2O was chosen over scikit-learn to achieve the model development and selection.

4.3.2 Preprocessing & model design considerations

Considerations raised during pre-processing and modeling:

1. Number of inputs to be used when training the multilayer perceptron, consider as inputs:

   - Temperature, vehicle classes 1, 2, 3, 4, 5, 6 East & West (13 total), or,
   - Temperature, vehicle classes 5, 6 East & West (5 total)

   The *Results* subsection has included the model performances using both sets of the aforementioned model inputs to determine if one approach produces superior performance models. Of relevance though, Farreras Alcover used a training dataset corresponding to SG2 (3^{rd} degree polynomial) to estimate model parameters using least squares regression for which the following cases were considered (among others) *i*) $B_{\Delta t}$ defined by all classes, and, *ii*) $B_{\Delta t}$ defined by classes 5 & 6. Farreras Alcover found the models developed using classes 5 and 6 to define $B_{\Delta t}$ clearly outperformed the rest of the models (and cases) both in terms of MSE and MAPE[6].

2. Choice of training/validation & test split, which of the following two approaches would be more appropriate:

   - Replicate the approach used by Farreras Alcover using 2012 data as the training/validation & 2011 as the test set, or,
   - Combine 2011 & 2012, reshuffle and split the data by some common training/validation and test split

   This is a defining step, as at this stage of model training, it is considered critical to ensure a complete range of temperature and traffic conditions are used, to accurately capture the input-output dependence[6]. It was decided to use the data from 2012 as the training/validation set and the data from 2011 as the test set as to best replicate the model conditions simulated by Farreras Alcover[6].

3. Choice for the number of folds to be used during cross validation:

    - 5 -fold is default for H_2O and said to be better for large datasets, or,
    - 10-fold is considered a reasonable choice for most applications, but for smaller training/validation sets it could be beneficial to increase the number of folds[4].

4. Choice of granularity of the data used to train the models:

    - Time discretization $\Delta 12$ hours (or less), or,
    - Time discretization $\Delta 24$ hours

    Unfortunately, the strain-based performance indicator data provided has been aggregated into a daily time discretization. Manipulating these could introduce bias results in the models. In addition, Farreras Alcover noted that a smaller time discretization of hourly models resulted in lower squared errors than when a daily time discretization, however a close set of results when assessing the cumulated predictive performance, Ψ[6].

5. The effect of model performance considering the discrepancy in dataset sizes:

    - Farreras Alcover carried out research using a training/validation (2012) and test set (2011) size of 221 & 77, respectively.
    - Post processing, the training/validation (2012) and test set (2011) size of 224 & 71, respectively. However, it was decided to combine the 2011/2012 datasets and recombine to ensure a representative dependance was captured.

6. The H_2O random grid search returns a list of the top models ranked according to MSE (or another defined performance metric). From the collection of models, the top performance model is considered in the discussions herein. As such, the 'best' performance model is defined as one with low MSE score, where in contrast Farreras Alcover considered the 'best' performance model the model which the lowest AICc score and hence lowest complexity. The consideration to be made here is whether the model performances can be compared - this is discussed in the *Conclusions* section.

7. The random grid search using H_2O, returns a collection of models ranked according to the cross-validation mean square error (MSE). A design decision had to be made to select either the top performance model from the grid search, or a lower ranked model - if a model reflects high performance on the training/validation data, and low performance on the test data, the model has overfit the data and can be described as more descriptive versus predictive.

As such, the selection of a lower performance model could yield a good compromise to effectively select a model which performs well both on the training/validation & test data.

4.3.3 Data Cleaning & Preprocessing

The data available for training/validation and testing of the MLP, has been processed accordingly:

Temperature data (consisting of two data files, 1998-2011 & 2012):

1. Each data set is loaded into a Pandas dataframe.
2. Column names are assigned to each column in the datasets.
3. A timestamp is created from the existing timestamp format in each Pandas dataframe.
4. This new timestamp column is then assigned as a Pandas 'DatetimeIndex'.
5. The datasets are filtered for observations occurring in the years 2011 & 2012.
6. The temperatures are grouped by day and an mean temperature for each day is calculated.

Figure 4.3.1 illustrates an example of the 2012 cleaned temperature data, with the index representing a given day of the year and a plot of the 2012 Temperature data, where the data has been distributed into frequency bins (176 bins used) and represented using a histogram:

Timestamp	Average Temperature (°C)
2012-01-01	5.967727
2012-01-02	6.130697
2012-01-03	3.617217
2012-01-04	4.386617
2012-01-05	4.275696

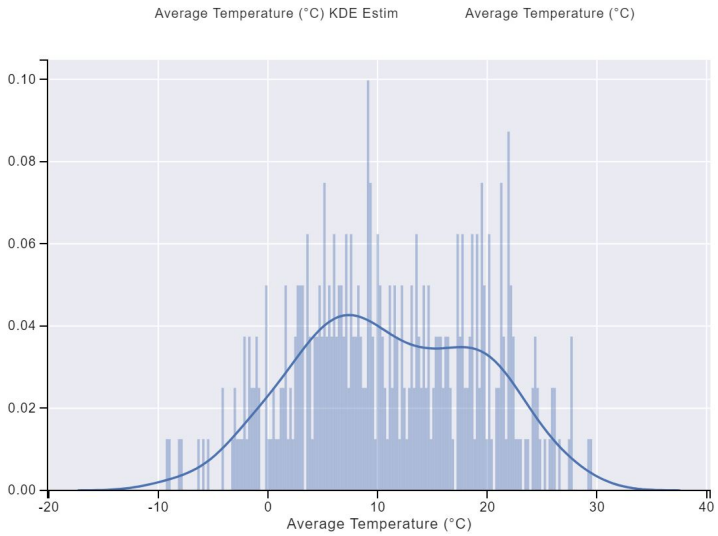

Figure 4.3.1: Example cleaned temperature data and temperature distribution plot $T_{\Delta t=24hours}$

Traffic frequency data (consisting of two data files, 1998-2011 & 2012):

1. Each data set is loaded into a Pandas dataframe.
2. Column names are assigned to each column in the datasets.
3. A timestamp is created from the existing timestamp format in each Pandas dataframe.
4. This new timestamp column is then assigned as a Pandas 'DatetimeIndex.
5. The datasets are filtered from observations occurring in the year 2011 & 2012.
6. The traffic counts are grouped by day and an mean traffic count for each day is calculated.

