The road to intelligent asphalt concrete mixture design:
A Data driven analysis of common asphalt concrete property prediction methods and a solution to the inverse problem

Course name:	**MSE thesis**
Course code:	**MS53035**
Instructor:	**Dr. M.H.F. Sluiter**
Institution:	**Delft University of Technology**
Department:	**Materials Science and Engineering, 3ME**
Public defence:	**22 June 2023**
Commission:	**Dr. M.H.F. Sluiter** **Dr. G.H.J. Langejans** **Dr. S. Kumar** **Ir. R.N. Khedoe**
Student:	**L.J. Hopman**
Student number:	**4214161**

Acknowledgements

I would like to express my sincere gratitude and appreciation to all those who have supported me throughout the journey of conducting this research and completing my thesis on the use of machine learning in asphalt concrete mixture recipes.

First and foremost, I extend my heartfelt thanks to DIBEC who took me in precisely when I needed them to. The valuable guidance and deep conversations I've experienced since my start made this thesis possible and made me a better engineer. The access to their resources for this work was invaluable, though I hope my contributions made it mutually beneficial. I'd like to especially thank Jan, Stefan, Thomas and Arjan, who were there with me from the start, and Radjan and Toni for their technical insights.

My sincere gratitude also goes to the Delft University of Technology. Going there was a dream come true. I would like extend a special thank you to Dr. Marcel Sluiter who always greeted me with kindness and though questions. The support, understanding and patience I've met with from the faculty staff, especially Saskia van der Meer made it possible for me to do this. Finally I'd like to thank Dr. Michael Janssen who, when I told him of my plan years ago said to me: "Dat gaat je lukken". Those words held up.

I am also grateful to my friends and family. My parents, stepparents, brothers and sisters who have kept on supporting me and humouring my ramblings on ML modelling, asphalt concrete and data analyses with kindness and patience. Special thanks goes to my friends Bram, Kayin and Nick for their invaluable role, providing me with much needed relaxation, laughter and companionship.

Lastly I would like to express my deepest appreciation to my girlfriend Jeanine for her unwavering support and understanding during the countless hours spent researching, analysing data and writing. Her love, encouragement and belief in my abilities were a constant source of motivation and I am incredibly fortunate to have her by my side.

To all those mentioned above and to anyone else who has contributed to this thesis in one way or another, please accept my deepest appreciation and heartfelt thanks.

Contents

Abstract

Asphalt concrete is one of the most widely used materials in modern road construction. Predicting its functional properties is crucial in the design of new asphalt concrete mixtures. However, current prediction models are limited in accuracy and applicability due to the complex nature of asphalt concrete properties. This thesis researches the use of machine learning algorithms to greatly improve upon existing prediction models. The input is limited to standardized test results in line with Dutch regulations, the output focusses on functional design parameters including stiffness, fatigue resistance, water sensitivity and resistance to permanent deformations. The performance of several machine learning algorithms and the effects of different regression methods are compared. Furthermore, a solution is found for the inverse problem, which allows for greater flexibility when using the models to design new asphalt concrete mixtures. The results show that machine learning algorithms outperform traditional models on accuracy while simplifying the model input parameters. Machine learning algorithms were also able to predict a greater range of output parameters, most of which with a high accuracy. The analysed possibility of modelling asphalt concrete mixtures directly from their desired functional properties is shown to be promising. The proposed machine learning models and their inverse problem counterparts have the potential to greatly improve the accuracy and practical usability of the prediction of asphalt concrete properties, ultimately leading to better mixture design and more durable roadways.

1. Introduction

1.1. Project introduction

The inspiration for this project is two-fold. The first is the new *Pavement Information Modelling s*oftware (PIM-software) for pavement constructors and engineers, which has gradually been implemented since 2019 (CROW, 2021). This new software, produced by the cooperation of 8 road construction companies in the Netherlands, created the opportunity for a big data analysis. The second inspiration comes from recent studies into the link between asphalt mixture recipes and their test results. These focussed on either improving the accuracy of existing models or using *inverse problem theory* to back-calculate the effects of recycled asphalt concrete, usually with a small sample size.

The objective of this work is to chart the relationship between asphalt concrete constituents and the asphalt concrete functional properties both for the forward problem and the inverse via an empirical big data analysis of standardised Dutch asphalt concrete mixtures. The goal is to do these analyses with only the data available in the Dutch standardised type tests and to use automated analysis via machine learning to get the results. Limiting the scope to the information used in standardised testing is an important feature as this allows other companies or students to easily replicate the results.

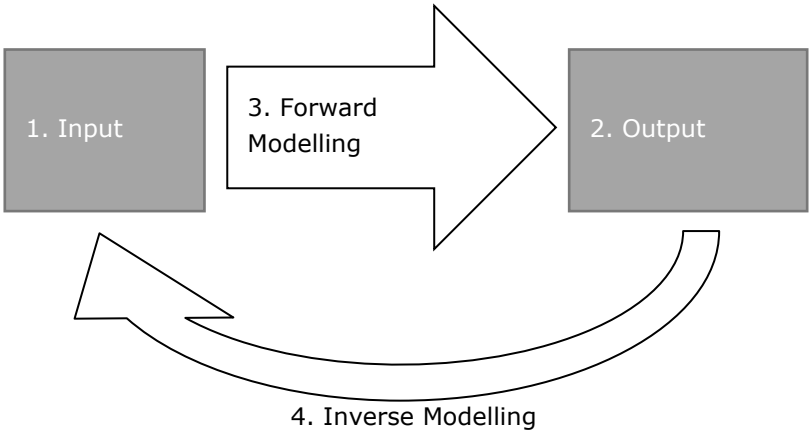


Figure 1-1: General analysis model for the thesis

Figure 1-1 forms a clear outline for the thesis work, 1. Input and 2. Output will be discussed in chapter 2.1 and 2.2 respectively. The forward and inverse models will be discussed in chapters 3.2 and 3.3.

In general, the thesis is summarised in 4 sections: Input, Output, Forward Modelling and Inverse Modelling. The Input consists of the *weight* ratio and properties of materials which are mixed to get asphalt concrete. The Output consists of the standardised test results which are used in the Dutch pavement design methodology. Datasets containing Input and Output (found through testing) are available for use in the thesis. After compiling a dataset with both input and output, the links between these are sought. First in forward modelling where the test results are predicted from the asphalt mixture recipe. Then a solution to the inverse problem is sought where, by giving the desired test results an asphalt mixture may be found. While there have been solutions for the forward model in past research, this thesis looks to more output properties and keeps the scope limited to standardised tests and mixtures, as available in the PIM-environment. This short project description is further explained in the following paragraphs. The primary literature study in chapter 2 tells the scope and how that defines the in- and output for the models. The secondary literature study in chapter 3 tells first how forward modelling is used to find the link between input and output in the specific case of asphalt concrete and functional property prediction. Chapter 3 also described how inverse modelling may be used in this specific case and how it has successfully been used in similar cases.

1.2. Research questions

1.2.1. The goal

The main research question is: 'How can a structural pavement design be made, based exclusively on its given design parameters and the available mixture ingredients?' A solution can be found by studying varying asphalt property prediction models via a big data analysis.

The setup of this research first requires both a quick analysis script, using multiple asphalt mixture models, and a machine learning algorithm, so that a similar analysis in the future will only need the data in the right format. Secondly, it needs a script that derives possible asphalt mixture recipes from desired test results. This inverse modelling analysis script will either work with constants that are derived from the forward modelling analysis or with a machine learning algorithm.

This requires the following sub-goals:

- Building a testing and training database from the PIM-environment;
- A complete worked out example of such a forward modelling analysis;
- A forward sensitivity analysis on the materials used;
- A worked out inverse modelling analysis;
- A sensitivity analysis on the inverse solution.

There are (at least) two possible flaws when working with machine learning that depend on the database:

1. Too little data for a successful machine learning algorithm;
2. Too much noise in the data for a concrete analysis.

If the database contains too little data for a successful machine learning algorithm, there are ways to simplify the work for the algorithm so that a result can be achieved with less data. If there is too much noise for a concrete analysis, this also will be a valuable result. As the link between mixture recipes and functional properties is already established (in the use of type tests), noise or a lack of reliability in the models may be because the input parameters aren't distinctive. Should this turn out to be the case, the reason why must be explored extensively.

1.2.2. Research questions

The overarching research question this thesis is going to be dealing with is: 'How can a structural pavement design be made, based exclusively on its given design parameters and the available mixture ingredients?' A main requirement for coming to an answer is an inverse modelling analysis of asphalt property prediction models, to then build on to.

Adding to this analysis, this question requires a three-fold attack roughly categorised as follows (in the order in which they need to be solved (1 first, 3 last)):

1. Big data analysis on modern modelling techniques;
2. Machine learning algorithm vs modern modelling techniques;
3. Inverse problem solving.

These three main research categories are tackled by asking the following main- and sub research questions.

1. How accurate are modern asphalt mixture modelling techniques when calculating Dutch asphalt mixtures (forward problem)?
   1.1. Which asphalt mixture modelling techniques are generally used?
   1.2. What additional material characteristics do these modelling techniques require?
   1.3. How can the output of these models be translated into functional properties?
   1.4. How accurate can these models predict the functional properties from the asphalt mix design?
2. How does a machine learning model stack up against conventional models (addition to the forwards problem)?

2.1. Which machine learning algorithms may be used?
2.2. Is additional material-information needed?
2.3. Can the functional properties be simplified?
2.4. Is it needed to link certain parameters?
2.5. How accurate can the algorithm predict the mixture properties?
2.6. What improvements may be considered in the future?
2.7. How does the machine learning algorithm compare to the conventional modelling techniques?

3. How accurate can inverse problem solving predict asphalt mixtures from their OIA-properties?
3.1. Which asphalt mixture modelling techniques are good enough for this method (including the machine learning algorithm)?
3.2. What additional material characteristics do these modelling techniques require?
3.3. How can the output (asphalt mixture recipes) best be visualised?
3.4. How does the code to predict asphalt mixture recipes from their OIA-properties work?
3.5. How does the accuracy of the predictions compare to the forward problem?

After answering these research questions the thesis is concluded. Recommendations for further research are included afterwards.

1.3. Approach

1.3.1. Critical path

A distinction is made between the main path and the critical path. The main path is based on all work being done by 1 person. The critical path shows the interdependencies of the work, visualised in Figure 1-2.

The main path is as follows:

- Startup:
  - Finishing up the research methodology;
  - Literature study on conventional asphalt mixture modelling;
  - Literature study on usable machine learning algorithms;
  - Literature study on a usable inverse problem theory;
- Database-building:
  - Set up the databases;
- Coding:
  - Write the code for the forwards problem with conventional predictive models;
  - Implementing the machine learning algorithms;
  - Implementing a usable inverse problem theory;
- Testing and analysis:
  - Acquire the results;
- Analysis of the database:
  - Analyse the results;
  - Discussions on the results;
  - Draw conclusions and give recommendations.

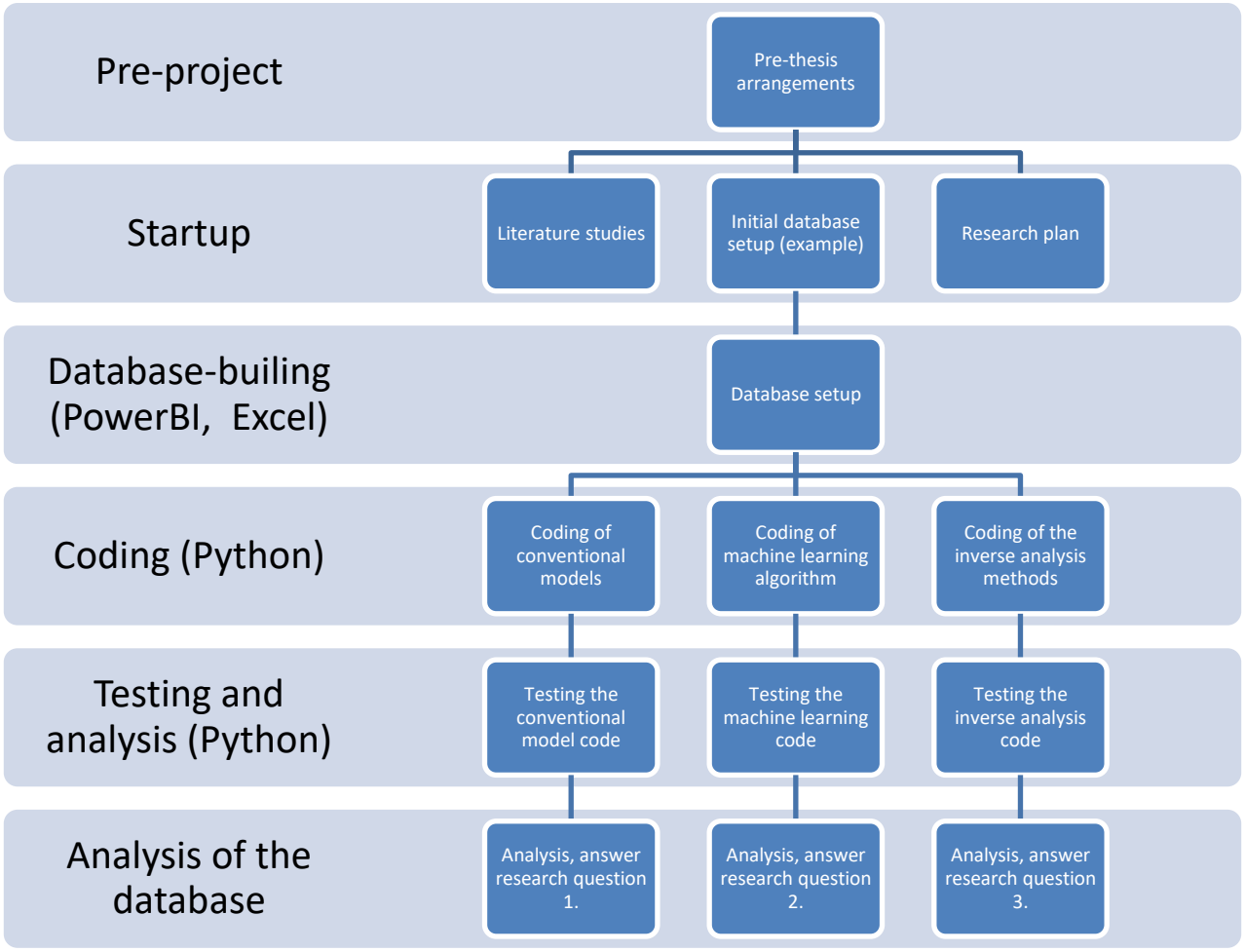


Figure 1-2: Project flowchart

1.4. Scope

The scope is defined per category.

Database sources: The database is limited to the Ballast Nedam database. In the initial idea the cooperation in the PIM environment allowed for the use of the databases of multiple companies. Keeping all the data strictly confidential per companies is an important additional risk for these companies. So, the choice is made to work with 1 dataset first and only expand when needed. No expansion beyond the Ballast Nedam database is needed to get significant results.

Limits to the database: the data used is limited to standardised testing. The purpose of this is to develop models that are easy to use by companies in the future, without the need for additional tests. The scope is sharply defined by standardised tests for asphalt concrete in the Netherlands, further limited to asphalt concrete mixtures that are tested for stiffness and fatigue tests only.

Models: The predictive models are limited to models already used successfully on asphalt concrete databases in the past (multiple linear regression and the catboost machine learning algorithm) (Martini, 2019) and adjacent models that increase the analytical insight in the power of these models (sklearn ridge model and the sklearn decision tree model).

1.5. Chapter overview

Chapter 1 introduces the thesis with a research approach and the research questions. Chapter 2 lists the primary literature. Primary literature is defined as literature that defines the scope and consists of the standardisations of Dutch standardised asphalt concrete tests and information on the pavement engineering design methodology as standardised in Dutch pavement design software. Chapter 3 continues the literature study with secondary literature. The primary literature contains standardised norms and pavement modelling techniques as enforced by the Dutch governing bodies while the secondary literature study contains research-oriented literature. Chapter 4 starts with an overview of the database, and the pre-processing steps undertaken to get a final database ready to be analysed. The final database has a clearly defined input and output as shown (1 and 2) in Figure 1-1. Chapter 5 contains the analytical work on the forward model, shown as (3) In Figure 1-1. Firstly, conventional models are fitted to the database, then machine learning models are trained and reviewed. Chapter 6 analyses solutions to the inverse problem and gives a practical use-case for this technique. Shown as (4) in Figure 1-1. Chapter 7 summarises the results, answers the research questions. Chapter 7 also contains the recommendations for further research.

2. Primary literature review

This primary literature review focusses on the *input* and *output* structure of the general thesis model, as shown in Figure 1-1. For clarity a brief introduction in standard Dutch pavement designs is given in the transversal cross section in Figure 2-1. The longitudinal direction is the traffic travelling direction. The thesis is limited to the asphalt concrete materials defined as a surface layer (surf), binding layer (bin) and base layer (base).

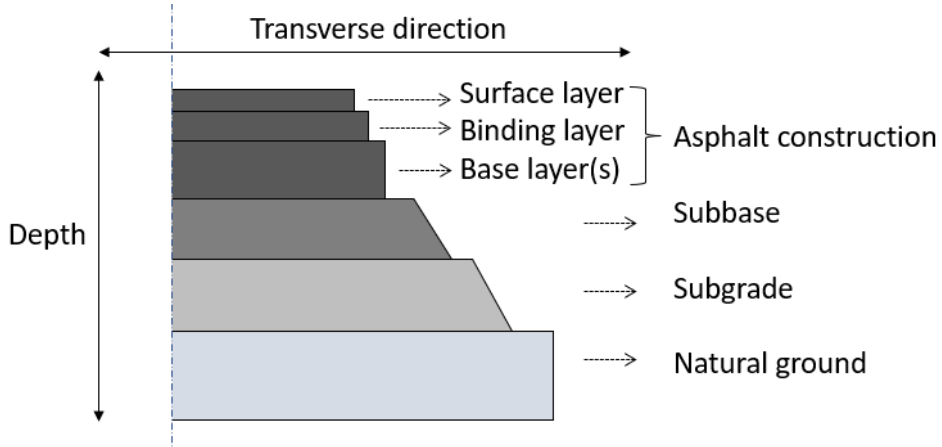


Figure 2-1: Standard Dutch pavement construction design

2.1. Standardised input in pavement design

This chapter introduces asphalt concrete as the combination of its constituents as standardised in the Eurocodes (European Committee for Standardization, 2016). A brief introduction of all constituents and their parts in the material design is given.

2.1.1. Asphalt mixture recipe

An asphalt concrete mixture recipe contains all the constituents with their weight percentages that need to be mixed in an asphalt plant to get a specific asphalt concrete mixture. Each asphalt concrete mixture has its functional properties found via standardised testing. (European Committee for Standardization, 2016)

A common asphalt concrete mixture exists of the following constituents: Aggregates (including sand), binder and filler. Additionally certain additives may be included.