After the traffic count data was cleaned, classes 7, 8, 9 &10 (both East/West) were removed from the dataframe as this data was not included in the study by Farreras Alcover[1]. *Figure 4.3.2* illustrates an example of the 2012 cleaned traffic data, with the index representing a given day of the year:

Timestamp	Class 1 - East	Class 2 - East	Class 3 - East	Class 4 - East	Class 5 - East	Class 6 - East	Class 1 - West	Class 2 - West	Class 3 - West	Class 4 - West	Class 5 - West	Class 6 - West
2012-01-01	25	12150	135	26	382	23	16	8901	86	27	227	9
2012-01-02	49	8784	304	131	1479	71	37	7206	158	109	1392	63
2012-01-03	43	7843	210	156	1608	87	44	7301	176	157	1664	84
2012-01-04	35	8440	202	141	1666	78	39	8543	201	151	1675	86
2012-01-05	39	8874	178	177	1502	92	41	10043	283	195	1797	91

Figure 4.3.2: Traffic frequency data (2012) $B_{\Delta t=24hours}$

An example of the frequency distribution for traffic Class 1 - East, has been plotted in *Figure 4.3.3*:

Strain-based performance indicator data (consisting of two data files, 2011 & 2012):

1. Each data set is loaded into a Pandas dataframe.
2. The matrices are transposed.
3. Column names are assigned to each column in the datasets.
4. A timestamp vector is created for each year (lengths 365 for year 2011 and 366 for year 2012). ** a row of NaN was appended to the data to reflect a 366 observations for the year 2012.*

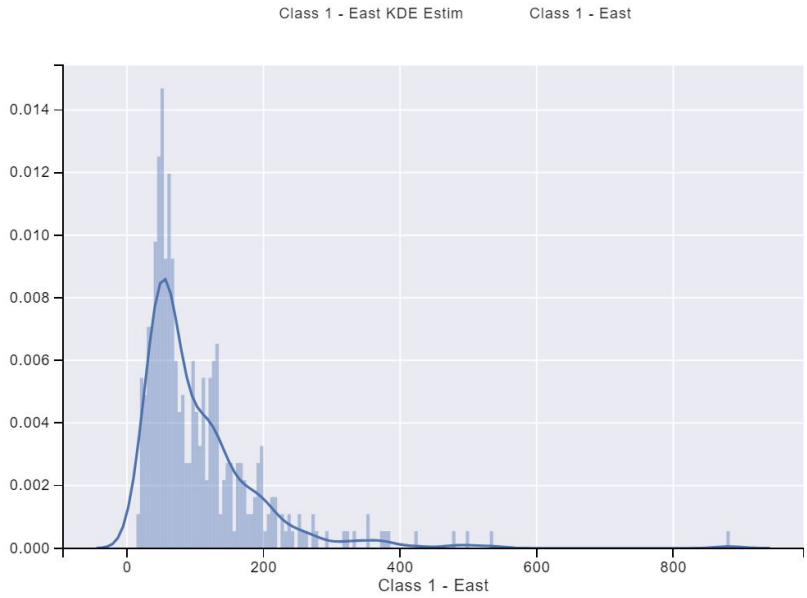

Figure 4.3.3: Traffic frequency distribution for vehicle Class 1 - East ($B_{\Delta t}$)

5. The timestamp vectors are assigned as a Pandas 'DatetimeIndex'.

Figure 4.3.4 illustrates an example of the 2012 cleaned strain-based performance indicator ($D_{\Delta t}$) data, with the index representing a given day of the year:

	SG1	SG2	SG3	SG4	SG5	SG6	SG7	SG8	SG9
Timestamp									
2012-01-01	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2012-01-02	723383.750000	5.076596e+06	5.628534e+05	180456.187500	2.278566e+06	511672.179688	675989.039062	6.698126e+06	640238.085938
2012-01-03	775992.117188	5.524494e+06	1.200359e+06	159717.062500	2.110696e+06	395373.476562	498252.835938	5.920283e+06	833872.109375
2012-01-04	752713.867188	5.747740e+06	8.342862e+05	161735.640625	2.385279e+06	401435.476562	554914.460938	5.950119e+06	602239.078125
2012-01-05	721434.656250	4.890578e+06	4.288384e+05	125526.585938	2.229993e+06	454186.750000	725954.531250	6.009062e+06	590401.039062

Figure 4.3.4: Strain-based performance indicator data (2012) $D_{\Delta t=24hours}$

The separate processed Pandas dataframes were then combined (by index represented by a date within the 2011 & 2012 years) into one amalgamated dataframe, consisting of daily-averaged pavement temperatures, daily-aggregated traffic counts for classes 1 to 6 (East & West) and a strain-based performance indicator (SG1 - 9). Finally, any incomplete observations (NaN or and missing data) were removed, resulting in a 'complete' dataset, as illustrated in *Figure 4.3.5*.

Timestamp	Average Temperature (°C)	Class 1 - East	Class 2 - East	Class 3 - East	Class 4 - East	Class 5 - East	Class 6 - East	Class 1 - West	Class 2 - West	Class 3 - West	...	Class 6 - West	SG1	SG2	SG3	SG4	SG5
2012-01-02	6.130697	49	8784	304	131	1479	71	37	7206	158	...	63	723384	5.076596e+06	5.628534e+05	180456.187500	2.278566
2012-01-03	3.617217	43	7843	210	156	1608	87	44	7301	176	...	84	775992	5.524494e+06	1.200359e+06	159717.062500	2.110696
2012-01-04	4.386617	35	8440	202	141	1666	78	39	8543	201	...	86	752714	5.747740e+06	8.342862e+05	161735.640625	2.385279
2012-01-05	4.275696	39	8874	178	177	1502	92	41	10043	283	...	91	721435	4.890578e+06	4.288384e+05	125526.585938	2.229993
2012-01-06	2.822041	48	9296	222	138	1114	74	54	10183	232	...	78	547466	3.831561e+06	3.285336e+05	78462.500000	1.658412

Figure 4.3.5: Processed dataframe consisting of average temperature $T_{\Delta t}$, vehicle frequency $B_{\Delta t}$ and the strain-based performance indicator $D_{\Delta t}$

4.4 Modeling

4.4.1 Cross-Validation

The processed Pandas dataframe, containing aggregated data from temperature, traffic frequency counts and the strain-based performance indicator values, is primarily divided into two sections, namely the training/validation set and the test set.

The training/validation set is used to estimate a set of model parameters which characterise the normal correlation between the explanatory variables $B_{\Delta t}$ & $T_{\Delta t}$, and the response variable $D_{\Delta t}$. The evaluation during the training/validation phase can be done using the k-fold cross validation method, which is used to validate a models performance internally. K-fold cross-validation can be conceptually seen in *Figure 4.4.1*, where the training dataset provided is subdivided into K subsets.

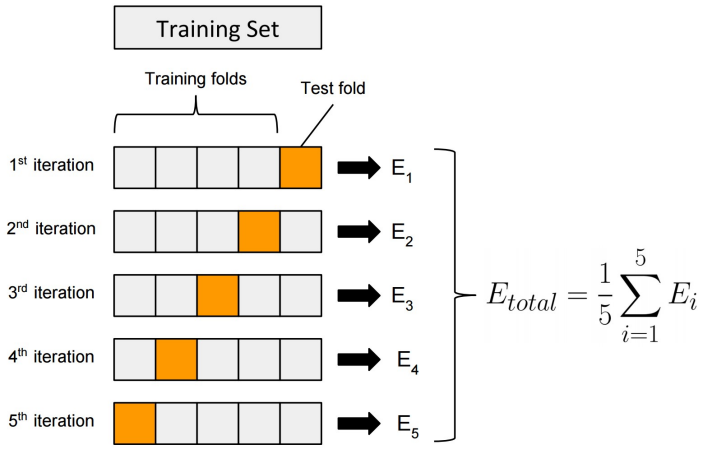

Figure 4.4.1: K-fold cross-validation

A common choice is $K = 10$ (opposed to $K = 5$, for larger datasets) as more training data will then be used in each iteration, which results in a lower bias towards estimating the generalisation performance by averaging the individual model estimates[4]. The advantage of K-fold cross validation is a more stable estimation of model performance during training/validation, reducing the probability of a "lucky" training/validation split in which the model is trained on data and validated on a set of indistinguishable or similar data (when using a training/validation split). The test set is left until the end of modeling process and used as a final means for determining an unbiased estimate on the models performance using data the model has not yet seen before.

4.4.2 Hyper-parameters Tuning

Multilayer Perceptrons can be 'tuned' or configured to increase model accuracy, with the objective to find a set of hyper-parameters for which a model yields good performance (using some predefined metrics) on both the training/validation and test data. Some common hyper-parameters include the choice of activation function, number of hidden layers, the number of units in a layer, the learning rate, momentum and regularisation (among numerous others). On an assumption a machine learning algorithm has n hyper-parameters and each hyper-parameters can take on three values, the total number of possible combinations is 3^n. H_2O's library allows approximately sixty deep learning hyper-parameters to be specified. Assuming a binary choice for each hyper-parameters, a conservative $1.15e + 18$ possible models is estimated. Considering all the alternative hyper-parameters configurations available as a potential candidate set, it becomes clear the space to identify the optimal set of hyper-parameters for a neural network is large and optimisation of these models is non trivial.

Broadly, there are typically three approaches to solve this optimisation problem: a Cartesian search (discrete or exhaustive) which is a 'brute-force' approach, a random grid search or Bayesian optimisation. Cartesian search exhaustively evaluates the hyper-parameters search space, however this is often less desirable as the process can be computationally expensive. The random grid search executes a random search over the hyper-parameter space. The random search, however has been known to return similar results as the Cartesian grid search[23]. Finally the Bayesian approach is considered a more systematic hyper-parameters search, where the algorithms develop a knowledge about the relation between the hyper-parameters settings and model performance in order to make a 'smarter' choice for the next choice of parameter settings. Bayesian approaches, however are know to be computationally expensive, although, for more difficult learning problems has outperformed Cartesian grid or random searches[23]. The H_2O library allows the user to define how many models to be built & evaluated (or time permitted to search), before ending the hyper-parameters search.

4.4.3 Model approach

During the modeling phase, the objective given the time and computational constraints was to complete a hyper-parameters search using random grid search for each strain gauge. That is, to iterate over this process nine times (SG1 - 9) - using the same feature matrix X, however change the target variable y (the strain-based performance indicator) to reflect the strain gauge being considered. This process was repeated twice to ensure the considerations raised in subsection *Preprocessing & model design considerations* were addressed during the research.

4.5 Model Selection & Results

4.5.1 Model Selection

An effective model can be defined as model that optimises a particular performance indicator (or indicators). The candidate models are evaluated and selected against their respective performance(s) on the training/validation and test data.

There are many choices of performance metrics which can be used to effectively determine model performance. Primarily, these metrics in this section have been chosen as they are among the more frequently chosen for ranking regression models (prediction of a continuous value). Second, these metrics were used to evaluate the regression models explored by Farreras Alcover[6], which allows for a more appropriate or suitable comparison of model performance.

Some of the biggest challenges of a data mining project, lie in developing a model of appropriate complexity. This is a common problem during model selection: pick a model that is too simple and it will "under-fit" the data, but pick a model that is too complex and it will "over-fit" the data. Two performance indicators which effectively take this into consideration are the AIC & AICc. The Akaike Information Criterion (AIC) is defined by *equation (4.2).*

Note : In the following equations, the variables m and n represent the number of features used to build the model and the number of datapoints/observations in $T_{\Delta t}$, $B_{\Delta t}$, $D_{\Delta t}$, respectively:

$$AIC = n \cdot \ln\left(\frac{\sum_{i=1}^{n}(y_{observed} - y_{predicted})^2}{n}\right) + 2m \tag{4.2}$$

The AIC metric is biased for small samples. For this reason, a bias corrected version referred as AICc, the Akaike Information Criterion (Corrected), can be used for evaluating large samples[24]. The AIC & AICc penalise more complex models - thus, a lower AIC or AICc values for a given model, implies a less complex model which for particular applications can be more desirable. The AICc performance metric is defined as:

$$AIC_c = AIC + \frac{2m \cdot (m+1)}{n-m-1} \tag{4.3}$$

Another common metric for model performance for regression models is the Mean Squared Error (MSE) represented by *equation (4.4)*. The MSE quantifies a measure of the goodness of fit of a model, where a model with lower MSE value implies a more accurate model:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_{observed} - y_{predicted})^2 \tag{4.4}$$

The Mean absolute percentage error (MAPE) is another common metric for evaluating regression model performance and has been defined in *equation (4.5)*. The MAPE value is the average absolute error expressed as a percentage and quantifies in relative terms a measure of the goodness of fit of a model. As with the MSE performance metric, a model with a lower MAPE values implies a more accurate model:

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_{observed} - y_{predicted}}{y_{observed}}\right| \cdot 100 \tag{4.5}$$