2.1.2. Aggregate

The *aggregate* creates the 'skeleton' of the mixture which dissipates all forces from the top of the asphalt layer to the base layers underneath. In The Netherlands, aggregates always consist of broken stones. The stones are broken and fractioned according to different sieve sizes. For example: Scottish Granite 8/16 is the aggregate type of Granite sourced in Scotland that passed the 16 mm sieve but stayed in the 8 mm sieve. The sieve sizes are not definitive as some smaller stones and aggregate dust may still be included. The sieve sizes are also used in industrial sifters.

Asphalt mixtures include a range of differently fractioned aggregates to get to the desired aggregate skeleton. This range may include a larger part of sand (sieves 0/2) to create a 'sand skeleton' as is the case in mortar asphalt or may include a gap to create a 'gap-graded skeleton', which results in porous asphalt. (VBW Asfalt, Benelux Bitume, 2006)

2.1.3. Binder

This thesis is limited to *binders* consisting of *penetration bitumen* or *polymer-modified bitumen*. While the binder does not have to necessarily consist of bitumen, use of different materials is either to gain a specific colour in the top layer or is highly experimental. For clarity, bitumen refers exclusively to the *refined* version, a hydro-carbon product of the crude oil industry.

When oil is extracted from oilwells, a large variety of hydrocarbons are pumped up, the lightest of which are methane gas and the heaviest of which is the tar-like substance called bitumen. Bitumen was initially regarded as a waste product, but soon a use was found in the pavement engineering industry as a flexible pavement material.

There is an ongoing effort by oil manufacturers to increase the amount of crude oil and lessen their waste products, which partly means lessening the bitumen output. A process called *bitumen upgrading* partly changes bitumen into synthetic crude oil. (Oil Sands Magazine, 2020) For the asphalt industry these efforts mean that the binder product 'bitumen' has subtly, yet continuously, changed in functional properties. Bitumen forms 3% of European oil refineries crude oil products. So, while it is the most important asphalt concrete constituent, it is one of the least important oil refinery export products. Besides these ongoing efforts the source of crude oil used within the EU is changing. Figure 2-2 shows how the source countries have changed from 2000 to 2018 (Werkgroep Asfalt Impuls, 2021). Ongoing global conflicts like the 2022 Russian invasion of Ukraine may both change source locations further in the future and accelerate innovative investments in bitumen upgrading. (Zuk & Zuk, 2022)

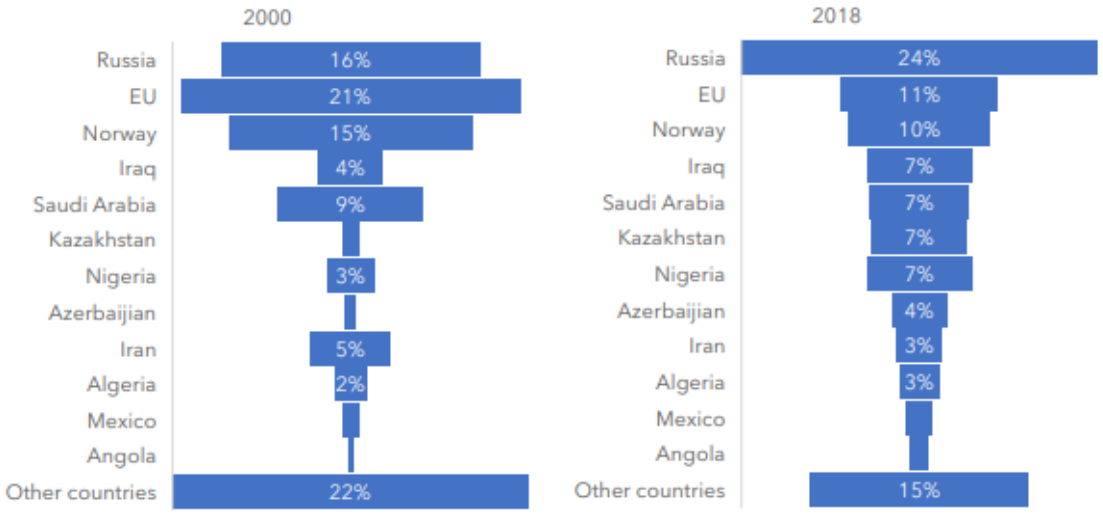


Figure 2-2: Source countries of crude oil used in Europe (workgroup Asfalt Impuls, Grip op Bitumen, CROW, 2021, Figure 7)

It is important to note that bitumen is a *viscoelastic* material, and that asphalt is a mixture of *constituents* with no chemical alterations. This means that asphalt also behaves like a viscoelastic material and is unsuited for any construction type that does not restrain its flow over time. Asphalt behaviour is generally described as a *Burgers material* where its behaviour is characterised by two elastic springs, E_1 and E_2 and two viscous flows η_1 and η_2, see Figure 2-3.

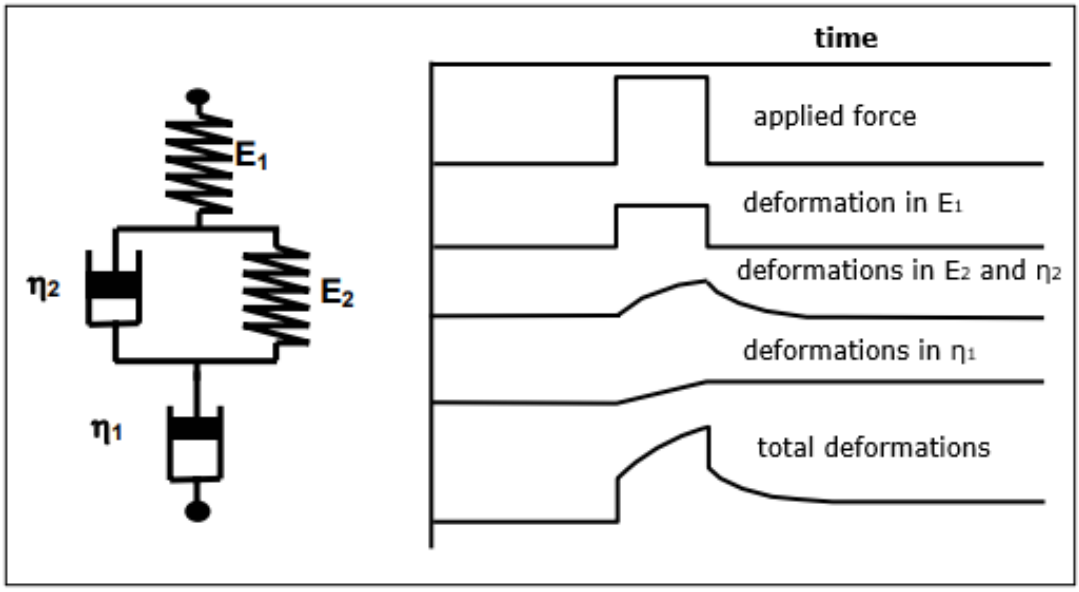


Figure 2-3: Burgers model with accompanying deformations (VBW Asfalt, Benelux Bitume, 2006)

The constituents that make up bitumen are generally classified by their solubility in four groups: *Asphaltenes* (insoluble in hydrocarbons), *saturates* (elution with n-heptane), *aromatics* (elution with toluene) and *resins* (elution with methanol). Other classification methods include solubility by *polarity* and by their *hydrodynamic volumes*. (Branthaver, et al., 1993)

In The Netherlands, standard refined bitumen is classified as *penetration grade bitumen* and is characterised by the *penetration* test. (European Committee for Standardization, 2009). An example of penetration grade bitumen is pen grade 40/60, the penetration measurement lies between 40 and 60 0.1mm. A higher pen grade in general means it is more viscous, has a lower stiffness, a higher fatigue resistance and better workability at lower working temperatures. Because the asphalt stiffness is one of the most important properties, there is a continuous search for the sweet spot between a high stiffness and a high resistance to fatigue. Besides the penetration test the *softening point test* is also needed for full characterisation, though this does not influence the penetration grade. The softening point is defined as the temperature at which bitumen can no longer support a standardised metal ball which at that point sinks into the bitumen. Unfortunately, the standardised bitumen characterisation and classification do not link directly to the bitumen functional properties. The current standardised method for bitumen characterisation has been described as 'risky for producers' and 'incomplete'. (Molenaar & Klarenaar, 2008)

Another bitumen variant is *polymer modified bitumen*, commonly referred to as PMB. (Shell Bitumen, 2015) PMB is a catch-all term for penetration bitumen modified by different types of polymers. A 2014 study investigated the optimization procedures for a standard penetration grade bitumen (70/100) using 3 different types of modifications: *Polyethylene wax* (PW), *Styrene-Butadiene-Styrene copolymers* (SBS) and *Crumb rubber* (CR). The study resulted in optimised blending charts. The modifications found were used to influence the bitumen susceptibility to weather and improve the resistance to rutting and cracking. It is important to note that rutting and cracking are similar, both originating from micro-cracks, but have completely different failure mechanisms. Rutting is a surface problem and cracking may have a variety of origins. However, if the origin of cracking is the asphalt layer itself (the fatigue failure mechanism) then it occurs from the bottom. The effectiveness of the PMB was determined using the *penetration index*, a value determined from penetration- and softening point-measurements. It has the same translation problem as regular bitumen but amplified because it treads outside the bounds of 'penetration grade bitumen'. (Munera & Ossa, 2014)

In the thesis dataset, the penetration and softening point are available. The penetration index is a method to combine the bitumen penetration and softening point into a single number to better compare different bitumen on a single scale. It is included as it may prove useful in modelling attempts. The

dimensionless aspects of PI in Equation 2-1 and the use of logarithms show that the function was (empirically) derived without regards to the equation input units. Use of this equation may be outdated and needs to be used with the proper context in mind. The penetration index is calculated as follows:

$$PI = \frac{1952 - 500\log(pen_{25}) - 20SP}{50\log(pen_{25}) - SP - 120}$$

Equation 2-1: Penetration Index (Van der Poel, 1954)

PI [-]: Penetration Index
Pen_{25} [0.1mm]: Penetration at 25 °C, (standardised test NEN-EN 12591)
SP [°C]: Softening point (standardised test NEN-EN 12591)

2.1.4. Filler

Filler in an asphalt mixture has a role in the 'asphalt mortar'. Together with the sand and bitumen it forms the paste that holds the aggregate together. Filler is similar to aggregate but specified as 'through the 0.063µm sieve'.

Filler has the following three roles to play in asphalt mix design: it is part of the aggregate mixture, improves the workability and improves the bonding between the bitumen and the aggregate. It does this by its relatively high specific surface area, which means there is a lot of surfaces for the bitumen to attach to compared to its volume. The filler, if completely submerged, increases the total volume of the mortar, especially where there is entrained air space between the aggregate. The bitumen will first fully bond with the filler, then with the sand and lastly with the aggregate, which means that too much filler will cost bonding power to the aggregate. (VBW Asfalt, Benelux Bitume, 2006)

In the past, The Netherlands used gravel as an aggregate, which had hardly any filler on the stones. Therefore, traditionally additional manufactured filler was needed. This filler could be conventional aggregate material, pigments in coloured asphalt, or some form of bitumen-inhibitor. Nowadays limestone and lime hydrate are commonly used as filler, sometimes adjusted with various forms of fly ashes. (Van der Ven & De Jong, 2013)

Even though gravel is no longer used in pavement grade asphalt in the Netherlands, the problem with too little dust still exists because aggregate is usually imported. The handling and transportation process removes a large part of the aggregate's own dust, thus a small amount of additional filler is still added to the mixture where this may not be necessary in other countries.

In the thesis dataset the filler is described by *type* and *weight percentage in a mixture*. The type refers to whether it is made up of residual dust or which type of added filler is used (no distinction between residual dusts is made).

2.1.5. Additives

Additives in general refer to pigments and/or drip inhibitors. In the production of asphalt concrete additives are optional, the term is a bit of a catch-all for anything that does not fit the other constituent groups. Because pigments, especially red pigments, come in the form of dust these are generally counted as filler. The materials counted as filler should be inert in asphalt concrete. Drip inhibitors are added to ensure a certain and even bitumen film thickness especially throughout the transportation. (VBW Asfalt, Benelux Bitume, 2006)

The asphalt industry moves towards less greenhouse gas emissions. This created the incentive to lower the asphalt concrete mixing temperature. A 2020 study gave an overview the use of additives to lower the mixing temperature (up to slightly above 100 °C) while keeping the onsite workability. It was noted that this is an emerging technology and that, while the effects of certain additives are known, the underlying interactions and phenomena that cause these effects are not yet understood. (Caputo, Abe, Loise, & Porto, 2020)

2.1.6. Reclaimed asphalt

Reclaimed asphalt consists of aggregate and leftover bitumen. The bitumen from reclaimed asphalt is combined with the added 'fresh binder' The penetration of the mixture is calculated from the fresh binder penetration and an assumed recycled binder penetration.

Recycled asphalt is named for its source category, in the thesis database there are four distinct categories:

- Base layer recycled asphalt;
- Bin/-surf layer recycled asphalt;
- Porous recycled asphalt;
- ECO recycled asphalt.

The base-, bin-, surf layer describes the original location of the recycled asphalt in the pavement structure (shown in Figure 2-1). *Surf* stands for surface layer and is the top layer in the construction, *bin* stands for binding layer and is usually the layer right below the surface layer. The *base* layers are usually the layers between the binding layer and the subbase material. Porous recycled asphalt has a gap-graded aggregate which may prove problematic in continuous graded mixtures. Finally, there is ECO recycled asphalt. This type is sourced from tar-contaminated asphalt which has been thermally cleaned until all tar is broken down. The cleaning process may impact the material properties which may lead to undesired effects when used as recycled asphalt, but recycling this material is desirable because it gives a second life to what would otherwise be a waste material.

2.2. Standardised output for pavement design

First an overview is made as to what would constitute the functional properties of an asphalt concrete mixture. The functional specification of asphalt mixtures is standardised in the asphalt *type test*, a collection of standardised tests as set in NEN-EN 13108-20 (European Committee for Standardization, 2016)

2.2.1. Functional specification of asphalt concrete mixtures

With the introduction of European rules and regulations (the Eurocodes) in 2008, the Netherlands has chosen to follow the path of *functional specification* in asphalt concrete. This effectively means that asphalt mixtures are tested in standardised tests the results of which relate to desired functional properties during the lifetime of an asphalt concrete pavement. (Sleuer & Stigter, 2014) The functional requirements are noted in the Standard RAW Bepalingen of which periodically a new version appears (2010, 2015, 2020). Over the years the methodology has shifted somewhat while knowledge and experience in functional specification grew with usage.

Since 2018 CROW Asfalt Impuls has started, in cooperation with stakeholders (both public and private), to design and implement a systemic approach to *functional verification*.

The method of functional specification answers the following questions:

1. What are the functional properties of this asphalt mixture as tested in a laboratory?
2. Is the asphalt mixture as found in a newly constructed pavement similar enough to the lab design to be assigned the mixture design specifications?

Functional verification aims to change this to:

1. What are the functional properties of this asphalt mixture as tested in a laboratory?
2. Are the *functional properties* of the asphalt mixture as found in a newly constructed pavement comparable enough to the designed specifications?

As this review focusses on asphalt concrete specification in the lab, the functional verification approach gives important insight into what specific functional properties are most relevant in asphalt concrete mixture design. The properties in use in the current functional verification approach are listed in Table 2-1. These functional properties will be further discussed in this chapter.

Property	Functional specification method	NEN-EN norm
Mixture density	Weighing above and underwater	NEN-EN 12679-6
Mixture stiffness	4-point bending	NEN-EN 12697-26
Resistance to fatigue	4-point bending	NEN-EN 12697-24
Resistance to deformation	Cyclic compression test	NEN-EN 12697-25
Water sensitivity	Static indirect tensile testing	NEN-EN 12697-12

Table 2-1: Functional properties, their specification method and the applicable NEN-EN norms

In the standardised pavement design *healing* is an important parameter, which will therefore also be discussed. In this discussion it will become apparent why this was not considered a viable functional property for function verification. (CROW, 2012) The advised mixing temperature of an asphalt mixture is not directly a desired functional pavement property but is however directly linked to the CO_2 footprint and therefore included in this review.

2.2.2. Mixture density

During pavement construction the asphalt concrete mixture density is one of the most important measurements for its ease of use and links to other functional properties. For this reason, the mixture density is considered as an output parameter in this review. As it stands however, it is inadvisable to include the mixture density of laboratory made specimens of asphalt concrete in the output section. During specimen preparation the mixture density is partly a forced outcome. When the mixture has been heated, mixed and ready to be compacted, the compaction is fine-tuned to reach the desired mixture density.

Because the mixture stiffness is especially used during in situ work verification due to its predictive links with functional properties, using the calculated mixture density in *forward models* allows these links to be better established. Using the measured mixture density as an output parameter may not give much more information than the offset between the measured density and the calculated density. It is a subject that may be interesting when calibrating the test specimen preparation methodology, but not useful in a comprehensive asphalt concrete prediction model. An output parameter that may prove useful however is the amount of force or time needed during the compaction process. If a mixture is almost impossible to compact to its calculated density, then this may prove a problematic aspect of this mixture during construction.

Instead of the mixture density itself it is therefore advised to use the *number of gyrations* needed during the preparation of gyration specimens. Gyration specimens are compacted by a gyrating movement of the bottom plate of the mould. This movement is countable and included in the dataset. If a link can be established between the number of gyrations, the gyration tablet height and the mixture design this may prove an important step in the determination of asphalt mixture design workability during construction. The number of gyrations is however also dependant on the way the gyrator was loaded, and the thesis dataset has no standardised method for this. (De Bruin, Jabobs, & Rering, 2011)

2.2.3. Stiffness

Asphalt *stiffness* is described as a temperature dependent curve where the variable coefficients are determined in a lab environment with standardised equipment. The curve is characterised at a temperature of 20 $^{\circ}$C and a loading frequency of 8 Hz. (CROW, 2012).

The curve is described in Equation 2-2. Of the model coefficients c_1 to c_4 only c_1 is not pre-defined. The equation is empirically derived and has no meaningful way to distinguish between the units of the different coefficients. Care must be taken to use the right input units.

$$\mathrm{Ln}(E_a) = c_1 + c_2 \cdot T_a + c_3 \cdot {T_a}^2 + c_4 \cdot {T_a}^3$$

Equation 2-2: Stiffness from model coefficients (CROW, 2012)

Stiffness model coefficient	Value
c_1 [-]	Equation 2-3
c_2 [-]	-0.018400189
c_3 [-]	-0.001098345
c_4 [-]	0

Table 2-2: Stiffness model coefficients (CROW, 2012)

$$c_1 = \ln\left(E_{a,250} - \Delta E_{a,250}\right) + 0.80734$$

Equation 2-3: Stiffness model coefficient 1 (CROW, 2012)

$E_{a,250}$ [MPa] is the asphalt modulus as determined (European Committee for Standardization, 2012)

$\Delta E_{a,250}$ is a reduction in asphalt stiffness depending on spread and number of times the test is repeated. Two tables are included in the norms for when either 18 prisms are tested per test or when 6 prisms are tested per test. The test is standardised for 20˚C at an interval of 8 Hertz.