Finally, an additional performance indicator Ψ is used to assess the performance of the models in predicting the cumulative values of $D_{\Delta t}$ (expressed as a percentage). Here the numerator represents the cumulated value of $D_{\Delta t}$ predicted by the model and the denominator the analogous quantity calculated considering the monitored strain signal:

$$\psi = \frac{\left\{\sum_{t=1}^{t_f} D_{\Delta t}(t)\right\}_{predicted}}{\left\{\sum_{t=1}^{t_f} D_{\Delta t}(t)\right\}_{observed}} \tag{4.6}$$

4.5.2 Results

The performances of the two approaches outlined in *4.3.2 (1.)* of subsection *Preprocessing & model design considerations*, are described in this section. More specifically, the development of models using all 13 input features (Class 1 - 6 East, Class 1 - 6 West, temperature) in addition to the development of models using 5 input features (Class 5 East, 6 East, Class 5 West, 6 West, temperature). The performances of the models developed during the research are compared to the performances of the results found by Farreras Alcover.

The random grid search using H_2O, returns a collection of models (the number of models to be built for each strain gauge can be defined). From this collection, the models are ranked according to the cross-validation mean square error (MSE) and the 2nd top MSE performance model was selected to represent each of the nine strain gauges. The selected models are then used to calculate the other respective performances (using the defined performance metrics) on the training/validation and test data.

13 input features:

Table 4.5.1 shows the performance results of the models selected (from 600 models for each strain gauge) using 13 features, and their performance summary and comparison against the multiple linear regression models developed by Farreras Alcover[6]

	Training/validation				Test			
Model	**AICc**	**MSE**	**MAPE**	**Ψ(%)**	**AICc**	**MSE**	**MAPE**	**Ψ(%)**
SG1 - linear regression	-703.5	3.41E+11	15.9	98.3	N/A	2.08E+11	35.1	106.9
SG1 - MLP	-37.4	9.18E+11	47.8	103.2	N/A	2.13E+11	80.9	118.1
SG2 - linear regression	-911.7	6.87E+11	8.8	102.1	N/A	1.52E+12	35.0	119.1
SG2 - MLP	-37.9	9.13E+11	13.5	103.7	N/A	1.44E+12	31.9	115.0
SG3 - linear regression	-464.9	2.00E+11	27.1	103.0	N/A	1.26E+11	41.5	99.5
SG3 - MLP	-36.2	3.99E+11	56.4	111.1	N/A	4.41E+11	79.1	94.1
SG4 - linear regression	-615.1	3.48E+10	22.2	97.6	N/A	7.01E+10	32.4	104.5
SG4 - MLP	-33.4	1.03E+11	115.1	108.2	N/A	4.41E+11	136.8	105.1
SG5 - linear regression	-799.3	1.76E+10	13.5	99.2	N/A	5.05E+11	45.1	125.2
SG5 - MLP	-41.2	1.03E+11	79.6	21.4	N/A	2.43E+12	80.0	22.6
SG6 - linear regression	-641.2	3.17E+11	20.4	97.5	N/A	5.00E+11	36.8	111.7
SG6 - MLP	-39.4	1.03E+11	66.4	42.7	N/A	7.12E+11	71.9	43.4
SG7 - linear regression	-622.2	3.54E+11	22.1	97.2	N/A	3.70E+11	54.6	122.0
SG7 - MLP	-40.0	1.03E+11	69.7	36.1	N/A	7.44E+11	71.6	39.6
SG8 - linear regression	-900.2	1.31E+12	10.0	99.9	N/A	4.05E+12	38.6	116.7
SG8 - MLP	-46.6	1.03E+11	93.6	6.7	N/A	4.66E+13	94.1	6.7
SG9 - linear regression	-795.9	1.08E+11	13.2	98.9	N/A	1.43E+10	32.9	107.3
SG9 - MLP	-39.4	1.03E+11	66.8	38.5	N/A	1.09E+12	68.6	36.9

Table 4.5.1: Model performance summary for strain gauges 1 to 9 (600 models built for each strain gauge) - developed using 13 input features

In addition, the results have been plotted in *Figures 4.5.1 - 4.5.4* for easier interpretation of the performance of the MLP regression models developed across strain gauges 1 to 9, compared to the performance of the multiple linear regression models developed by Farreras Alcover for the same strain gauges.

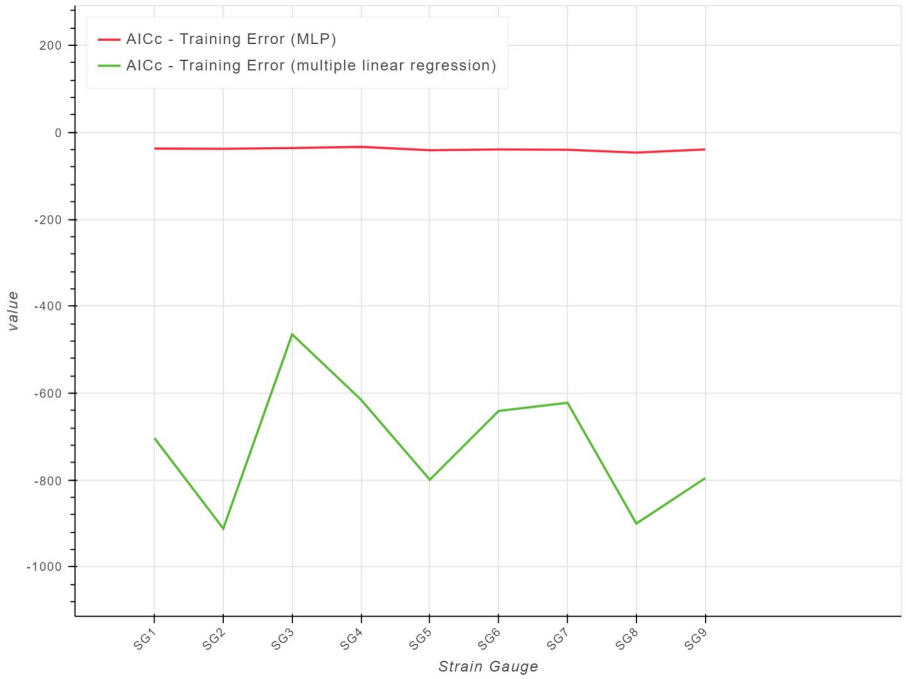

Figure 4.5.1: Plot of model AICc performance - 13 input features

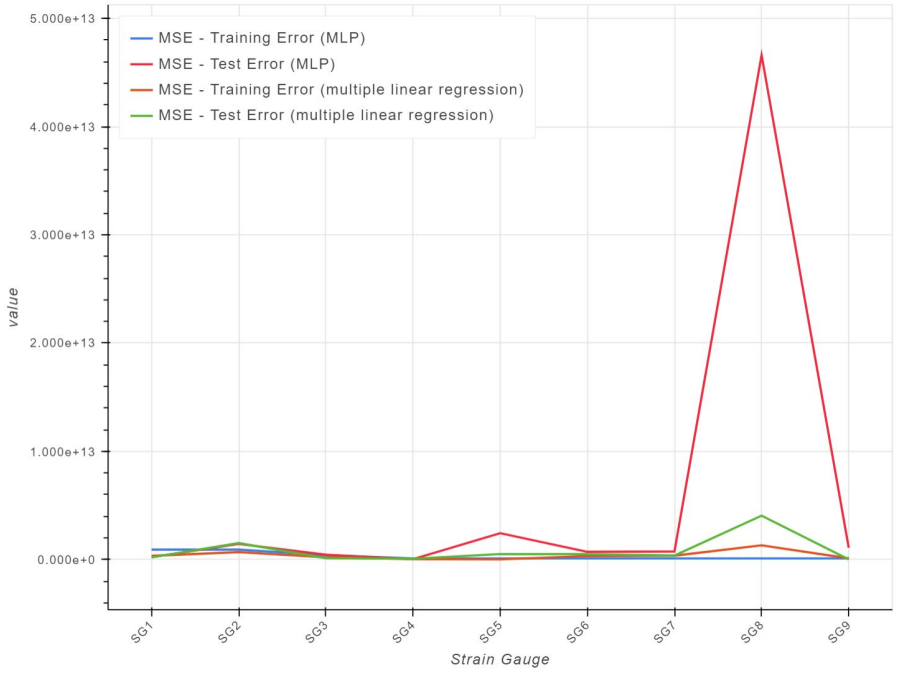

Figure 4.5.2: Plot of model MSE performance - 13 input features

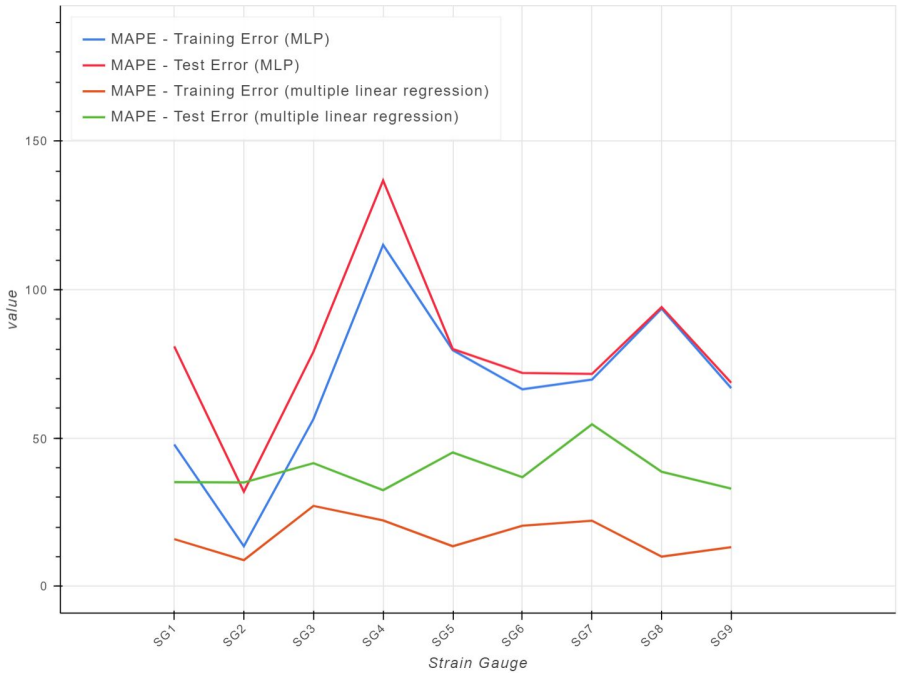

Figure 4.5.3: Plot of model MAPE performance - 13 input features

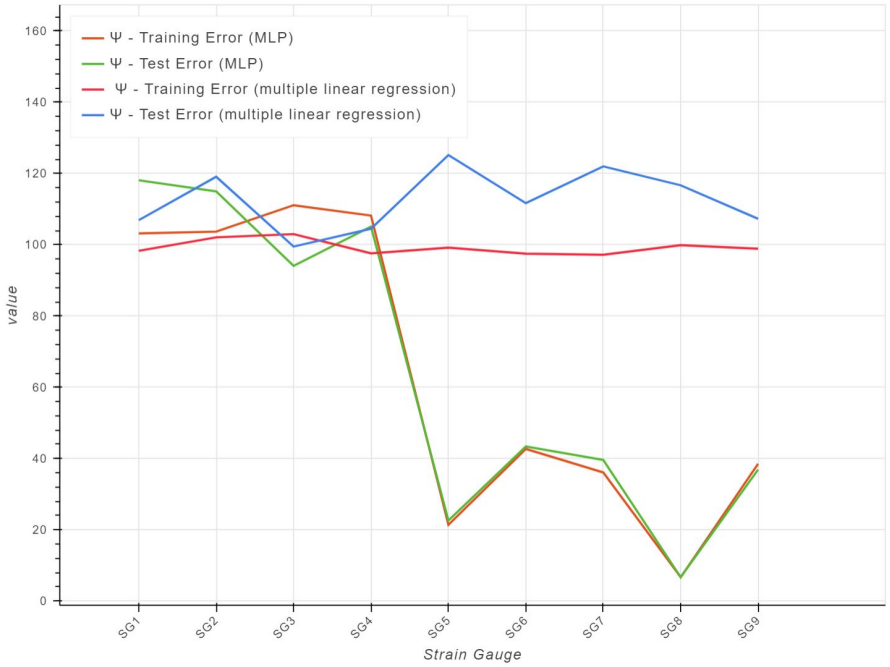

Figure 4.5.4: Plot of model Ψ performance - 13 input features

5 input features:

Similarly, the models were produced using 5 features as inputs, the results populated in *Table 4.5.2* and the appropriate graphs plotted in *Figures 4.5.5 - 4.5.9.*(*Figure 4.5.6* was repopulated omitting SG8 which skewed the scale - this has been shown in *Figure 4.5.8*)