T_a [˚C] is the asphalt temperature. The standardisation includes a method to adjust the asphalt stiffness for the average vehicle speed by adjusting the asphalt temperature.

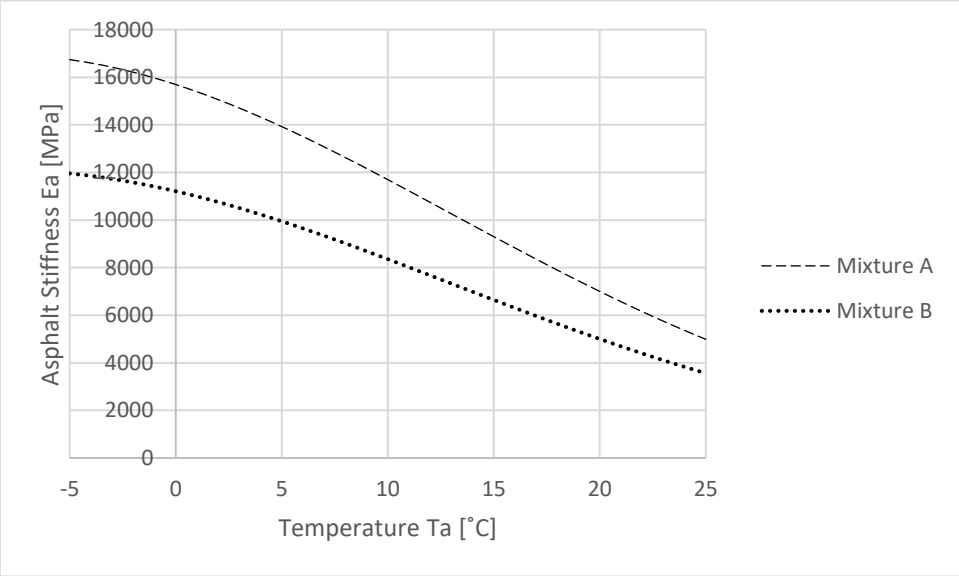


Figure 2-4: Temperature vs asphalt concrete stiffness for 2 mixture types

Figure 2-4 shows an example of the stiffness over the temperature curve. For mixture A $E_{a,250}$-$\Delta E_{a,250}$=7000 MPa and for mixture B $E_{a,250}$-$\Delta E_{a,250}$=5000 MPa. The curvature itself is pre-defined and set around T_a=20˚C because the average asphalt concrete temperature at the interface of the base layer and the subbase is approximately 20˚C. The pre-defined functions and lack of parity between the units show that these curves are based in statistical research and not grounded in physical relationships.

2.2.4. Fatigue

In asphalt concrete, *fatigue* is the main failure mechanism that occurs directly within the asphalt concrete and is not a surface layer issue. Therefore, fatigue calculations form the basis of Dutch structural pavement design. (CROW, 2012)

Fatigue occurs in the asphalt base layer (Figure 2-1). The asphalt construction in a standard design with mixed granulate has a much higher stiffness (a factor 10 to 30 higher) than the subbase underneath the asphalt. This causes the asphalt construction to work partly as a half-space and partly as a beam. The higher the difference in stiffness between the asphalt concrete and the subbase the more the asphalt concrete behaves like a continuously supported beam. A higher asphalt concrete stiffness also causes the stressed area from a passing wheel to be divided over a bigger area in the subbase and subgrade layers. Behaving in part like a beam during a wheel pass causes tensile stresses to occur in the bottom layer of the asphalt concrete and the binder, due to its viscous nature, will always deform slightly under tensile stresses.

$$\ln(N_{fat}) = c_1 + c_5\left(\ln(\varepsilon_a) + c_2\ln(E_{a,design})^2 + c_3\ln(E_{a,design}) + c_4\right)^2$$

Equation 2-4: Pavement fatigue per strain-range (CROW, 2012)

N_{fat} is the total amount of vehicles for that particular strain (ε_a) and stiffness ($E_{a,design}$).

During the design process the occurring strain is calculated in a multi-layer half space for different axle loads and vehicle wheels. This thesis is only interested in the material properties used for the design, the design itself lies outside of its scope, though a brief explanation of how it is used may benefit understanding of the properties itself. For the fatigue model coefficients c_1 to c_5 only c_4 and c_5 are not pre-defined.

Fatigue model coefficient	Value
C_1 [-]	0
c_2 [-]	0.33796
c_3 [-]	-7.3642
c_4 [-]	p
c_5 [-]	Equation 2-5

Table 2-3: Fatigue model coefficients (CROW, 2012)

The variable p [-] is defined as the slope of a linear regression between the applied force (x-axis, logarithmic) and the amount of repetitions until failure (y-axis) The design stiffness is the same as the stiffness at that location (the stiffness of the asphalt layer in which the strain occurs, usually the base layer Figure 2-1).

$$c_5 = \ln(10^6) - c_2\ln(1.2824\cdot E_{a,250})^2 - c_3\ln(1.2824\cdot E_{a,250}) - c_4\ln(\varepsilon_6)$$

Equation 2-5: Fatigue model coefficient 5 (CROW, 2012)

ε_6 [m/m] is the occurring maximum strain in a theoretical fatigue test prism that can withstand 10^6 load-cycles. In fatigue tests, a prism is considered destroyed when it is remaining stiffness is half its original stiffness (the stiffness slowly lowers during a fatigue test due to the formation of microcracks).

It may have become apparent that the fatigue behaviour used during design is described completely by the combination of material properties ε_6, p and $E_{a,250}$. These three will therefore acts as 'the material properties in the category fatigue'.

2.2.5. Resistance to permanent deformation

In asphalt concrete pavements the resistance to permanent deformation is linked to a resistance to rutting. As asphalt concrete is a viscoelastic material rutting does not stem from a destructive mechanism but from the permanent flow η_1 from the Burgers model in Figure 2-3. Figure 2-5 shows an example of excessive permanent deformation in the form of rutting.

Figure 2-5: An example of rutting (Rotated and cropped, user Burda, CC BY-SA 3.0, via Wikimedia Commons)

The tests for permanent deformation in asphalt concrete are standardised in NEN-EN 12697-25. This test is performed by cyclic loading of a specimen. Whereas during the stiffness- and fatigue tests specimens are loaded alternating on both sides of a prism both in compression and in tension, the test for permanent deformation uses a cylindrical specimen under exclusively a compressive force. The resistance to permanent deformation is then calculated by the cumulative axial strain after n load cycles (European Committee for Standardization, 2016).

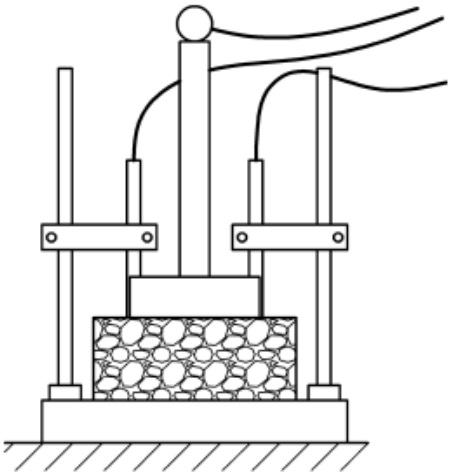


Figure 2-6: Test apparatus (NEN-EN 12697-25-2016 figure 3)

$$\varepsilon_n = \left(\frac{u_n}{t_i}\right)$$

Equation 2-6: Cumulative strain in accordance with NEN-EN 12697-25

ε_n [m/m]: Cumulative strain after n loading cycles
u_n [mm]: Cumulative permanent deformation
t_i [mm]: Initial specimen thickness

2.2.6. Water sensitivity

At the bitumen-aggregate interface three mechanisms are responsible for the formation of a bond (VBW Asfalt, Benelux Bitume, 2006):

1. The bitumen anchors itself mechanically in small crevices of the aggregate;
2. The bitumen-aggregate mixing results in a lessening of surface tension relative to a sum of its parts;
3. Dipole interactions between the negatively charged aggregate and positively charged bitumen.

As bitumen is hydrophobic, in a mixture of asphalt concrete and water the most favourable position for water molecules is within air pockets or at the bitumen-aggregate interface. Water inclusions in air pockets are not problematic but those at the bitumen-aggregate interface certainly are. These result in *stripping* which lessens the bonding power between the binder and the aggregate. To get to the binder-aggregate interface water needs to travel through the hydrophobic bitumen, either by diffusion or via microcracks. As such, the binder forms an adjustable and tuneable boundary. (Sengoz & Agar, 2007) The diffusion of water through rock (specifically the type used in asphalt concrete) is negligible compared diffusion through entrained air and along the binder. Water sensitivity is especially important in porous asphalt and in asphalt with higher entrained air percentages. The expansion of freezing water at the interface during winter is especially problematic as this strains the bitumen which at lower temperatures has an increased brittleness. (De Jong, 2009)

Figure 2-7: Typical example of frost damage in asphalt (De Jong, ZOAB vorstschade en onderhoud, VBW-Asfalt, 2009)

The determination of the water sensitivity can happen in different ways in accordance with NEN-EN 12697-12 (European Committee for Standardization, 2008). One of these methods is described to give an idea of the methodology. A water sensitivity test may follows these steps (note that this method has been simplified):

1. Prepare 2 sets (dry and wet) of asphalt concrete specimens (tablets, d=150 mm);
2. Place the wet set in a water bath for 68 to 72 hours at T=40°C (store the dry set on a flat surface at room temperature);
3. Bring the test specimen to the same temperature. Put the dry set in a plastic bag, place the dry set (in the bag) and wet set (directly) in a controlled water bath for 4 hours at T=20°C;
4. Dry the water bath specimens with a towel and immediately determine the indirect tensile strength of both sets;
5. Calculate the water sensitivity in Equation 2-7.

$$ITSR = \frac{ITS_w}{ITS_d}$$

Equation 2-7: Calculation of a measure for the asphalt concrete water sensitivity

ITSR [%]: Indirect tensile strength ratio, a measure for water sensitivity
ITS_w [kPa]: Average indirect tensile strength of the wetted specimens
ITS_d [kPa]: Average indirect tensile strength of the dry specimens

2.2.7. Healing

In asphalt concrete self-healing properties have been observed since its first use. An explanation for this lies in the viscoelastic properties of bitumen as discussed in chapter 2.1.3. A 1994 study addressed the important question 'Healing in asphalt concrete pavements: Is it real?'. The results showed that, after a fatigue-induced decrease of the stiffness modulus, the stiffness modulus could increase after a rest period (40 minutes) or after heating the asphalt concrete (to 49 °C). Remarkable is that healing was shown during conditions that may occur in situ (nieuwsredactie nu.nl, 2022).

The self-healing properties of asphalt concrete have since been defined as a process where load-induced microcracks close during resting periods. (CROW, 2012) In the Dutch pavement design process, *healing* is taken into consideration as a partial factor on the characteristic fatigue strength.

The healing factor value is limited to between 1.0 and 4.0. When modified bitumen or additives are used the healing factor equals 1.0. The healing factor may be calculated directly from the penetration value and the bitumen-percentage of the mixture, Equation 2-8. (Dienst Grote Projecten en Onderhoud, 2016)

$$SF_{healing} = \max_{4}\left(1 + 0.0000419 \cdot {V_b}^{1.06} \cdot pen^{2.45}\right)$$

Equation 2-8: Shift factor healing

$SF_{healing}$ [-]: Shift factor due to healing, applied to the characteristic fatigue strength of the asphalt.
Pen [10^{-1} mm]: penetration of the (mixed) bitumen in accordance with NEN-EN 13108-1 ANNEX A.
V_b [%m/m]: Bitumen percentage in accordance with NEN-EN 13108-7 article 5.2.3.

Equation 2-8 shows that fatigue life increases when the mixture contains more bitumen and when the bitumen used is of a softer variety, as it may more easily flow back into formed microcracks. Equation 2-8 again is an empirically derived function where no particular note was given to equating the units. Care must be taken in using the units as described.

3. Secondary literature review

This chapter addresses literature that will not define the scope directly but lies adjacent to the research. First short description of the different asphalt types will be given, after that information will be given on methods to solve for the forward and inverse problems.

3.1. Different asphalt concrete types

From a mechanical mixture design viewpoint 4 asphalt mixture types may be identified purely by their mix design (mortar vs aggregate vs voids, mortar is the mixture of binder, filler and sand):

1. *Mortar asphalt*, *sand skeleton asphalt* or *overfilled mixtures*: the aggregate 'swim' in a sea of mortar and are not interconnected. This mixture type is most common in asphalt concrete layers other than the top layer.
2. *Stone mastic asphalt* or *filled mixtures*: The aggregate form a continuous skeleton throughout the asphalt concrete, the mortar filles up the voids in the aggregate so that the voids are not interconnected. In practice, this mixture type is used as top layer asphalt due to its stable character and slight noise-reducing capacity.
3. *Porous asphalt* or *underfilled mixtures*: The aggregate form a continuous skeleton, the voids remain open and interconnected. The mortar binds aggregate to aggregate. This mixture type is used as a top layer for its noise-reducing and water bearing capabilities.
4. *Mastic asphalt:* This asphalt mixture gets its strength from the mastic (binder + filler) where both the sand and aggregate do not interconnect. This is mainly used for its watertight properties. Mastic asphalt is left outside of the scope of this thesis.

Only asphalt of type 1 has fatigue tests and extended stiffness tests as a part of their type test, this is because this is in general the only asphalt type used as the *base layer* and *bin layer* (See Figure 2-1) while the other types are used as *surf layers* or have a different function entirely. Fatigue in an asphalt pavement occurs in the bottom of the asphalt construction at the interface with the subbase where tensile forces occur. (VBW Asfalt, Benelux Bitume, 2006)

3.2. Forward Modelling

The Forward Models discussed in this chapter deal exclusively with the modelling of the functional properties of asphalt concrete from the constituents and/or mixture design (Figure 1-1). In general, research has been handled per functional property. A complete model for all interlinked functional properties of asphalt concrete has not been attempted, though some researchers have tried to combine different properties. Regarding these existing models, the reviewed research has the following problems:

1. Models do not translate well between different asphalt testing datasets;
2. Models do not translate well between different asphalt types;
3. Most asphalt literature focusses solely on predicting the asphalt stiffness.

3.2.1. Early modelling attempts (pre-2010)

Droogers researched the prediction of asphalt concrete stiffness noted the accuracy of 8 different models (the pre 2010 models) and tested these against a database consisting of 7 different Dutch asphalt mixtures. The tested models had all been verified during their creation against local asphalt mixtures, which could deviate from Dutch mixtures. It concluded that some models should be avoided due to their inaccuracy and inapplicability, while other models showed great promise (accuracy ±10%). It is noted that this predictive model used linear regression analyses (Jacobs model). (Droogers, 2018)

3.2.2. Modern models

One of the models from Droogers for the prediction of asphalt concrete stiffness was later adjusted by the author for the Dutch database that was used to verify the models. This became the adjusted 'Droogers

Model' based on linear regression analyses. Four different sets of predictive parameters (input) were tested to obtain the asphalt concrete stiffness (output). (Droogers, 2018)
The three volume fractions in Table 3-1 together sum to 100%.

Quantity	Unit	Description
$S_{bit,blend}$	MPa	Stiffness of the bitumen blend
V_a	m/m%	Volume fraction Air
V_g	m/m%	Volume fraction Aggregate
V_b	m/m%	Volume fraction Binder
Pen_{mix}	0.1·mm	Penetration value of the binder mix
T_{R+B}	°C	Binder softening temperature
C_u	-	Coefficient of uniformity
S_{mix}	MPa	Output: Asphalt concrete stiffness

Table 3-1: Droogers Model parameters

Of the four different sets of tested input parameters 3 parameters were found to be predictive and were used in the 'Droogers Model': $S_{bit,blend}$, V_a and V_b.

$$S_{mix} = 24.131 \cdot S_{bit,blend} - 1113.6 \cdot V_a - 826.35 \cdot V_b + 22062$$

Equation 3-1: Droogers model

In a database containing only penetration bitumen (no polymer binder modifications), a verification of this model led to R^2=0.85. The Droogers Model is not compatible with polymer modified bitumen, only mixtures with penetration grade bitumen. After fitting the model to the PMB-modified mixture, it resulted in the stiffness of the bitumen playing a more significant role (times 114.23 instead of times 24.131). The model refitted to the PMB-modified dataset resulted in a poor correlation, possibly because the dataset size filtered for PMB-modified bitumen was too small. Droogers concluded that more research was needed in this specific field.

A 2019 study by Martini successfully used *machine learning algorithms*, specifically the *gradient boosted decision tree machine learning model CatBoost*, to predict multiple functional properties of asphalt concrete with standardised lab input parameters. The two types of algorithms were compared: *Multiple Linear Regression* and *Gradient Boosted*. (Martini, 2019)

The multiple linear regression algorithm is comparable to the work of Droogers and led to comparable results, while the gradient boosted algorithm significantly outperforms the use of multiple linear regression. For the asphalt concrete stiffness R_{MLR}^2=0.79 while R_{GB}^2=0.97. This research was done using the NL-Lab Dataset.

Martini addressed the Jacobs and Droogers models, which both performed poorly on the NL-Lab dataset. The Jacobs Model predicted stiffness: R^2=0.55 with refitted coefficients. The Droogers Model predicted stiffness R^2=0.49 with its original coefficients and R^2=0.37 with refitted coefficients. How the author managed to decrease the accuracy of the Droogers model by refitting them is not clearly explained.

In a solution for the forwards problem Martini proved MLR-models to be inflexible when exported from the dataset used to verify them, whereas GB-models looked promising. A problem with the NL-lab Dataset was that it used input parameters which were not standardised and readily available in the PIM-environment (mixing method and compaction method). A GB-model without these parameters has also been included and it resulted in a similar fit%. The GB-model parameters using standardised quantities are summarised in Table 3-2.

Quantity	Unit	Description	Normalised importance
ρ_{mix}	Kg/m^3	Asphalt density	1
Pen_{mix}	0.1mm	Penetration value of the binder mix	3
V_f	m/m%	Volumetric parameter: Filler	4
V_b	m/m%	Volumetric parameter: Binder	2
S_{mix}	MPa	Output: Asphalt concrete stiffness	

Table 3-2: GB-model parameters

Figure 3-1 shows the difference between Multiple Linear Regression- and Gradient Boosted Algorithms. This figure is copied from figure 6.16 in (Martini, 2019) and shows the relationship between stiffness and the independent variables for MLR- and GB-algorithms. Note how the range of gradient boosted relationships is consistently smaller than that of linear regression and that the gradient boosted sets can include step functions.

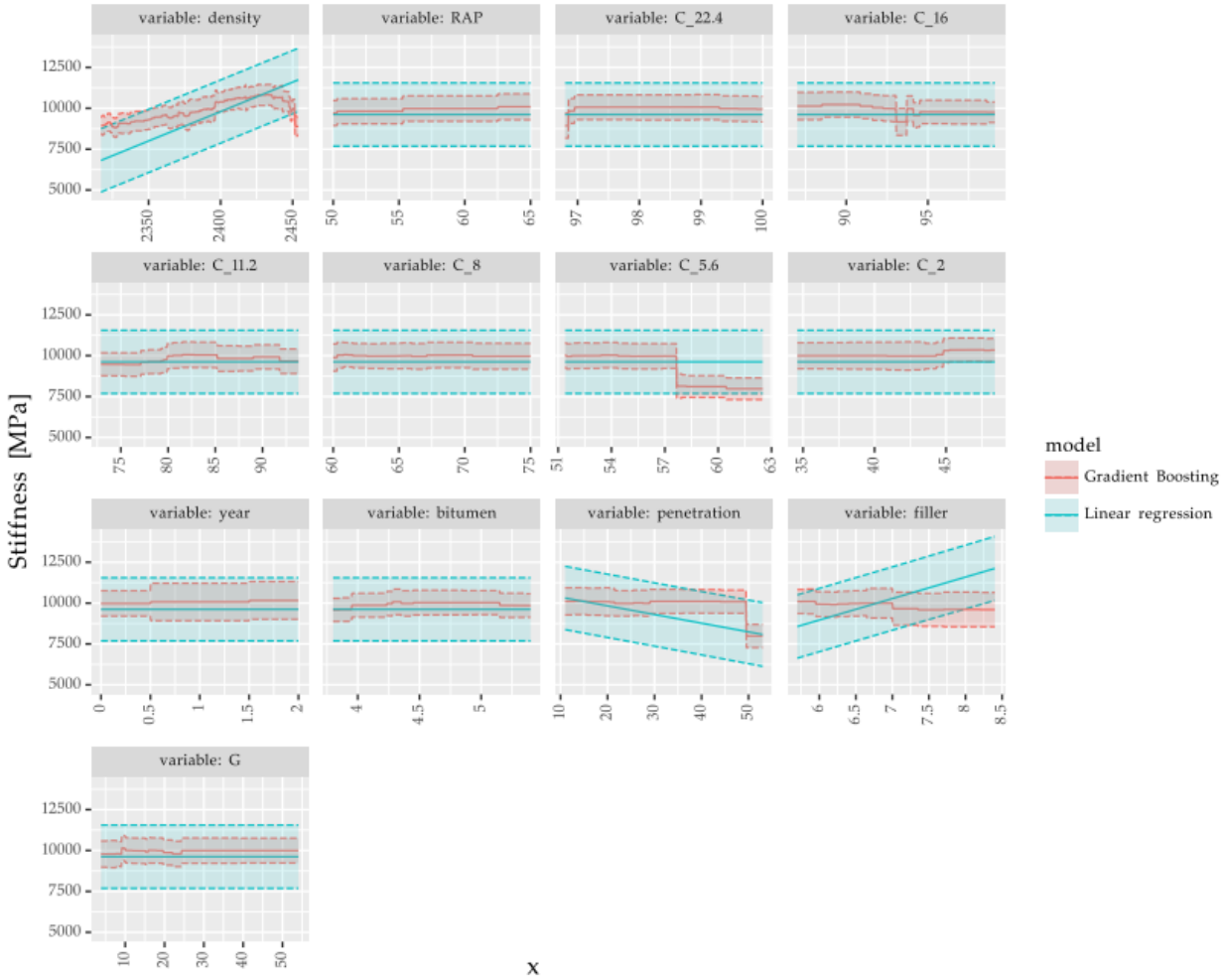


Figure 3-1: Difference between multiple linear regression and gradient boosted algorithms (Martini, 2019)

Martini also researched additional parameters: Resistance to permanent deformation and the indirect tensile strength (which is used in the water sensitivity parameter). Predicting the resistance to permanent deformation was achieved with an accuracy of R_{MLR}^2=0.67 while R_{GB}^2=0.92. The indirect tensile strength was modelled with an accuracy of R_{MLR}^2=0.69 while R_{GB}^2=0.88 (accuracies based on the complete dataset, training + testing).

3.2.3. Decision tree algorithms and Gradient boosted regression methods

The new methodologies in predicting asphalt concrete properties, shown in paragraph 3.2.2, show great promise compared to the earlier models. A brief introduction in the mechanics of specific machine learning algorithms is therefore given.

The first step in deciding on a machine learning model is based on the availability of data. In asphalt concrete property prediction, the goal is to calculate from the asphalt mix design (input) the laboratory test results (output). All models discussed in paragraph 3.2.1 and 3.2.2 are based on a dataset where input and output is available. Modern methods split the dataset in a testing and training set and use *supervised (machine learning) algorithms*. Supervised machine learning algorithms are machine learning algorithms where the input- and output data are classified by the researcher.

Supervised machine learning algorithms may be further classified in two categories, *classification models* and *regression models*. Classification models define the input in classes from which follows an output (i.e., by *if-then statements*) while regression models map the input space unto an output value. (Nateski, 2017) Asphalt concrete test results databases consist of mostly numerical input values and some categorical input values, an example from Martini's model is which compactor was used in testing. (Martini, 2019) Another example may be the use of penetration grade- or PMB-bitumen.

A flexible choice in supervised machine learning with both numerical and categorical input their and a tolerance to irrelevant or redundant data is the use of a decision trees algorithm. (Osinanwo, et al., 2017) The choice is flexible because there are different ways of building a decision tree.

Decision tree algorithms consists of nodes where each node may lead to either a different node or an output value, depending on the attribute values (input or manipulated input value). (Nateski, 2017) Figure 3-2 visualises a decision tree. Each decision node is an if-then statement which may be categorical or numeral. Each leaf node gives a predicted output value. The tree grows by changing a leaf node into a sub-tree. The flexibility in decision tree algorithms arises from the size and limits set on the sub-trees. A sub-tree with 2 decision layers (and 4 leaf nodes) needs more computing power but simultaneously adjusts the 3 decision nodes for the outcome of 4 leaf nodes, while the smaller sub-tree shown in Figure 3-2 needs less computing power per decision tree and is capable of building a bigger total decision tree in the same amount of time.

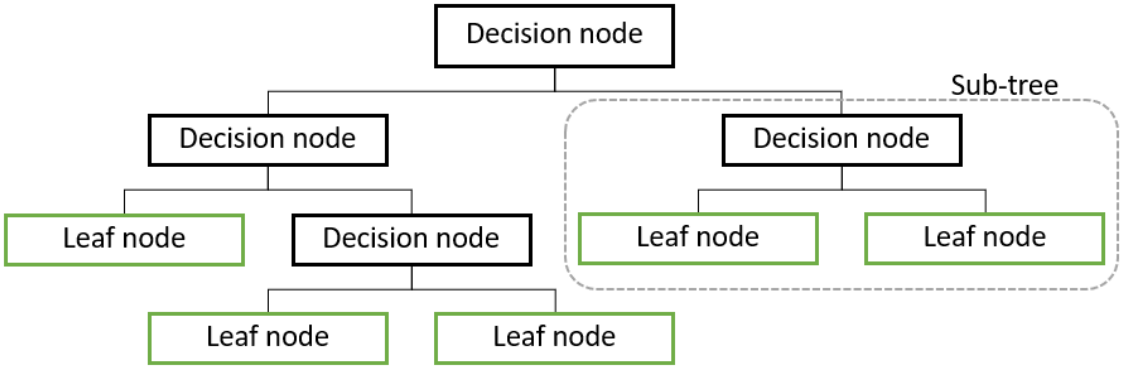


Figure 3-2: Decision tree breakdown (Navlani, 2018)

The model developed by Martini, as discussed in paragraph 3.2.2, uses the gradient boosted decision tree model CatBoost, developed by Yandex. Gradient boosting is often combined with decision tree algorithms, specifically so that categorical and numerical features may be part of the input space. (Dorogush, Ershov, & Gulin, 2017).

Gradient boosting uses a *learning rate* factor where the step taken in adjusting the leaf nodes in each new node is adjusted by the learning rate. Figure 3-3 gives a visual representation of the idea of fast learners vs slow learners. In general, a large amount of small steps in the right direction is more accurate than a smaller amount of bigger steps. Large vs small learning rates both have their positives and negatives, however. While a small learning rate may be more accurate on average, there is the possibility of ending in a low accuracy local minimum. A large learning rate may get into a positive feedback loop

where each step needs to adjust for the previous step but overshoots and needs correcting in the next step. (Goodfellow, Bengio, & Courville, 2016) Gradient boosting usually outperforms decision trees and random forest algorithms.

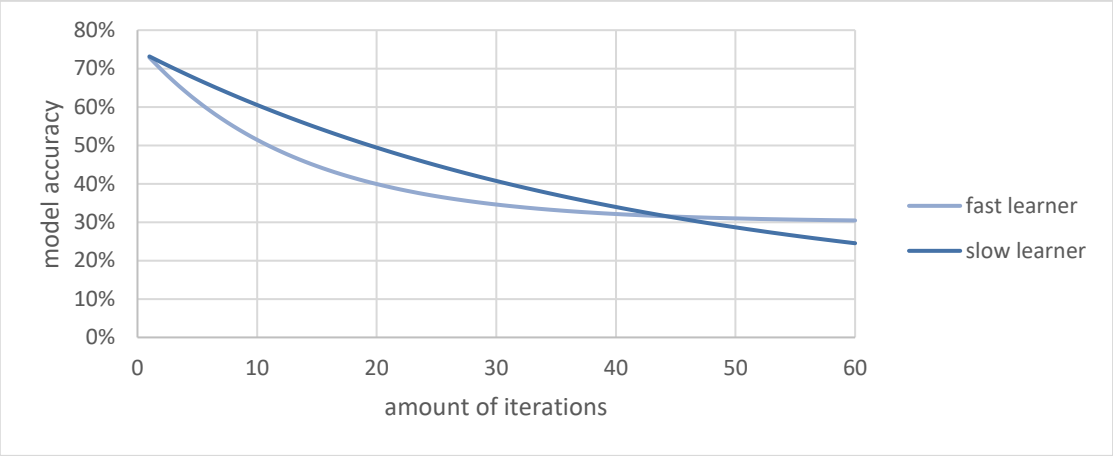


Figure 3-3: Visual representation of fast learners vs slow learners

3.3. Inverse modelling

The Inverse Models discussed in this chapter deals exclusively with the prediction of the asphalt concrete constituents from (desired) properties (Figure 1-1). As this method has not yet been used in this specific setting it the chapter deals more with methods used in similar problems and with possible roadblocks.

3.3.1. Inverse Problem Theory

Inverse problem theory concerns itself with finding a priori information from a set of measurements. An example of this is finding the location of the epicentre of a seismic event from multiple measurements of seismic waves. Each measurement gives the arrival time of the seismic wave, so a single measurement would only be able to give a sphere in which the seismic event took place. Multiple measurements with their uncertainties can narrow the answer further down depending on their relative location to the seismic event. (Tarantola, 2005) This is an example of a problem where the a priori information is unknown, the physical relationship between the event and the measurements is known and a multitude of measurements of the same type are used to give the answer. The problem description of asphalt concrete mixture design is slightly different. The physical relationship between the mixture design and the functional specifications are unknown and the functional specifications are all a different type i.e., each has their own relationship to the mix design.

In nonlinear inverse problems where no analytical expression for the forwards problem is possible and where linearization of the problem is unsuccessful a mathematical approach is impossible. In these cases, Monte Carlo methods may prove useful in finding input parameters given a certain output. A 2002 study proposes this methodology. Monte Carlo simulations are essentially brute force attacks on the problem where randomised sample of the input space are analysed to find the resulting output space. A given output can then be coupled with an (approximate) input space. (Mosegaard, 2002)

Chaotic complex systems require more attention in inverse problem theory as slight changes in the initial settings may lead to radically different outcomes. Methods to successfully find probable input parameters in chaotic systems have been established, the focus shifts from directly finding the input parameters to a *maximum likelihood parameter estimation*. A 2011 study proposed this approach in state-space systems (particularly for the nonlinear Van der Pol oscillator) noting that the approach may be useful for a broad range of estimation problems. (Pence, Fathy, & Stein, 2011)

3.3.2. Use of Machine Learning in Inverse Problem Theory

A forward model is not necessarily necessary with modern machine learning techniques. Paragraph 3.3.1 shows that if a forward model is possible, it may be reversed to approximate the input from the output. If a reversible forward model may be created, it may also be possible to find the inverse model directly via

machine learning. A 2012 study successfully used this method to the pavement layer stiffnesses, subdivided in an asphalt layer stiffness, subgrade layer stiffness and base layer stiffness from falling weight deflectometer measurements. (Gopalakrishnan, Angrawal, Ceylan, Kim, & Choudhary, 2013)

4. Dataset under study

4.1. The standardised database

4.1.1. Origin of the datasets

The dataset under study is from Ballast Nedam, an active partner in the development of the PIM software. PIM software collects data from lab technicians in a centralised database. A copy of this database is made available daily to the PIM partner companies for business analytics of which this thesis is one.

For reproducibility the exact names of the tables are given with their meaning and use:

- Bitumengrade: Bitumen type and whether the polymer modified bitumen is used.
- Mengselontwerp: Mixture design, this table contains information about the structure of the mixture such as the bitumen mixed penetration, bitumen softening point, amount of bitumen and the amount of contained air, but the shiftfactor healing which is a calculated output.
- DefinitiefMengselontwerp: Final mixture design, this table contains all tested output, results from the standardised tests.
- NominaleKorrelDiameter: The maximum aggregate size.
- BouwstofType: Constituent type, this table is used to categorize MengselontwerpBouwstof by type
- Korrelgroep: Sieve groups, this table is used to categorize MengselontwerpBouwstof by aggregate size, only aggregates are considered for this.
- MengselontwerpBouwstof: Mixture design constituents, per mixture design the amount in weight percentage of constituents is given.

These raw tables are combines and structured into 3 tables:

- Properties per mixture.
- Amount of constituents in weight percentage per mixture
- Amount of aggregate per sieve group per mixture

The three tables may then be combines into one big table with all the input- and output data per mixture. The reason 3 tables are needed as a steppingstone to the one big table is because each table has its own calculations. Properties per mixture is straightforward, no further calculations needed. Amount of constituents in weight percentage per mixture contains the sum the constituents by constituent type. If, for example, 2 different types of filler are used, which is almost always the case, then the sum of these is combined under constituent type ‘filler’. The amount of aggregate per sieve group per mixture sums the constituents by sieve group instead of constituent type. Not all mixture types need stiffness and fatigue testing, therefor the resulting dataset is filtered by mixtures that have these requirements.

4.1.2. Dataset overview

The dataset resulting from the steps in chapter 4.1.1 contains 728 mixture designs of which 214 mixtures contain data from standardised stiffness and fatigue testing. Figure 4-1 shows the distributions for all parameters in the database. The figure is meant as a quick overview, a complete overview is given in the annexes.

Asphalt Density [Kg/m^3]	Use of polymer modifications	Binder content [W%]	Binder Softening Point [°C]	Binder Penetration [0.1 mm]
Entrained Air [V%]	Filler [W%]	Sand [W%]	Mineral Aggregate Density [Kg/m^3]	Max Aggregate Size [mm]
Recycled Asphalt Base [M%]	Recycled Asphalt Bin/Surf [W%]	Recycled Asphalt ZOAB [W%]	Thermally Cleaned Sand [W%]	
Stiffness [MPa]	Stiffness Deviation [MPa]	Fatigue Parameter ε6 [m/m]	Fatigue Parameter Sxy [m/m]	Resistance to deformation
Amount of Gyrations	Shift factor Healing			

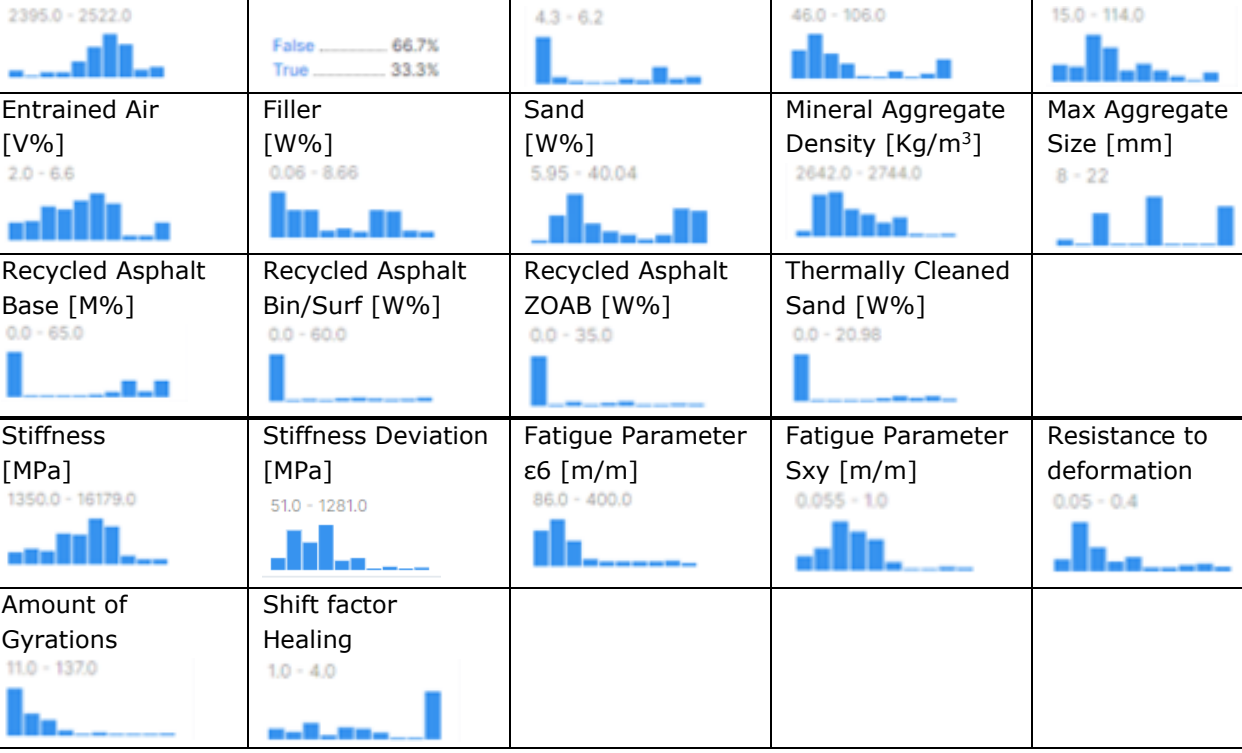


Figure 4-1: Database distributions per input- and output-parameter

4.1.3. Dataset pre-processing

To keep the work accessible the amount of dataset pre-processing is kept to a minimum. In line with the standardised division of aggregate into filler (<63 µm), sand (63 µm to 2 mm) and aggregate (>2 mm), all the aggregate types will be sorted in these three categories by their sieve sizes and/or categories in weight percentages. Because these three categories plus the bitumen weight percentage sum to a total of 100%, giving all weight percentages as input parameters is an overdetermined system. As input parameters this causes no issues, but when the solution to the inverse is solved, this does lead to a problem. Each parameter has its own model attached to it and is individually determined. This method leads to mismatches in overdetermined system. The easiest way to solve this is to leave one parameter out when the inverse problem is solved. The parameter left out is the aggregate volume percentage. Sand, filler and bitumen volume percentages are easier to adjust in a mixture, from a mix design viewpoint these values are more important.