	Training/validation				Test			
Model	**AICc**	**MSE**	**MAPE**	**Ψ(%)**	**AICc**	**MSE**	**MAPE**	**Ψ(%)**
SG1 - linear regression	-703.5	3.41E+11	15.9	98.3	N/A	2.08E+11	35.1	106.9
SG1 - MLP	-55.4	8.88E+11	58.2	101.5	N/A	2.40E+11	99.3	108.3
SG2 - linear regression	-911.7	6.87E+11	8.8	102.1	N/A	1.52E+12	35.0	119.1
SG2 - MLP	-55.3	8.69E+11	12.3	101.2	N/A	1.38E+12	37.7	114.1
SG3 - linear regression	-464.9	2.00E+11	27.1	103.0	N/A	1.26E+11	41.5	99.5
SG3 - MLP	-54.0	3.62E+11	51.5	108.9	N/A	4.68E+11	53.0	91.6
SG4 - linear regression	-615.1	3.48E+10	22.2	97.6	N/A	7.01E+10	32.4	104.5
SG4 - MLP	-51.4	9.04E+10	95.4	99.8	N/A	4.68E+11	120.2	104.2
SG5 - linear regression	-799.3	1.76E+10	13.5	99.2	N/A	5.05E+11	45.1	125.2
SG5 - MLP	-58.7	9.04E+10	78.6	19.8	N/A	2.49E+12	78.5	22.4
SG6 - linear regression	-641.2	3.17E+11	20.4	97.5	N/A	5.00E+11	36.8	111.7
SG6 - MLP	-57.0	9.04E+10	61.2	39.4	N/A	7.15E+11	65.3	43.0
SG7 - linear regression	-622.2	3.54E+11	22.1	97.2	N/A	3.70E+11	54.6	122.0
SG7 - MLP	-57.6	9.04E+10	64.9	33.3	N/A	7.58E+11	66.0	39.3
SG8 - linear regression	-900.2	1.31E+12	10.0	99.9	N/A	4.05E+12	38.6	116.7
SG8 - MLP	-64.1	9.04E+10	93.8	6.2	N/A	4.67E+13	93.5	6.6
SG9 - linear regression	-795.9	1.08E+11	13.2	98.9	N/A	1.43E+10	32.9	107.3
SG9 - MLP	-57.0	9.04E+10	63.4	35.5	N/A	1.10E+12	63.4	36.6

Table 4.5.2: Model performances for strain gauges 1 to 9 (600 models built for each strain gauge) - developed using 5 input features