4.2. Parameters

4.2.1. Predictive parameters (Input)

Table 4-1 lists the input parameters and their information type.

Name	Summary	Information type
Asphalt Density [kg/m^3]	Calculated mixture density in kg/m^3 as measured in the laboratory	Float
Bitumen Type	Use of polymer modified bitumen of penetration bitumen	Binary
Binder content [W%]	Total amount of binder in the mixture in mass percentage	Float
Binder softening point [°C]	Softening point of the mixed binder	Float
Binder penetration [0.1 mm]	Penetration of the mixed binder	Float
Entrained air [V%]	Amount of entrained air in the mixture in volume percentage	Float
Filler [W%]	Total amount of filler in the mixture in mass percentage	Float
Sand [W%]	Total amount of sand in mass percentage	Float
Aggregate [W%]	Total amount of aggregate, bigger than 2 mm, in mass percentage	
Max aggregate size [mm]	Maximum aggregate sieve size	Categorical
Recycled asphalt type	Source of the recycled asphalt: 'Base', 'bin/surf', 'ZOAB'	Categorical
Thermally cleaned sand [W%]	Amount of thermally cleaned recycled asphalt in mass percentage	Float

Table 4-1: Predictive parameters (input) used in the models

4.2.2. Predictable parameters (Output)

Table 4-2 contains the output parameters along with their information types.

Name	Summary	Information type
Stiffness	Stiffness of the asphalt concrete in MPa	Float
Stiffness deviation	Variation in the stiffness measurements in MPa	Float
ε_{6min}	Maximum strain at 10^6 cycles during standardised testing, a measure for the long-term low force fatigue behaviour	Float
Sxy	Variation in the fatigue measurements	Float
f_{cmax}	Resistance to permanent deformation	Float
ITSR	Indirect Tensile Strength Retained, a measure for the water sensitivity of asphalt concrete	Float
Amount of gyrations	Amount of gyrations needed in the process of creating gyration specimen, possibly a measure of workability for the asphalt	Float
SF Healing	Shiftfactor for healing, value between 1 and 4	Limited float
Mixing temperature	Advised mixing temperature in degrees Celsius	Float

Table 4-2: Output parameters used in the models

4.3. Statistical methodologies

4.3.1. Randomizing and splitting

The dataset entire dataset as described in chapter 4.2 has the row numbers randomised and is split in a training- and a testing set. The different models are then fitted to the training dataset. This is so that the model can be tested on a dataset that was not part of its own training. The training/testing ratio is 70%/30%. For randomizing and splitting the python module 'train_test_split' from 'sklearn.model_selection' is used. This module allows for the randomization process to work from a specific seed so that the process is repeatable given the same dataset.

4.3.2. Coefficient of determination R^2

In the comparison between predicted results and measured results the ratio should ideally be 1:1 meaning the predictions are equal to the measurements. An example is given in Figure 4-2 where the blue dots represent the training dataset and the red dots represent the testing dataset. The 1:1 line is also given in these types of charts.

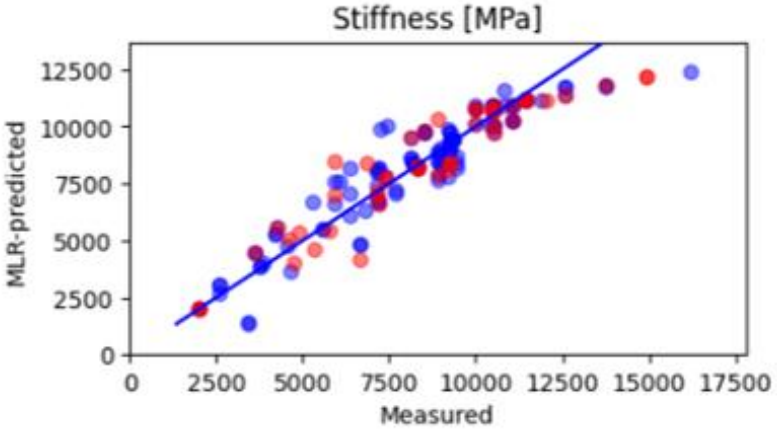


Figure 4-2: Example of a measured-predicted graph with a 1:1 line

The coefficient of determination (R^2) is a measure of how good a set of datapoints follow a linear line and can be any value under 1. It works by comparing the distance of every dot from the model to that of a horizontal line. The horizontal line in practice means that the model just gives the average value for each modelling attempt. When the value is 1 the model either works perfectly or something has gone wrong in the modelling. When the value is 0 the model has no predictive value. When $R^2 < 0$ the predictive model is worse than just taking the average value each time. The coefficient of determination is given per predictive parameter as follows:

- R^2 training: 0.90
- R^2 test: 0.86
- R^2 total: 0.89

R^2 works only as a measure for linear models and is in this work only used to compare the accuracy of predicted values to the measured values.

4.3.3. Pearson correlation coefficient for linear models (PCC)

The Pearson correlation coefficient, commonly denoted as PCC, is a value between -1 and 1 that denotes the linear correlation between two datasets. When the value is negative the correlation is inverted (a higher X means a lower Y and vice versa). If the value is 1 or -1 the linear correlation is perfect. In the linear modelling attempts PCC is used to compare the predictive capabilities of the different input parameters.

4.3.4. Feature importance for decision tree models

The feature importance for decision tree models is a measure for each parameter how important its contribution is to the predicted value. It gives a value as a percentage, the percentages of all parameters sum to 100%.

Equation 2-1 gives the function for the determination of the feature value. The feature value for each parameter is the sum of the change it is responsible for. As each leaf only deals with a single parameter the amount of change is already categorised so that it can all be summed and divided by the total to get how important the parameter is to the end result.

$$FI = \sum_{tree,leaf,sf} \left(v_1 - \frac{v_1 \cdot c_1 + v_2 \cdot c_2}{c_1 + c_2}\right)^2 c_1 + \left(v_2 - \frac{v_1 \cdot c_1 + v_2 \cdot c_2}{c_1 + c_2}\right)^2 c_2$$

Equation 4-1: Function for the feature importance of a decision tree

FI [%]: Feature importance
v_1 [-]: 1st function value of a decision node
v_2 [-]: 2nd function value of a decision node
c_1 [-]: amount of objects in the 1st node
c_2 [-]: amount of objects in the 2nd node

5. Forwards problem analysis

This chapter deals with the forward modelling, step 3 in Figure 1-1. First 2 multiple linear regression models are examined, then a decision tree machine learning method and finally a gradient boosted decision tree machine learning model is tested. Whereas the two multiple linear regression models are quite comparable, the two decision tree models vary greatly in methodology, thus the two decision tree models are handled in different chapters. In the end a comparison between the models is given.

Multiple linear regression models have historically seen a high amount of use in the prediction of asphalt concrete functional properties. The gradient boosted decision tree algorithm has recently been used to achieve good results in property prediction. A 'standard' decision tree algorithm has also been included to better understand the gradient boosted decision tree algorithm.

5.1. Multiple linear regression models

In this chapter the two algorithms *ordinary linear regression* and *Ridge* regression are discussed. Both are multiple linear regression models that are relatively comparable with their difference in the regression methodology.

5.1.1. Module and object descriptions

Two types of linear regression are used in this analysis: *Linear Regression* and *Ridge*. Both models cannot use interdependence in outputs to improve the model per output, so the model runs for each desired output independently. The Python object linear regression from the module Scikit Learn (sklearn.linear_model.LinearRegression) is the most basis form of multiple linear regression. The linear regression model makes use of unweighted ordinary least squares linear regression. Ridge (sk.learn.linear_model.Ridge) is a similar method but the square sizes in the least squares regression is squared, the result of this is that negative- and positive squares no longer cancel each other and there is a harsher bias in ridge regression towards outliers.

5.1.2. Ordinary linear regression model analysis

Figure 5-1 shows the measured output results versus the ordinary multiple linear regression predicted results. The blue dots represent training data, and the red dots represent testing data. The line follows the 1:1-line, ideally all testing- and training results follow this line. Some output parameters have strong outliers, an example of this is 'Deviation in stiffness'. Note that the outliers are on the horizontal axis thus this was an outlier in the measurements.

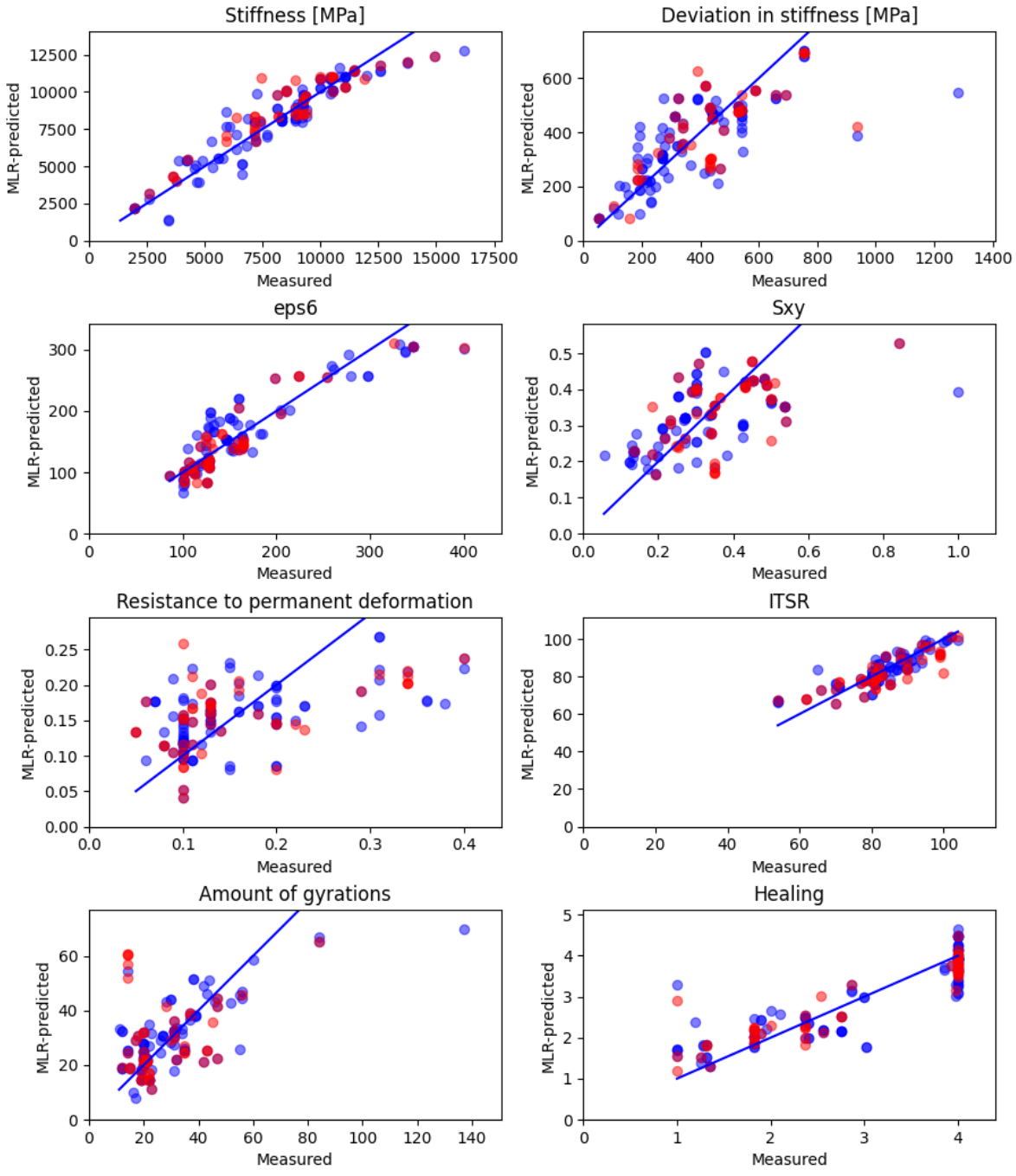


Figure 5-1: Results from the ordinary linear regression method

	Stiffness	Stiffness deviation	ε6	Sxy	f_c	ITSR	AoG	Healing
Bitumen Type	1625	-91,01	6,175	-0,093	-0,055	1,410	8,458	0,320
Asphalt Density	-3,637	0,528	0,230	0,000	0,000	0,123	-0,346	-0,012
M% Binder	-1102	-85,38	37,13	-0,080	0,032	9,65	-0,261	-0,069
Binder softening point	-67,38	0,338	2,157	0,003	0,001	0,133	-0,284	-0,020
Binder penetration	-101,3	-3,673	1,481	0,002	0,000	-0,065	-0,039	0,018
V% Air	-129,0	-18,65	-5,803	0,001	0,012	-4,061	4,613	0,239
Density mineral aggregate	29,24	1,025	-0,014	0,000	-0,001	-0,046	0,286	0,003
M% Filler	598,4	22,95	-6,500	0,001	0,015	3,099	2,249	-0,106
M% Sand	-102,6	-10,86	3,082	0,005	-0,002	-0,348	0,541	-0,029
Max aggregate size	20,76	-1,372	1,258	0,009	0,008	-0,652	-1,476	0,021
M% RA 'base'	42,74	-1,818	0,876	0,003	-0,001	0,361	0,452	-0,045
M% RA 'bin and surf'	17,38	-2,508	0,743	0,003	0,001	0,377	0,353	-0,032
M% RA 'ZOAB'	9,83	2,926	1,134	0,002	0,002	0,340	0,676	-0,011
M% Thermically cleaned sand	-76,84	8,821	-0,952	-0,002	0,000	-0,095	0,078	-0,013

Table 5-1: Coefficients of the ordinary linear regression model

$$M_{coefficients} \cdot \overline{V_{Input}} = \overline{V_{Output}}$$

Equation 5-1: Multiple linear regression model

$M_{coefficients}$: A 13x7 matrix of the coefficients in Table 5-1 (transposed)
$\overline{V_{Input}}$: A vector of length 13 of the desired input parameters
$\overline{V_{Output}}$: A vector of length 7 of the calculated output parameters

Table 5-1 gives the coefficients for all the different parameters. These coefficients can be used to recreate the model from scratch with Equation 5-1. In the input parameter column is also noted whether the coefficient works categorical [CAT] or by regression [%]. The categorical coefficients can only be compared pairwise to other categorical coefficients and regression coefficients with regression coefficients. Negative signs mean the influence of that parameter lowers the output parameter.

	$R^2_{Training}$	R^2_{Test}	R^2_{Total}
Stiffness	89%	86%	88%
Stiffness deviation	58%	55%	58%
ε6	84%	83%	84%
Sxy	35%	18%	32%
f_c	29%	28%	29%
ITSR	73%	71%	73%
Amount of gyrations	52%	-49%	30%
Healing	79%	85%	81%

Table 5-2: Accuracy of the ordinary linear regression model (see chapter 4.3.2)

Table 5-2 gives a quick overview of the results for the ordinary linear regression model. In Droogers (Droogers, 2018) literature review Stiffness was the easiest and most predictable parameter with an accuracy of R^2 up to 80%. R^2 for the testing set reaches 86% and for the training set R^2 = 89%. Overall, the result is comparable when switching the input parameters to standardised parameters.

There is simply put no decent fit for the stiffness and fatigue deviations, so this model may not be adequately used to predict the reproducibility of asphalt prisms, for which this is a measure. A more remarkable find is the results for ε6, a measure for fatigue, where the training set fits up to 84% and the

testing set to 83%. Even though fatigue (ε6) is one of the most important design parameters in pavement design, predicting this so well with such a simple model gives hope for the more sophisticated models. Both ITSR and healing also show decent results. Healing is not a lab-tested output, but the healing factor is calculated directly from the amount of binder and binder penetration (see chapter 2.2.7). That the result is calculated instead of tested may explain the good accuracy.

	Stiffness	**Stiffness deviation**	**ε6**	**Sxy**	**f_c**	**ITSR**	**AoG**	**Healing**
Bitumen Type	-45,8%	-35,4%	60,2%	8,7%	-15,4%	-15,3%	-7,1%	27,1%
Asphalt Density	27,6%	26,7%	-24,1%	38,6%	-17,8%	-50,1%	-46,8%	-23,9%
M% Binder	-51,6%	-44,3%	47,4%	-46,7%	24,0%	59,6%	51,3%	41,7%
Binder softening point	-41,2%	-23,8%	69,0%	18,7%	-6,5%	8,9%	-16,3%	-3,0%
Binder penetration	-82,6%	-62,5%	64,2%	-5,1%	14,1%	-12,5%	19,9%	74,9%
V% Air	17,1%	14,7%	-28,3%	27,5%	-8,2%	-50,5%	-16,2%	-15,7%
Density mineral aggregate	-52,9%	-42,1%	56,1%	-29,1%	8,1%	29,0%	39,4%	46,7%
M% Filler	-58,3%	-50,8%	41,7%	-17,7%	9,7%	-18,1%	25,4%	71,8%
M% Sand	-64,2%	-53,1%	47,4%	-20,9%	6,1%	-19,1%	32,8%	71,9%
Max aggregate size	46,2%	45,3%	-27,9%	46,3%	-5,5%	-34,3%	-58,8%	-48,1%
M% RA 'base'	69,8%	57,0%	-45,5%	31,5%	-26,1%	-5,7%	-36,7%	-77,8%
M% RA 'bin and surf'	-9,8%	-12,7%	-4,5%	-17,4%	30,4%	36,2%	14,5%	8,1%
M% RA 'ZOAB'	3,5%	26,5%	-4,4%	-0,7%	8,5%	13,7%	14,8%	-9,7%
M% Thermically cleaned sand	19,2%	43,2%	-19,7%	14,5%	-13,5%	-15,6%	-11,5%	-35,4%

Table 5-3: Pearson correlation coefficients for the MLR model (see chapter 4.3.3)

5.1.3. Ridge regression model analysis

Figure 5-2 shows the measured output results versus the predicted results by the ridge regression model. The blue dots represent training data, and the red dots represent testing data. The line follows the 1:1-line, ideally all testing- and training results follow this line. The outlier of 'Deviation in stiffness' is not predicted any better than in the MLR model.

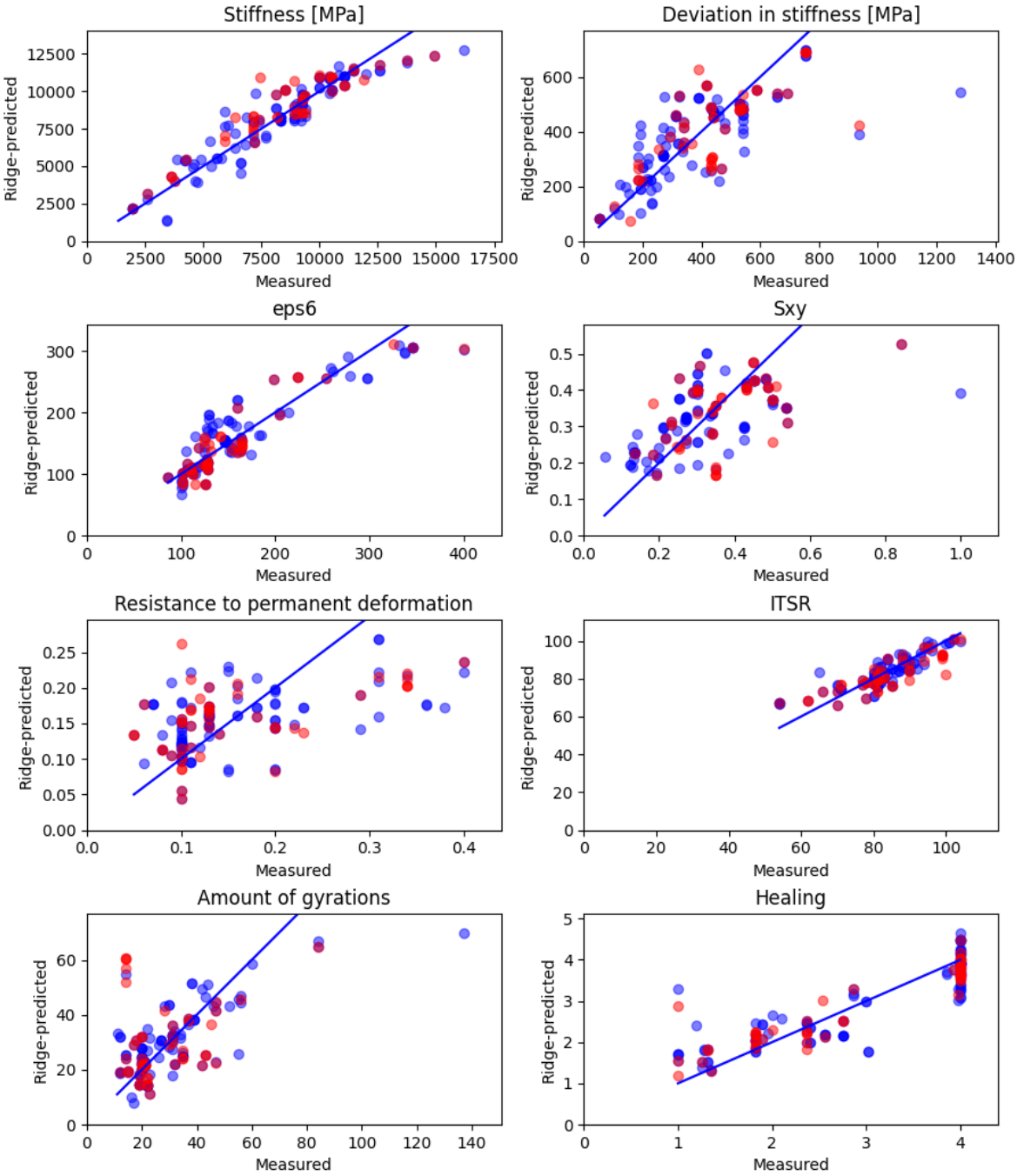


Figure 5-2: Results from the Ridge regression model

	Stiffness	Stiffness deviation	ε6	Sxy	f_c	ITSR	AoG	Healing
Bitumen Type	1505	-80,05	4,598	-0,082	-0,051	1,016	7,632	0,291
Asphalt Density	-2,311	0,665	0,171	0,000	0,000	0,107	-0,345	-0,012
M% Binder	-1061	-75,62	33,89	-0,070	0,031	8,800	-0,506	-0,072
Binder softening point	-65,55	0,162	2,182	0,003	0,001	0,139	-0,271	-0,019
Binder penetration	-100,3	-3,748	1,489	0,002	0,000	-0,063	-0,032	0,018
V% Air	-141,3	-17,42	-5,853	0,002	0,012	-4,045	4,477	0,233
Density mineral aggregate	27,81	0,888	0,043	0,000	-0,001	-0,031	0,286	0,003
M% Filler	587,6	23,57	-6,614	0,002	0,015	3,020	2,175	-0,107
M% Sand	-100,3	-10,77	3,026	0,005	-0,002	-0,361	0,549	-0,029
Max aggregate size	21,26	-1,125	1,194	0,009	0,008	-0,665	-1,486	0,021
M% RA 'base'	42,19	-1,689	0,837	0,003	-0,001	0,347	0,446	-0,045
M% RA 'bin and surf'	16,57	-2,436	0,723	0,003	0,001	0,368	0,348	-0,032
M% RA 'ZOAB'	7,454	3,118	1,105	0,002	0,002	0,330	0,659	-0,012
M% Thermically cleaned sand	-73,88	8,623	-0,936	-0,002	-0,001	-0,092	0,096	-0,013

Table 5-4: Coefficients of the Ridge linear regression model

Table 5-4 gives the coefficients for all the different parameters. These coefficients can be used to recreate the model from scratch with Equation 5-1, just like the MLR model as the difference between the MLR model and the Ridge model is the regression method.

	$R^2_{Training}$	R^2_{Test}	R^2_{Total}
Stiffness	89%	86%	88%
Stiffness deviation	58%	55%	57%
ε6	84%	83%	84%
Sxy	35%	17%	32%
f_c	29%	28%	29%
ITSR	73%	72%	73%
Amount of gyrations	52%	-49%	30%
Healing	79%	85%	81%

Table 5-5: Accuracy of the Ridge regression model (explained in chapter 4.3.2)

Table 5-5 gives a quick overview of the results for the Ridge regression model. Good results are achieved for the calculation of stiffness and ε6 (fatigue) and the shiftfactor for healing.

The stiffness result in literature the focus lies on predicting the stiffness methods where with linear regression an accuracy of up to 80% is to be expected. R^2 for the testing set reaches 86% and for the training 89%. Overall, the result is comparable even with the differences in the input parameters. There is simply put no decent fit for the stiffness and fatigue deviations, so this model may not be used to predict the reproducibility of asphalt prisms, for which this is a measure. A more remarkable find is the results for ε6, a measure for fatigue, where the training set fits up to 84% and the testing set to 83%. Even though fatigue (ε6) is one of the most important design parameters in pavement design, predicting this so well with such a simple model gives hope for the more sophisticated models. The last decent results are for ITSR and healing. Healing is not a tested output, but the healing factor is calculated directly from the amount of binder and binder penetration. A good result may be therefore be expected.

	Stiffness	Stiffness deviation	ε6	Sxy	f_c	ITSR	AoG	Healing
Bitumen Type	-45,8%	-35,4%	60,2%	8,7%	-15,4%	-15,3%	-7,1%	27,1%
Asphalt Density	27,6%	26,7%	-24,1%	38,6%	-17,8%	-50,1%	-46,8%	-23,9%
M% Binder	-51,6%	-44,3%	47,4%	-46,7%	24,0%	59,6%	51,3%	41,7%
Binder softening point	-41,2%	-23,8%	69,0%	18,7%	-6,5%	8,9%	-16,3%	-3,0%
Binder penetration	-82,6%	-62,5%	64,2%	-5,1%	14,1%	-12,5%	19,9%	74,9%
V% Air	17,1%	14,7%	-28,3%	27,5%	-8,2%	-50,5%	-16,2%	-15,7%
Density mineral aggregate	-52,9%	-42,1%	56,1%	-29,1%	8,1%	29,0%	39,4%	46,7%
M% Filler	-58,3%	-50,8%	41,7%	-17,7%	9,7%	-18,1%	25,4%	71,8%
M% Sand	-64,2%	-53,1%	47,4%	-20,9%	6,1%	-19,1%	32,8%	71,9%
Max aggregate size	46,2%	45,3%	-27,9%	46,3%	-5,5%	-34,3%	-58,8%	-48,1%
M% RA 'base'	69,8%	57,0%	-45,5%	31,5%	-26,1%	-5,7%	-36,7%	-77,8%
M% RA 'bin and surf'	-9,8%	-12,7%	-4,5%	-17,4%	30,4%	36,2%	14,5%	8,1%
M% RA 'ZOAB'	3,5%	26,5%	-4,4%	-0,7%	8,5%	13,7%	14,8%	-9,7%
M% Thermically cleaned sand	19,2%	43,2%	-19,7%	14,5%	-13,5%	-15,6%	-11,5%	-35,4%

Table 5-6: Pearson correlation coefficients for the Ridge model (see chapter 4.3.3)

5.1.4. Conclusion multiple linear regression models

	Ordinary linear regression			Ridge regression		
	$R^2_{Training}$	R^2_{Test}	R^2_{Total}	$R^2_{Training}$	R^2_{Test}	R^2_{Total}
Stiffness	89%	86%	88%	89%	86%	88%
Stiffness deviation	58%	55%	58%	58%	55%	57%
ε6	84%	83%	84%	84%	83%	84%
Sxy	35%	18%	32%	35%	17%	32%
f_c	29%	28%	29%	29%	28%	29%
ITSR	73%	71%	73%	73%	72%	73%
Amount of gyrations	52%	-49%	30%	52%	-49%	30%
Healing	79%	85%	81%	79%	85%	81%

Table 5-7: Results of the multiple linear regression methods (see chapter 4.3.2)

Table 5-7 gives the accuracy of both models. It is remarkable how similar the results are, even when no significant accuracy is found the lack of accuracy is similar in both models. Because the difference in methodology is expressed in the weight given to outliers a similar result means that outliers in the dataset are uncommon enough to not significantly change the accuracy of the models.

5.2. Decision tree machine learning models

5.2.1. Module and object description

The Python object decision tree from the module Scikit Learn (sklearn.tree.DecisionTreeRegressor) is used for the decision tree machine learning model. This model can use the interdependence of different outputs to improve the model per output, so the model needs to only run once for the entire dataset. The decision tree algorithm uses Classification And Regression Trees (CART) to construct the decision tree. With CART

binary trees are constructed where the new node is optimised to the largest gain of information. The regressor optimises for the Mean Squared Error (MSE).

5.2.2. Results with the standardised database

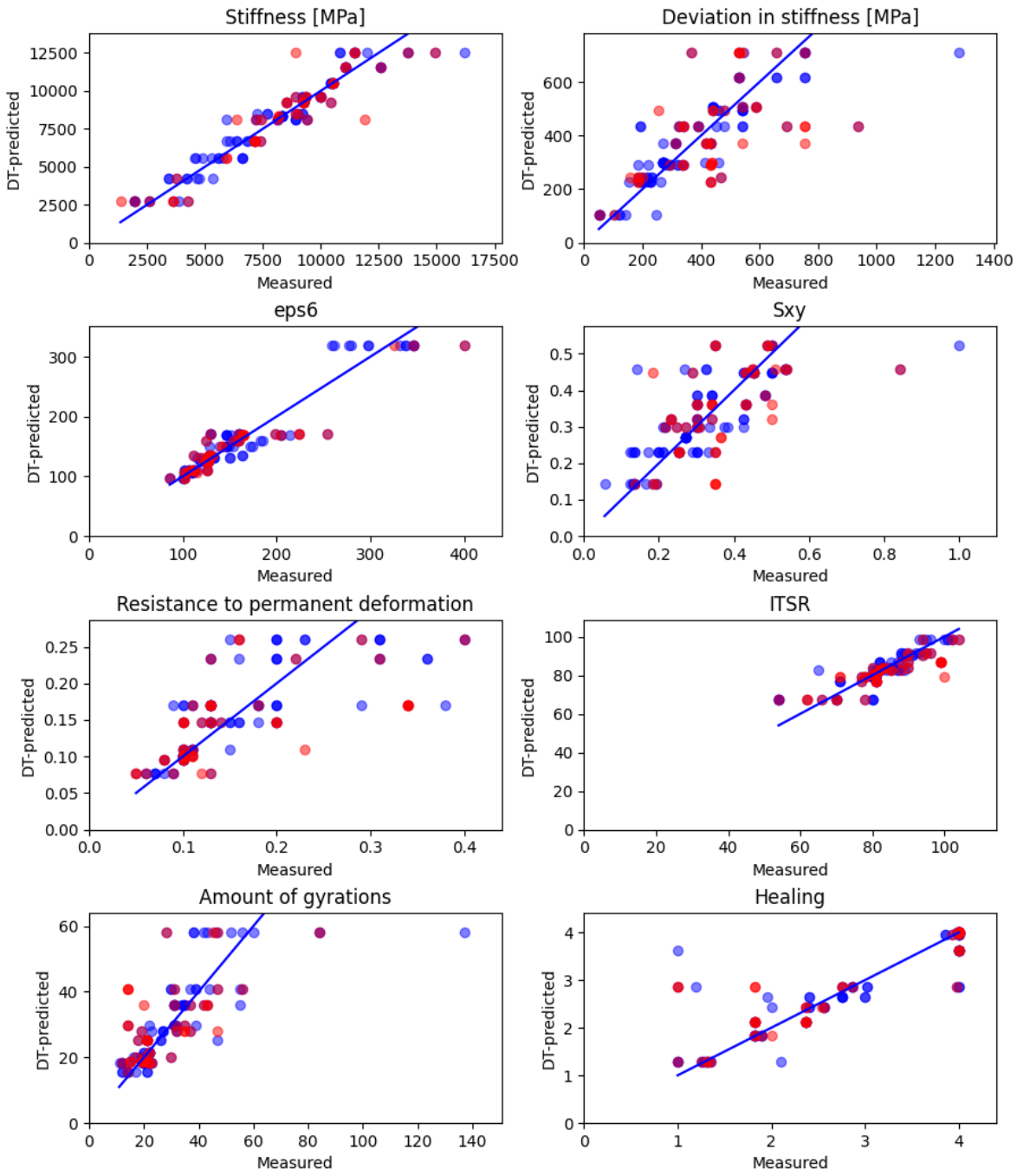


Figure 5-3: Decision tree machine learning model

Figure 5-3 shows the measured output results versus the predicted results by the decision tree model. The blue dots represent training data, and the red dots represent testing data. Table 5-8 gives an overview of

the accuracy per model. The stiffness, ε6, ITSR and Healing parameters can be found with modest to good accuracy.
The other four parameters, stiffness deviation, Sxy, resistance to permanent deformation and amount of gyrations, all show a remarkably similar pattern with decent training results and terrible testing results. The results of the testing dataset show that the good results cannot be reproduced on an unknown dataset. This is often a clear sign of overfitting, the true accuracy of the model is more in line with the testing accuracy.

	$R^2_{Training}$	R^2_{Test}	R^2_{Total}
Stiffness	92%	86%	90%
Stiffness deviation	68%	19%	56%
ε6	90%	86%	89%
Sxy	58%	20%	50%
f_c	56%	35%	49%
ITSR	75%	67%	72%
Amount of gyrations	58%	32%	52%
Healing	84%	80%	83%

Table 5-8: Accuracy of the decision tree model (see chapter 4.3.2)

5.2.3. Sensitivity analysis

	Stiffness	Stiffness deviation	ε6	Sxy	f_c	ITSR	AoG	Healing
Bitumen Type	0,0%	0,0%	0,0%	0,0%	0,0%	0,0%	3,3%	0,0%
Asphalt Density	2,1%	3,5%	0,0%	45,4%	1,5%	0,0%	0,0%	0,0%
M% Binder	9,5%	9,8%	24,9%	0,0%	0,0%	51,4%	0,0%	0,0%
Binder softening point	18,4%	1,7%	66,4%	1,3%	36,3%	4,4%	1,3%	8,2%
Binder penetration	67,6%	56,8%	0,0%	19,8%	0,0%	0,0%	0,0%	12,2%
V% Air	1,3%	0,5%	6,8%	4,9%	1,6%	13,0%	0,0%	0,0%
Density mineral aggregate	1,0%	0,6%	0,8%	18,9%	38,2%	0,9%	26,2%	0,0%
M% Filler	0,0%	27,1%	0,4%	8,4%	0,0%	0,1%	9,3%	77,8%
M% Sand	0,0%	0,0%	0,7%	0,0%	16,0%	27,8%	3,7%	0,8%
Max aggregate size	0,0%	0,0%	0,0%	1,3%	0,0%	0,0%	54,8%	0,0%
M% RA 'base'	0,0%	0,0%	0,0%	0,0%	0,0%	0,0%	0,0%	0,0%
M% RA 'bin and surf'	0,0%	0,0%	0,0%	0,0%	0,0%	0,0%	0,0%	0,0%
M% RA 'ZOAB'	0,0%	0,0%	0,0%	0,0%	6,4%	0,0%	1,3%	0,0%
M% Thermically cleaned sand	0,0%	0,0%	0,0%	0,0%	0,0%	2,3%	0,0%	0,9%

Table 5-9: Feature importance for the decision tree model (see chapter 4.3.4)

Table 5-9 shows the feature importance of the different input parameters per output parameter. A lot of parameters remain at 0% which means that they have no effect on the output parameter according to the model. While this may be factual it is also possible that there are (small) effects but the decision tree algorithm cannot model it effectively. As this algorithm searches continually for the most important parameter in a new leaf, some parameter may be forever 'the second most important' and never show up.

The importance of the binder penetration on the stiffness and the binder softening point on the ε6 fatigue parameter suggests the existence of an ideal binder that is optimised for both stiffness and fatigue. This may be worth exploring. The 3 binder parameters together determine the outcome of the stiffness parameter by 95.5%. The combination filler and binder is seen as the paste that holds the aggregate together. If the percentage filler is included the ε6 fatigue parameter is predicted for 95.4% by the combination binder/filler. This can be at least partially explained as the binder material is the softest constituent and fatigue cracks goes along the aggregate, through the binder (not through the aggregate as is the case with hardened concrete).

That the mass percentage of filler is so important in the prediction of healing is remarkable as healing is the only calculated output and the mass% of sand is not one of the input parameters. The input parameters are the mass percentage of bitumen and the binder penetration.

5.3. Gradient boosted decision tree model

5.3.1. Module and object description

For the gradient boosted decision tree model Catboost Regressor (catboost.CatBoostRegressor) is used. Gradient boost varies from normal decision tree models by building an ensemble of weak decision trees with a weight factor. The weight factor forces a new ensemble to take only a small step in the right direction. Many small steps in the right direction in general outperforms regression with a smaller amount of bigger steps. Another important detail of gradient boosting is that a newly added ensemble is slightly bigger, existing of multiple branches, than a random forest approach. This allows for more interconnection between the different input parameters as it is allowed for a bigger combination of input parameters to increase the accuracy of a new ensemble.