Figure 4.5.5: Plot of model AICc performance - 5 input features

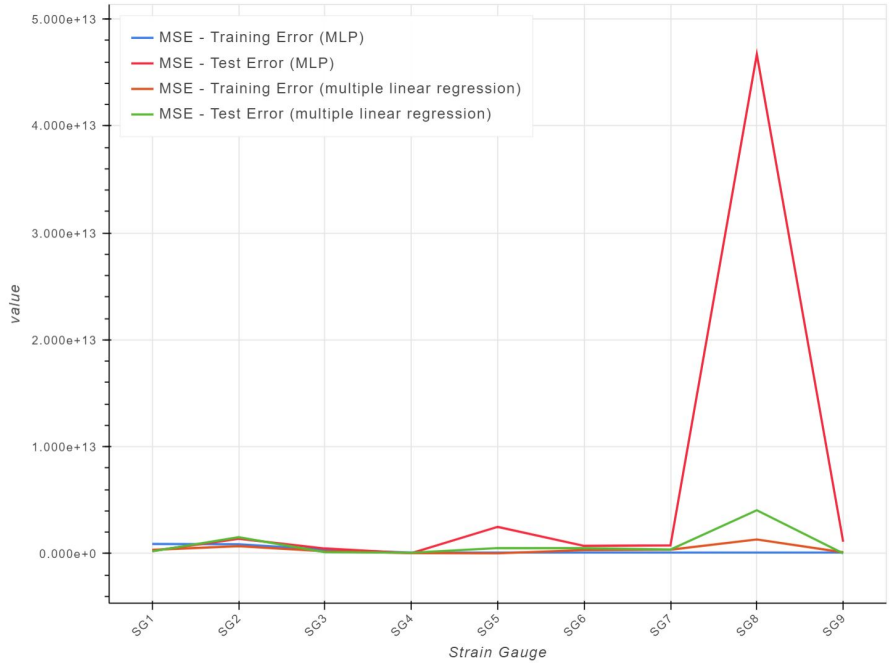

Figure 4.5.6: Plot of model MSE performance - 5 input features

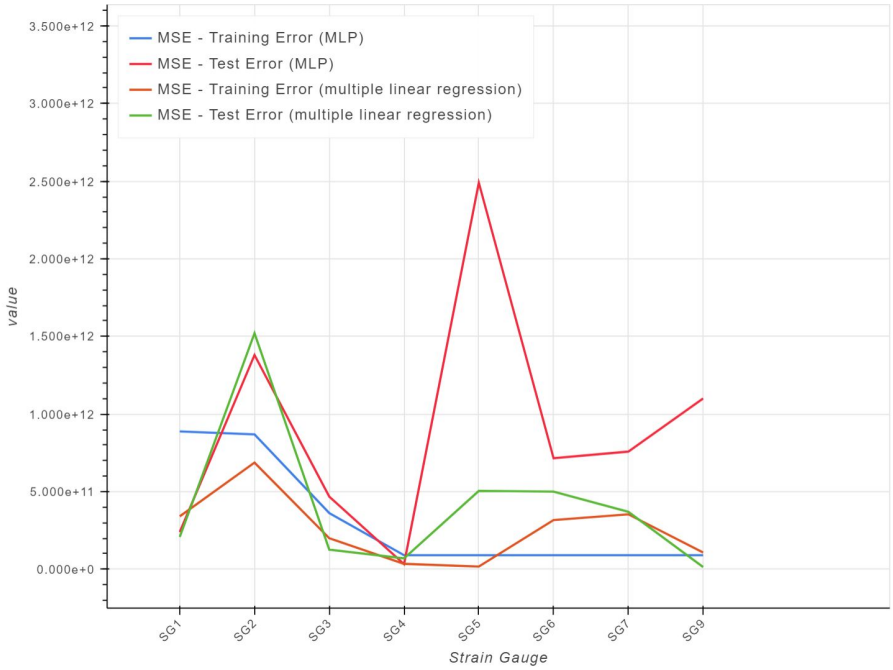

Figure 4.5.7: Plot of model MSE performance (omitting SG8) - 5 input features

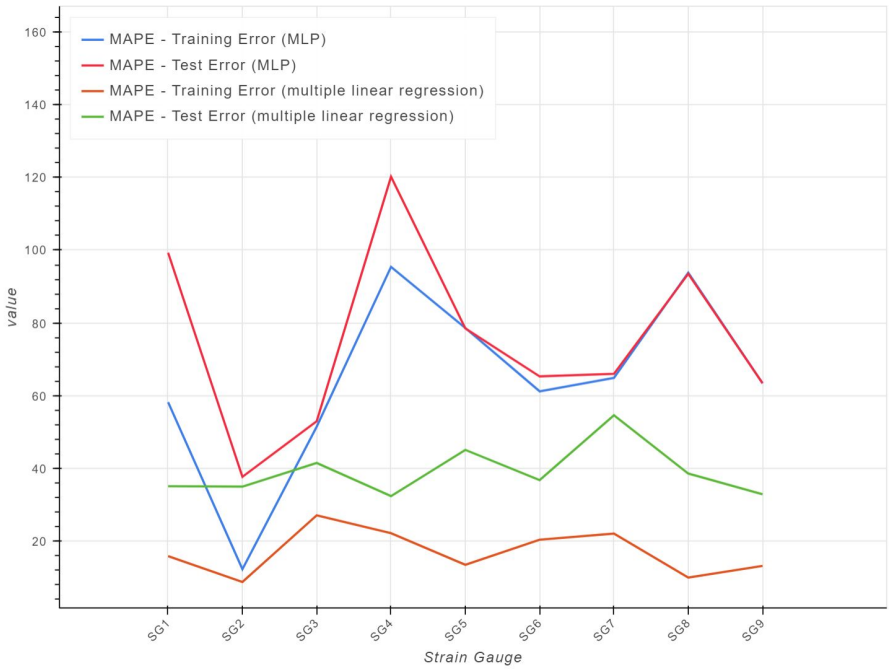

Figure 4.5.8: Plot of model MAPE performance - 5 input features

Figure 4.5.9: Plot of model Ψ performance - 5 input features

The greatest achievement of our technology may well be the creation of technology that allow us to go beyond engineering - that allow us to create more than we can understand.

- Danny Hills, The Pattern on the Stone

5 Conclusions

5.1 Summary

In summary, an opportunity was presented to investigate the performance of machine learning algorithms as a modern local data-based approach to structural health monitoring. More specifically, multilayer perceptron machine learning algorithms were trained on monitoring outcomes from the Great Belt Bridge in Denmark in order to evaluate their performance for predicting a strain-based performance indicator (proportional to S-N fatigue damage), using temperature and traffic frequency data alone.

The thesis described the process followed to prepare and clean the data required to train and evaluate a series of multilayer perceptron models. Appropriate performance metrics were defined after which the hyperparameter search space for nine individual strain gauges was explored using random discrete grid search. For each search space, the models were listed according to their training/validation MSE score and the 5^{th} model was selected to represent the MLP for each strain gauge. The collection of nine models were then used to populate a summary of training/validation and test performances. Finally, the MLP model results were graphically reviewed, discussed and compared to the performance of multiple linear regression models developed by Farreras Alcover[1]. On average however, the multilayer perceptron models were

unable to reproduce (or improve upon) the performance of the multiple linear regression models developed by Farreras Alcover[1].

5.2 Evaluation

The objective of this research was to evaluate the performance of multilayer perceptron machine learning algorithms in predicting the values of a strain-based performance metric. In addition, it was suggested the performance of the MLP was to be compared to the performance of the multiple (linear) regression used by Farraras Alcover, in order to understand the contexts in which multiple linear regression may be more or less suited, than the MLP, in effectively capturing or characterising relationships in data for predictive tasks.

The research concluded that the models developed for this particular application don't seem to offer any observable benefit over the multiple regression models and, on average, the performance of the multiple regression models exceeded the performance of the multilayer perceptron models. This was concluded after the following key observations were made when considering the models built with 5 inputs:

- The AICc value of the multiple regression models exceeded the AICc value for the multilayer perceptron models across all the strain gauges. This was expected as the AICc performance metric is defined such that more complex models are penalised, where MLP models are by design more complex than multiple regression models.
- The MSE values across the strain gauges were generally greater than those MSE values from the multiple regression models. The MSE training/validation error on the majority of models found was significantly less than the test error, suggesting the models may have overfit the training/validation data. In addition, with models built on 5 or 13 input variables for SG4 - 9 (omitting SG8), the training error plateaued which was not expected and the erro is suspected to be due to a coding error.
- The MAPE values for the MLP models generally exceeded those values exhibited by the multiple regression for both training/validation & test sets. In addition the MLP training/validation MAPE values across all strain gauges was equal to or lower than the test MAPE value, again suggesting the models may have overfit.
- The performance values for psi (Ψ) in the MLP models were lower in both the training/validation & test set for 6 of the 9 strain gauges.

During this investigation the following tasks were accomplished and theory explored:

- Data cleaning, aggregation of datasets
- Exploratory data analysis and data visualisation
- Theory of multiple linear regression machine learning algorithms for regression tasks
- Theory of multilayer perceptron machine learning algorithms for regression tasks
- Hyperparameter optimisation of neural networks, using grid search, random search and Bayesian optimisation techniques.
- Model performance metrics for model evaluation or comparison
- Theory of stress behaviour of welds in civil structures
- The CRISP data mining approach for systematically completing data mining projects
- The computational considerations or limits in hyperparameter search and optimisation

In addition, the following tools or technologies implemented:

- The typeset language Latex
- Python scripting language
- Jupyter notebook for exploratory data analysis and data science tasks
- Python libraries including Pandas, Numpy and H_2O (among others)

5.3 Future work

It became clear during the research, the importance of reproducibility for data mining projects. The development of these models typically involve numerous design decisions and in addition, the models are highly sensitive to the impact of these decisions. As such, careful consideration should be taken when making these design decisions and it becomes the responsibility of the researcher to effectively capture the logic and assumptions made during the process. In addition, it was observed that it is critically important to understand the statistical processes, the mathematics behind the machine learning and optimisation algorithms and the numerous effects hyperparameters can have on a given model.