5.3.2. Results with the standardised database

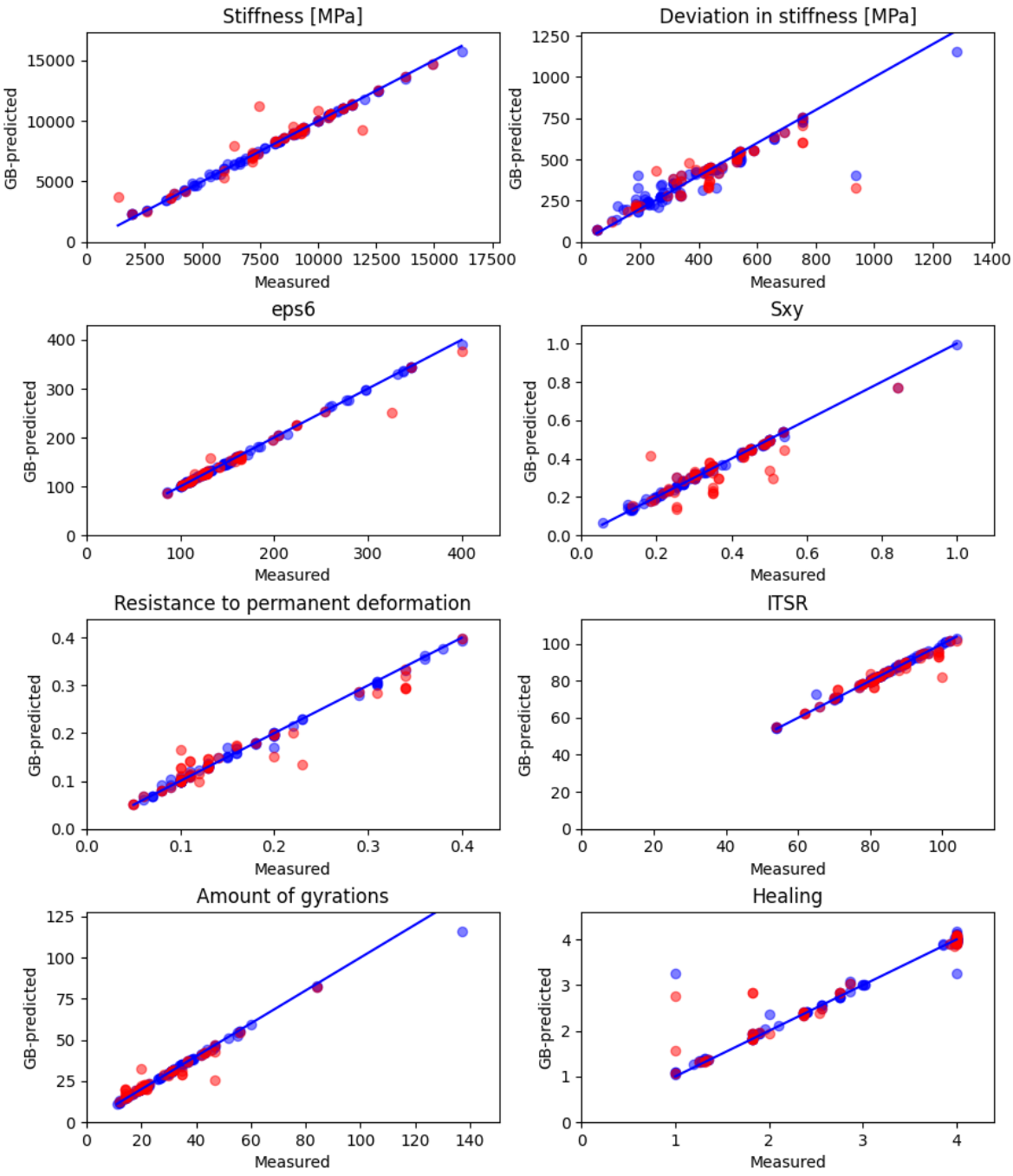


Figure 5-4: Gradient boosted decision tree models

Figure 5-4 shows the measured output results versus the predicted results by the decision tree model. The blue dots represent training data, and the red dots represent testing data. Table 5-10 gives an overview of the accuracy per model. In the training dataset all output parameters show a good accuracy, this may be because there are more input parameters than output parameters which increases the chance of overfitting. The Testing dataset gives insight into the scale of this problem. The problem of overfitting seems to be mostly a problem with the stiffness deviation and Sxy, all other output parameters achieve a test result of higher than 90%.

	$R^2_{Training}$	R^2_{Test}	R^2_{Total}
Stiffness	100%	92%	97%
Stiffness deviation	89%	67%	83%
ε6	100%	96%	99%
Sxy	99%	68%	92%
f_c	100%	93%	97%
ITSR	99%	91%	96%
Amount of gyrations	98%	90%	97%
Healing	95%	90%	94%

Table 5-10: Accuracy of the gradient boosted decision tree model (see chapter 4.3.2)

5.3.3. Sensitivity analysis

	Stiffness	Stiffness deviation	ε6	Sxy	f_c	ITSR	AoG	Healing
Bitumen Type	3,0%	1,9%	11,6%	4,2%	0,2%	1,0%	0,6%	0,6%
Asphalt Density	5,4%	6,9%	3,7%	11,3%	8,1%	9,0%	7,4%	4,1%
M% Binder	11,6%	4,4%	6,7%	8,2%	3,9%	23,0%	9,8%	2,4%
Binder softening point	21,3%	6,8%	46,3%	12,4%	17,8%	6,1%	4,8%	7,8%
Binder penetration	27,8%	18,0%	9,0%	6,0%	10,7%	4,2%	3,2%	26,9%
V% Air	4,7%	7,2%	2,0%	10,6%	12,3%	17,2%	6,4%	7,1%
Density mineral aggregate	5,9%	11,8%	5,6%	20,0%	11,9%	6,6%	6,3%	1,4%
M% Filler	1,6%	9,2%	2,4%	6,3%	11,1%	9,5%	17,8%	14,8%
M% Sand	5,5%	8,1%	3,3%	3,9%	6,3%	10,6%	8,1%	9,6%
Max aggregate size	3,7%	12,1%	3,8%	9,9%	9,3%	7,6%	27,8%	8,3%
M% RA 'base'	6,0%	6,2%	3,2%	0,4%	0,2%	1,7%	0,4%	14,3%
M% RA 'bin and surf'	0,8%	0,6%	0,6%	4,4%	5,1%	2,3%	3,0%	0,9%
M% RA 'ZOAB'	0,5%	4,0%	0,8%	0,1%	2,5%	0,3%	2,9%	0,5%
M% Thermically cleaned sand	2,1%	2,9%	1,0%	2,2%	0,6%	0,9%	1,5%	1,3%

Table 5-11: Feature importance for the gradient boosted decision tree model (see chapter 4.3.4)

Table 5-11 gives the feature importance per input parameter for all output parameters. All features have some importance and no input parameter has an importance higher than 50%, according to the GB model. The small steps method and the allowance of the 2nd and 3rd most important parameter during the creating of a new leaf allows the model to consider more nuanced effects which may explain the difference with the decision tree model's feature importance in Table 5-9.

5.4. Conclusions forward model

A summary and conclusion on the different models will be given.

5.4.1. Overview and comparison

To get an overview of the accuracy of the models R^2 of the testing datasets is compared in Table 5-12.

R^2_{test}	MLR	Ridge	DT	GBDT
Stiffness	86%	86%	86%	92%
Stiffness deviation	55%	55%	19%	67%
ε6	83%	83%	86%	96%
Sxy	18%	17%	20%	68%
f_c	28%	28%	35%	93%
ITSR	71%	72%	67%	91%
Amount of gyrations	-49%	-49%	32%	90%
Healing	85%	85%	80%	90%

Table 5-12: R^2 for the forward models (testing datasets) (see chapter 4.3.2)

The Ridge model and MLR model are incredibly similar, only when they are both terrible predictors there is a slight difference. Further optimizing linear models does not lead to a change in accuracy. The decision tree model is for ε6 a slight improvement and for ITSR and healing a slight decrease in accuracy. Only 4 of the 8 parameters, Stiffness, ε6, ITSR and Healing are predictable above R^2=60%.

On accuracy alone it is remarkable how well the gradient boosted decision tree scores above the rest. It is the only model that can predict the resistance to permanent deformation and the amount of gyrations and it is able to do so accurately. Furthermore, it is able to predict all 8 out of 8 parameters to a degree higher than R^2_{test}=60%.

Comparing the feature importance of the Decision Tree (Table 5-9) and the Gradient Boosted Decision Tree (Table 5-11) gives further insight into what allows the Gradient Boosting to improve the result so much. The DT model adds a new leaf each cycle, splitting the training dataset with the most distinguishing single parameter. Some of the parameters have a feature importance of 0%, this means that these parameters were never the most distinguishing parameter. When compared to the Gradient Boosted model not a single parameter has a feature importance of 0%. The GB model works by adding a set of leaves each step, so the top 3 distinguishing parameters are used per step. While parameters may never be the most important, they may be the second or third most important. The DT model is not able to take second and third places into account, but the GB model is able to do so. The unexpectedly high predictability of the sand mass% on the healing factor in the DT model is no longer visible in the GB model, with the sand mass% being the 2nd most influential parameter. Though it is still unexpected that healing is not exclusively predicted by the bitumen mass% and the binder penetration, as these two are the only input values in the calculation of the healing parameter.

The gradient boosted decision tree machine learning algorithm is clearly the most accurate model and will be used for the inverse analysis.

6. Inverse problem analysis

This chapter deals with step 4 from Figure 1-1: The inverse analysis. By the end of this chapter a method is given with which the input parameters may be found based on the desired functional properties. This is done with the gradient boosting decision tree algorithm which was also used in chapter 5.3. A sensitivity analysis of the different functional properties is also included.

6.1. Input-output description

The input and output in this chapter is the reverse of chapter 0. This creates an interesting problem, in the forward model all predicted output parameters are variables with only the healing shift factor containing a step function (the variable must lie between 1 and 4). Only the decision tree model of chapter 5.2 accurately handled the step function part with even the gradient boosted decision tree model going slightly above and below the limits.

The parameters 'Bitumen type' and arguably 'maximum aggregate size' are not variable but categorical. Bitumen type is an influential binary factor where it is either polymer modified bitumen or penetration bitumen. Maximum aggregate size is arguably categorical because the values represent actual maximum sizes which could theoretically lay on a spectrum. The argument for a categorical approach of the maximum aggregate size is that, even though aggregates come in all shapes and sizes, it actually describes the sieve through which aggregates need to go through, with passing aggregates going on to be used in the mixture. The sieves come in sized categories and no continuous spectrum is available. The algorithm advising the use of a sieve size other than the standardised sieve sizes would be undesirable as no sieve sizes would be available for production. This is why the maximum aggregate size is kept categorical.

6.2. Gradient boosted decision tree for the inverse model

6.2.1. Module and object description

The Catboost Regressor machine learning algorithm used in chapter 5.3 outperformed all other algorithms and is therefore also used for the inverse problem.

For the gradient boosted decision tree model Catboost Regressor (catboost.CatBoostRegressor) is used. Gradient boost varies from normal decision tree models by building an ensemble of weak decision trees with a weight factor. The weight factor forces a new ensemble to take only a small step in the right direction. Many small steps in the right direction in general outperforms regression with a smaller amount of bigger steps. Another important detail of gradient boosting is that a newly added ensemble is slightly bigger, existing of multiple branches, than a random forest approach. This allows for more interconnection between the different input parameters as it is allowed for a bigger combination of input parameters to increase the accuracy of a new ensemble.

6.2.2. Results with the standardised database

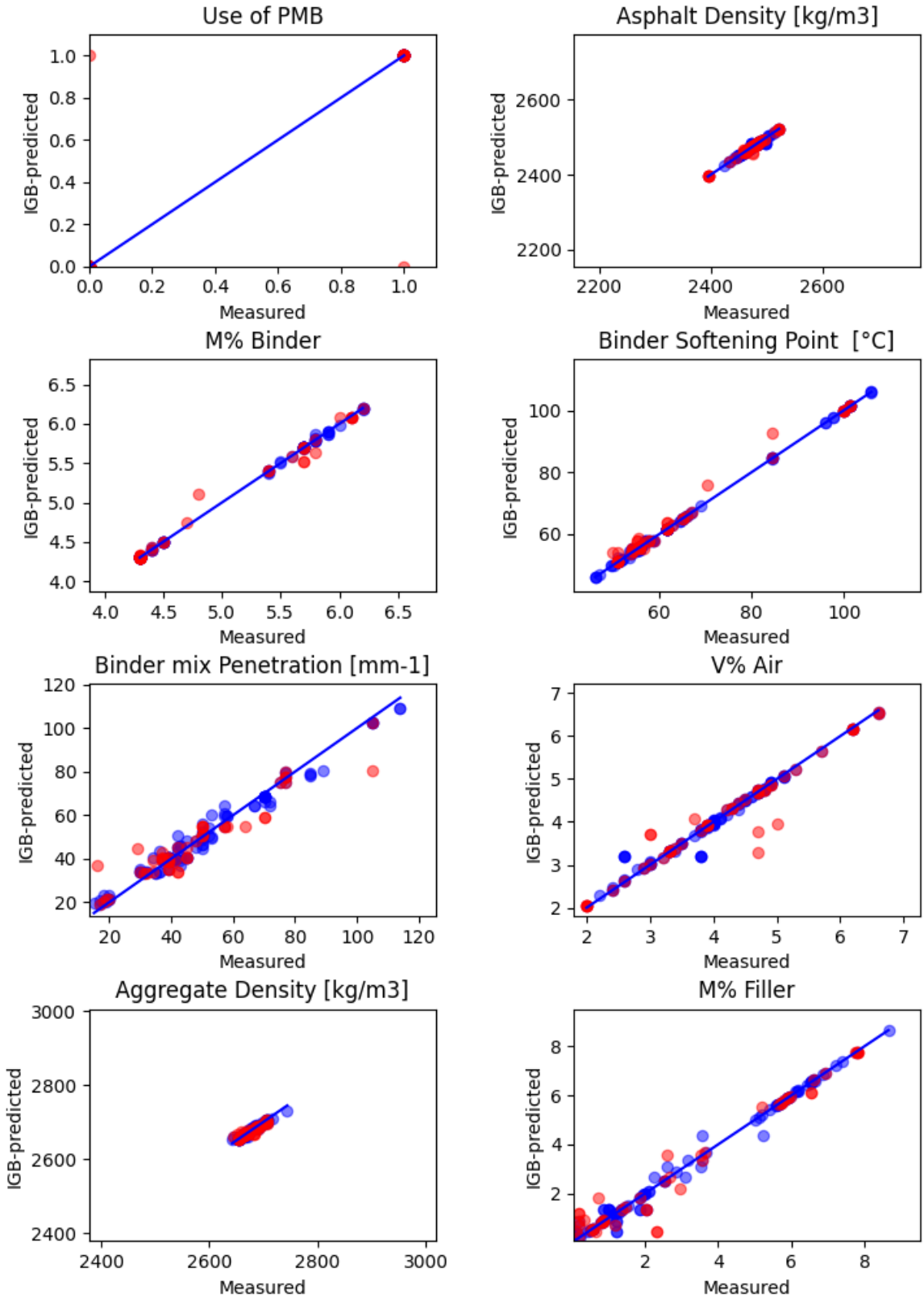


Figure 6-1: Inverse gradient boosted decision tree results (1/2)

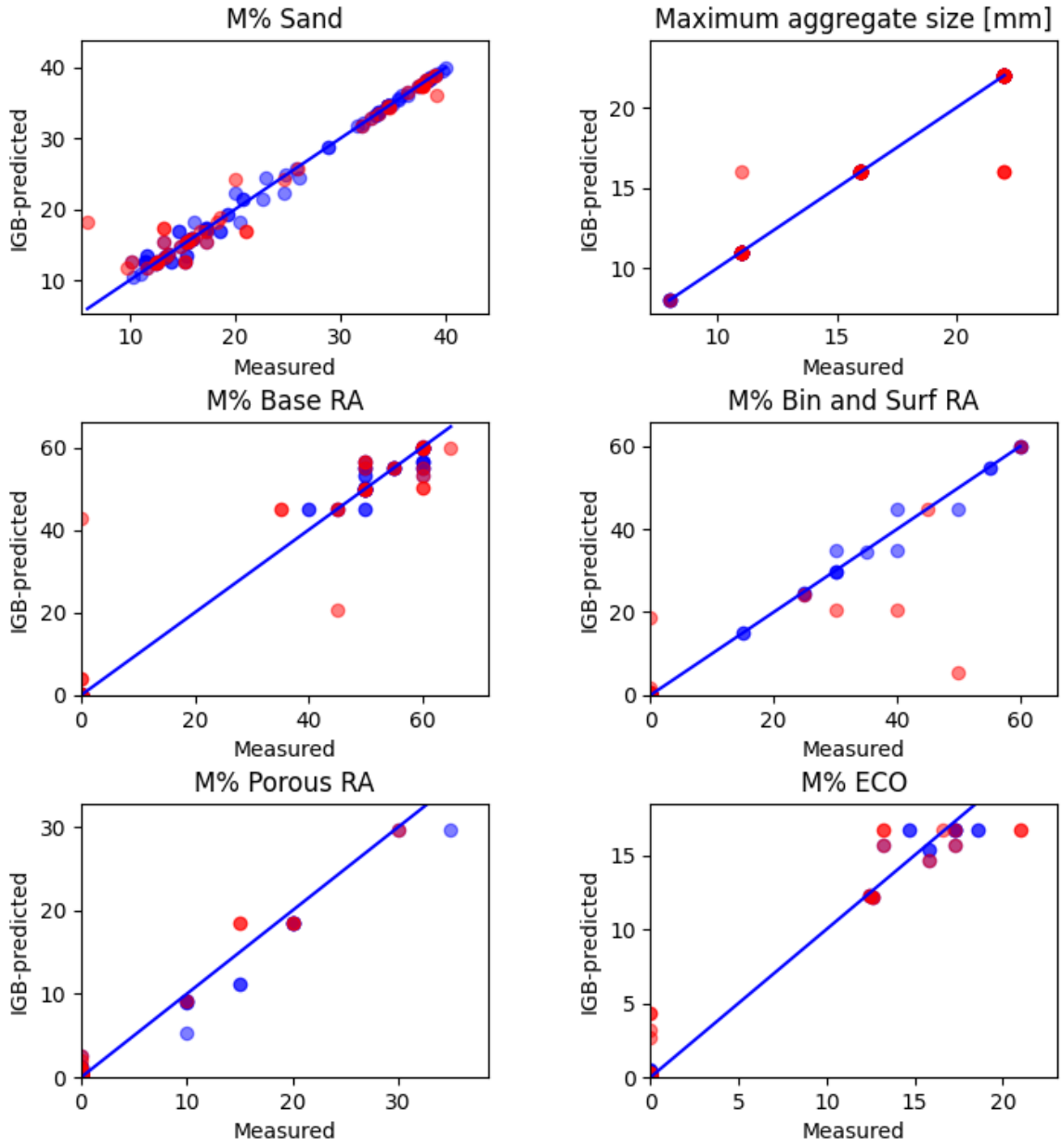


Figure 6-2: Inverse gradient boosted decision tree model results (2/2)

Figure 6-1 and Figure 6-2 show, just like in chapter 0, measured parameters against their predicted counterparts. All dots would ideally lie on the 1:1-line (meaning the prediction algorithm works perfectly). Blue and red again specify the training- and testing datasets respectively. Because there are more input parameters than functional parameters these figures consist of 14 parameters.

Table 6-1 gives the exact accuracies belonging to the figures. Only the mass percentage Bin and Surf recycled asphalt has an R^2 test below 80%. The gradient boosted machine learning model works in line with the forward model. It is notable that the binder, filler and sand mass percentages all have R^2 test accuracies ≥95%, these parameters have in the past formed the basis of an asphalt mixture design methodologies. That some of the different recycled asphalt content mass percentages are harder to predict may be because these parameters are not used in all mixtures (resulting in a lower amount of training- and testing sets if you do not count 0 M%). These also all serve a similar role in asphalt concrete which may make it hard for an algorithm to differentiate between the four. That the algorithm scores R^2_{Test}=68% for M% Bin and Surf RA is an oddity especially. The other three still score good and also show that, even though their role is similar, the algorithm can find a distinction between them based on their functional properties.