Futhermore, three particular limitations were noted during the course of this data mining task, namely *i*) the limitation in computational power *ii*) the limited time during which to explore alternative machine learning models/hyperparameters *iii*) the quality/quantity of the data provided. Given these limitation, the following points are identified for recommendations for future work:

i.) Explore more exhaustively the hyperparameter search space using distributed computing or graphics processing units (GPUs). Assuming more resources are available to develop and evaluate a greater selection of candidate models, the probability of finding a selection of higher performance model in the search space, is improved. Alternatively, the grid search stop criteria could be relaxed, allowing more of the search space to be explored.

ii.) Consider the implementation of alternative Gradient Boosting Machine (GBM) or Random Forest algorithms for the regression task presented. Traditionally, neural networks have been found incredibly effective at tasks such as feature extraction in image recognition, for example. The argument that particular algorithms serve specific use cases better, may hold some validity.

iii.) Evaluate MLP's as potential candidate models when more comprehensive, representative data is available which characterises the normal behaviour patterns. As the 'complete' aggregated data from 2012 (training/validation) & 2011 (test) contained approximately 224 & 71, respectively - these data represent a sample of the population characteristic relationships for a full year in 2012 of 366 days & in 2011, 365 days. A dataset size of 224 observations for development of neural network models, could be considered small data for today's standards and it is critical to stress the importance of the quality and representativeness of data when training these models.

Finally, it is suggested a more deliberate or individual approach be taken to understand the performance of MLP's for prediction of the strain-based performance indicator by focusing on one strain gauge at a time. Assuming this approach is taken, one could more strategically balance the bias/variance tradeoff in pursuit of a set of better hyperparameters and hence a more effective model which is able to generalise well to unseen data.

References

[1] Isaac Farreras-Alcover, Marios K Chryssanthopoulos, and Jacob E Andersen. Data-based models for fatigue reliability of orthotropic steel bridge decks based on temperature, traffic and strain monitoring. *International Journal of Fatigue*, 95:104–119, 2017.

[2] DFKI 2011. From past to present: Industry 4.0. http://www.ebnonline.com/author.asp?section_id=3443&piddl_msgorder=asc&doc_id=270310&image_number=1, 2011. [Online; accessed August 10, 2017].

[3] Storebælt. The great belt bridge, denmark. https://www.storebaelt.dk/english/bridge, 2013. [Online; accessed August 10, 2017].

[4] Sebastian Raschka. *Python Machine Learning*. Packt Publishing, 2015.

[5] Smart Vision - Europe. What is the crisp-dm methodology? http://www.sv-europe.com/crisp-dm-methodology/, 2016. [Online; accessed August 10, 2017].

[6] Isaac Farreras Alcover. *Data-based models for assessment and life prediction of monitored civil infrastructure assets.* PhD thesis, University of Surrey, January 2014.

[7] Klaus Schwab. *The fourth industrial revolution*. Crown Business, 2017.

[8] Big Data. for better or worse: 90% of world's data generated over last two years. *SCIENCE DAILY, May*, 22, 2013.

[9] Mike Hazas, Janine Morley, Oliver Bates, and Adrian Friday. Are there limits to growth in data traffic?: On time use, data generation and speed. In *Proceedings of the Second Workshop on Computing within Limits*, page 14. ACM, 2016.

[10] Rolls-royce and microsoft collaborate to create new digital capabilities. https://customers.microsoft.com/en-US/story/rollsroycestory. Accessed: 2017-08-10.

[11] John Deere. John deere is revolutionizing farming with big data. https://datafloq.com/read/john-deere-revolutionizing-farming-big-data/511, Unknown. Accessed: 2017-08-10.

[12] Charles R. Farrar and Keith Worden. *Structural health monitoring a machine learning perspective*. Wiley, 2013.

[13] Nikolaos Dervilis. *A machine learning approach to structural health monitoring with a view towards wind turbines*. PhD thesis, University of Sheffield, Nov 2013.

[14] AJ Weight. Critical analysis of the great belt east bridge, denmark. In *TBC*, 2009.

[15] Isaac Farreras Alcover, Marios K Chryssanthopoulos, and Jacob Egede Andersen. Regression models for structural health monitoring of welded bridge joints based on temperature, traffic and strain measurements. *Unknown*, 14, 10 2015.

[16] Glenn A. Washer Ken P. Chong, Nicholas J. Carino. Health monitoring of civil infrastructures, 2001.

[17] Charles R Farrar and Nick A.J Lieven. Damage prognosis: the future of structural health monitoring. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 365(1851):623–632, 2007.

[18] AiQun Li, YouLiang Ding, Hao Wang, and Tong Guo. Analysis and assessment of bridge health monitoring mass data—progress in research/development of "structural health monitoring". *Science China Technological Sciences*, 55(8):2212–2224, Aug 2012.

[19] Robert James Barthorpe. *On model-and data-based approaches to structural health monitoring*. PhD thesis, University of Sheffield, 2010.

[20] JE Andersen, M Enckell, I Carreras Alcover, and MK Chyssantopoulos. The structural health monitoring system of the izmit bay bridge: overview and shm–based fatigue assessment. In *Second Conference on Smart Monitoring, Assessment and Rehabilitations of Civil Structures, SMAR*, 2013.

[21] Jean Gaudart, Bernard Giusiano, and Laetitia Huiart. Comparison of the performance of multi-layer perceptron and linear regression for epidemiological data. 44:547–570, 01 2004.

[22] Pete Chapman, Julian Clinton, Randy Kerber, Thomas Khabaza, Thomas Reinartz, Colin Shearer, and Rudiger Wirth. CRISP-DM 1.0 Step-by-step data mining guide. Technical report, The CRISP-DM consortium, August 2000.

[23] James S. Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 2546–2554. Curran Associates, Inc., 2011.

[24] K.P. Burnham and D.R. Anderson. *Model Selection and Multimodel Inference: A Practical Information-Theoretic Approach*. Springer New York, 2003.

Attestation

I understand the nature of plagiarism, and I am aware of the University's policy on this.

I certify that this dissertation reports original work by me during my University project except for the following (*adjust according to the circumstances*):

- The equations in *Chapters 2* & *3* were interpreted from Andrew Ng's Machine Learning Course (Coursera) and some Latex code taken from the following URL: *http://www.luckycallor.com/index.php/2015/12/22/summary-of-course-machine-learning-by-andrew-ng-on-coursera/*
- Part of the code was inspired from examples on H2O's website, specifically from the following H2O GitHub repository - URL: *https://github.com/h2oai/h2o-3/blob/master/h2o-py/h2o/grid/grid_search.py*
- The thesis template was modified from the original Harvard PhD template which is licenced for distribution under the permissive AGPL licence - available from the following URL: *https://github.com/suchow/Dissertate*

Signature *David E. Haveron* Date *5th September 2017*