	$R^2_{Training}$	R^2_{Test}	R^2_{Total}
Bitumen Type	100%	81%	94%
Asphalt Density	97%	98%	98%
M% Binder	100%	99%	100%
Binder Temperature	100%	99%	100%
Binder mix Penetration	97%	89%	95%
V% Air	97%	93%	96%
Aggregate Density	96%	87%	94%
M% Filler	99%	96%	98%
M% Sand	99%	95%	98%
Maximum Aggregate size	100%	94%	98%
M% Base RA	99%	91%	97%
M% Bin and Surf RA	100%	68%	91%
M% Porous RA	99%	97%	98%
M% ECO	99%	95%	98%

Table 6-1: Accuracy of the Inverse gradient boosted decision tree model (see chapter 4.3.2)

6.2.3. Sensitivity analysis

	Bitumen Type	**Asphalt Density**	**M% Binder**	**Binder softening point**	**Binder penetration**	**V% Air**	**Aggregate density**
Stiffness	9,7%	9,5%	20,1%	13,1%	21,4%	8,2%	18,1%
Stiffness deviation	7,5%	8,0%	1,9%	5,2%	15,2%	13,6%	9,4%
ε6	35,9%	13,7%	4,0%	31,2%	10,8%	18,6%	11,6%
Sxy	22,4%	15,4%	18,9%	17,8%	5,1%	10,1%	17,1%
f_c	4,7%	7,0%	3,6%	6,8%	8,0%	9,5%	9,0%
ITSR	5,4%	27,4%	40,2%	4,1%	6,6%	22,3%	16,2%
AoG	8,4%	11,5%	6,7%	5,3%	6,0%	7,0%	8,1%
Healing	5,9%	7,4%	4,5%	16,6%	27,0%	10,7%	10,4%

Table 6-2: Feature importance for the inverse gradient boosted decision tree model (see chapter 4.3.4) 1/2

	M% Filler	**M% Sand**	**Max aggregate size**	**M% RA 'base'**	**M% RA 'bin and surf'**	**M% RA 'ZOAB'**	**M% 'ECO'**
Stiffness	9,3%	6,7%	9,2%	16,4%	9,5%	5,9%	9,5%
Stiffness deviation	5,7%	7,0%	8,4%	4,3%	6,8%	38,7%	12,2%
ε6	8,7%	8,2%	11,9%	6,4%	7,0%	7,9%	39,5%
Sxy	7,5%	5,5%	14,3%	4,3%	23,3%	5,3%	11,3%
f_c	7,4%	6,9%	18,4%	2,5%	9,9%	13,3%	2,6%
ITSR	5,6%	7,3%	11,9%	5,3%	22,9%	5,6%	9,8%
AoG	8,1%	6,5%	20,6%	8,6%	9,3%	18,0%	3,5%
Healing	47,8%	51,8%	5,2%	52,4%	11,2%	5,2%	11,7%

Table 6-3: Feature importance for the inverse gradient boosted decision tree model (see chapter 4.3.4) 2/2

Table 6-2 and Table 6-3 give the feature importance of each functional property on the input parameters. The first observation is that the feature importance is for a large part evenly distributed with some outliers.

With 8 parameters a completely even distribution would result in 12.5% each. Each input parameter reaches an importance of more than 12.5% for some functional property.

One that stands out is AoG (amount of gyrations), a property that was hard to predict in chapter 5 but here is an important parameter in the prediction of the binder M%. While the amount of gyrations is definitely influenced by the amount of liquid in the mixture (at the time of compaction). That its importance in predicting the amount of binder is so strong was unexpected because other parameters are measured more accurately and also strongly depend on the amount of binder. AoG may have been so important because it is the only measurement at a high temperature (at compaction temperature) when the binder properties are the most distinguishable.

Another observation is the importance of 'healing' for 5 functional properties: Binder softening point, Binder penetration, M% Filler, M% Sand and M% RA 'base'. Healing is a calculated value directly dependent on the binder type, penetration and volume percentage, so a high importance to some of these parameters is expected. That M% Sand and Filler are included is a bit odd. Perhaps these are important because their correlation to the binder M% in mix design. That the binder M% is not strongly predicted by healing was however unexpected.

6.3. Practical applications of the inverse model

This paragraph discusses the practical applications of the inverse model. The use of machine learning algorithms is flexible and versatile, but the use of a specifically trained model is limited to what it has been trained to. In the case of the model trained in chapter 6.2, it means that all functional properties are needed to get the asphalt mixture recipe. Not all functional properties are important in a practical sense. To show how this technique can be used to get practical information from a dataset, first the functional properties are further limited and then a new model is trained.

6.3.1. Further limitations on the dataset

For the further limitations Tables 81.2.16 and 81.2.8(5) of the standard RAW 2020 is referenced. The information needed for this exercise is summarised in Table 6-4. In standardised contracts in the Netherlands asphalt concrete is categorised in 4 groups by their functional properties. The name 'OL' stands for 'Onderlaag', the asphalt concrete base layer. The categories 'A, B and C' correspond to light, medium and heavy traffic. 'OL-A' is used in cycle paths while 'OL-C' is used in highways. 'IB' ('Intens belast') is for heavy and slow traffic, e.g. truck stops or roundabouts.

	OL-IB	**OL-A**	**OL-B**	**OL-C**
Definition	VA ≤ 250, v ≤ 15 km/h	VA ≤ 50	VA > 50, VA ≤ 2500	VA > 2500
V% Air	2.0 – 7.0	2.0 – 7.0	2.0 – 7.0	2.0 – 7.0
ITSR	≥ 70	≥ 70	≥ 70	≥ 70
Stiffness	7000 - 14000	4500 - 11000	5500 - 14000	7000 – 14000
f_c	≤ 0.2	≤ 1.4	≤ 0.8	≤ 0.4
ε_6	≥ 90	≥ 100	≥ 80	≥ 90

VA: Number of trucks per day per direction
v: Average speed in km/h

Table 6-4: Limitations per asphalt concrete base layer category

For this exercise the model will be trained using the 5 parameters V% Air, ITSR, Stiffness, f_c and ε6 as input. As output the model will predict mix design parameters M% binder, M% filler, M% sand and the mixed penetration grade of the bitumen.

6.3.2. Model accuracy

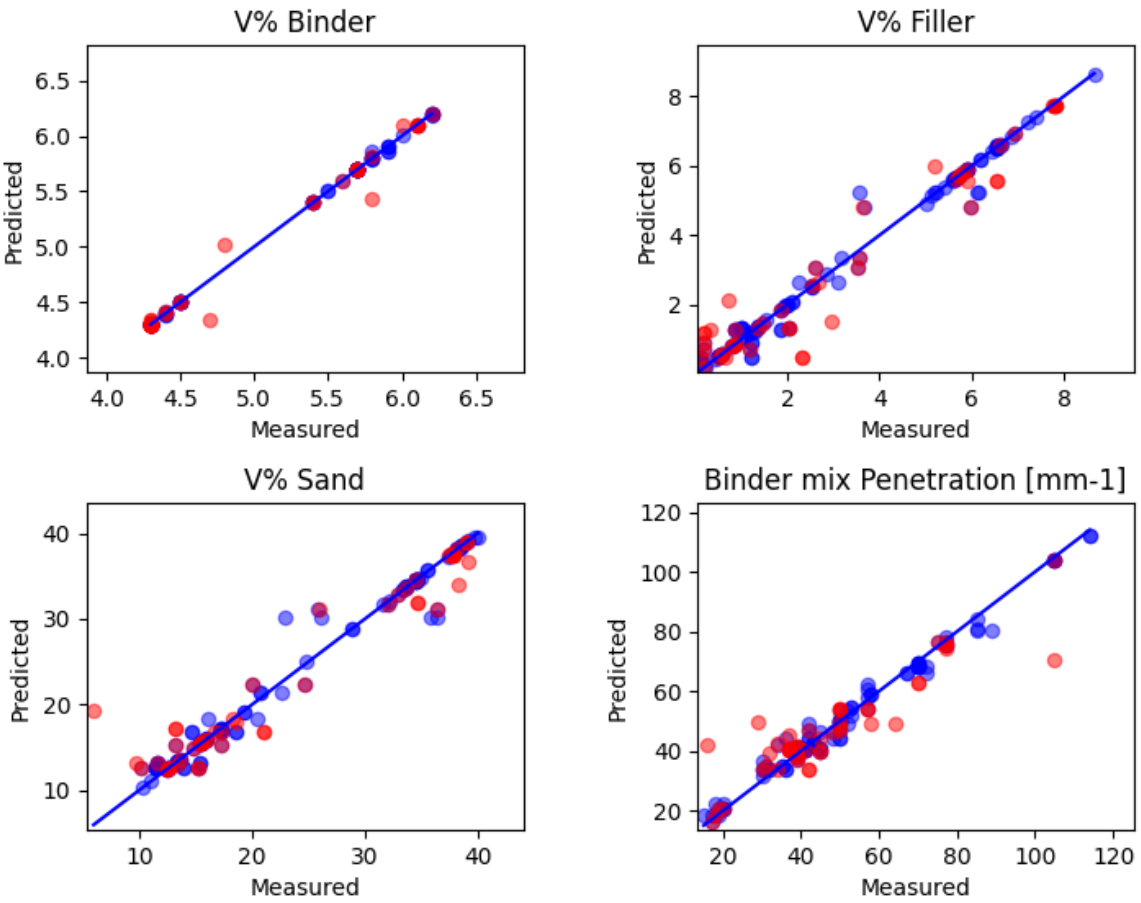


Figure 6-3: Practical model results

	$R^2_{Training}$	R^2_{Test}	R^2_{Total}
M% Binder	100%	99%	100%
M% Filler	98%	94%	97%
M% Sand	98%	93%	96%
Binder mix Penetration	98%	83%	94%

Table 6-5: Accuracy of the Practical model (see chapter 4.3.2)

Figure 6-3 and Table 6-5 show the results for the practical model in the same way as has been done for the other models. With R^2_{test}>80% the model achieves good results as well, remarkable for such a limited database.

6.3.3. Model results

The model was used to find mixtures within the parameters for asphalt category OL-C described in Table 6-6.

	OL-C
V% Air	Between 2.0 and 7.0
ITSR [%]	70
Stiffness [MPa]	Between 5500 and 14000
f_c *[m/m]*	0.8
ε_6 *[10^6 cycles]*	80

Table 6-6: Parametrisation of the practical model

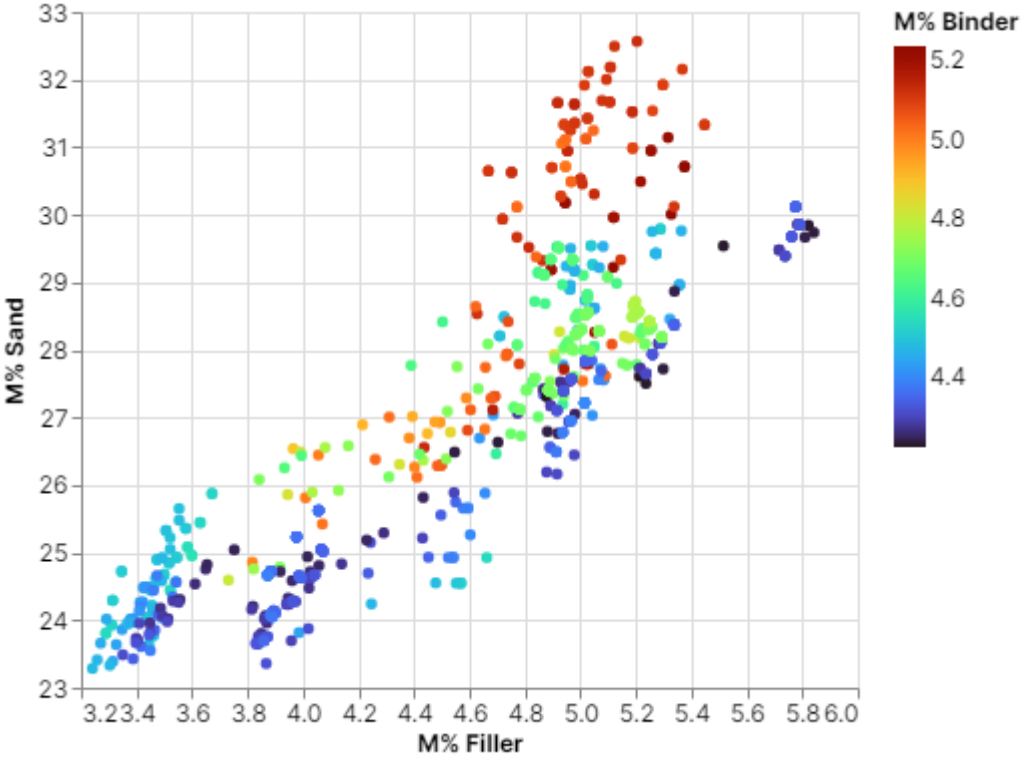


Figure 6-4: Colour chart of possible highway mixtures (Sand vs Filler vs Binder)

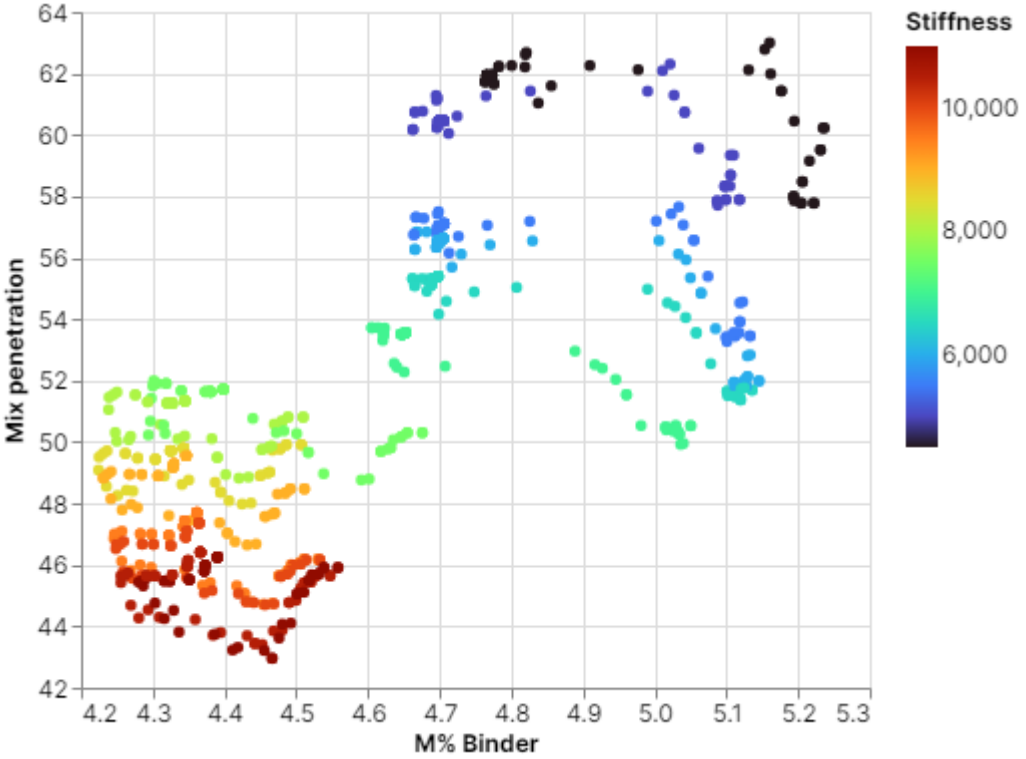


Figure 6-5: Colour chart of possible highway mixtures (Binder information vs Stiffness)

In the created mixture dataset 2 sets of data were of particular interest, shown in Figure 6-4 and Figure 6-5. These 3D (colour as the third dimension) are of particular interest because the graphs show diagonal clustering, the 3 parameters are dependent of each other.

The first combination, M% Binder, M% Filler and M% Sand show for example that when the mixtures have a low amount of filler and sand, there is also a low amount of binder. A possible (simple) explanation may be that when there is a high amount of relatively big aggregate the total amount of mortar (mix of filler, sand and binder) can be less.

The second interesting trio, M% Binder, Mix penetration and stiffness, also shows clustering around a low M% Binder. The amount of binder and its penetration grade is inverse correlated to the stiffness. This does not come as a surprise because the binder is by far the softest constituent in an asphalt mixture.

7. Conclusions

This chapter first addresses the answers to the research questions as devised in chapter 1.2, then some additional conclusions will be given. The recommendations for future research also includes an additional analysis of different machine learning algorithms that came too late to include in its proper chapter in this thesis, but was nonetheless included as it may form a basis for future research.

7.1. Answers to the Research Questions

First the sub-questions are answered, then the main research question is answered.

7.1.1. Sub-questions

The original research question that was set was:

1. *How accurate are modern asphalt mixture modelling techniques when calculating Dutch asphalt mixtures (forward problem)?*
   - *1.1. Which asphalt mixture modelling techniques are generally used?*
   - *1.2. What additional material characteristics do these modelling techniques require?*
   - *1.3. How can the output of these models be translated into OIA-parameters?*
   - *1.4. How accurate can these models predict OIA-parameters from the asphalt mixture?*

Historically functional property prediction models were either set up empirically in the form of a multiple linear regression model. Or they were set up as physical models. Of these two, the empirical models were historically more accurate. These models can predict the asphalt concrete stiffness with an accuracy of up to R^2_{Total}=79%. (No train/test split unfortunately). This accuracy has been improved upon in the dataset with R^2_{Total}=88% with a multiple linear regression model. The MLR-model uses only input parameters available in standard asphalt concrete mixture type tests. The functional properties stiffness, fatigue (ε6) and healing had good results (R^2_{test}≥75%) and ITSR had decent results (R^2_{test}≥70%). The stiffness deviation, fatigue deviation (Sxy), resistance to deformation and amount of gyrations had no significant results. Two methods for regression were tried (MLR- and Ridge-regression) but this did not result in a significant, or even noticeable, change in accuracy.

2. *How does a machine learning model stack up against conventional models (addition to the forwards problem)?*
   - *2.1. Which machine learning algorithms may be used?*
   - *2.2. Is additional material-information needed?*
   - *2.3. Can the OIA-output be simplified?*
   - *2.4. Is it needed to link certain parameters?*
   - *2.5. How accurate can the algorithm predict the mixture properties?*
   - *2.6. What improvements may be considered in the future?*
   - *2.7. How does the machine learning algorithm compare to the conventional modelling techniques?*

Two machine learning algorithms were used: A decision tree algorithm and a gradient boosted decision tree algorithm. The decision tree algorithm preformed similar to multiple linear regression. The gradient boosted decision tree algorithm proved far more accurate than the other models for all functional properties: Stiffness, fatigue (ε6), resistance to permanent deformation (f_c), water sensitivity (ITSR), amount of gyrations and healing had outstanding results (R^2_{test}≥90%). Stiffness deviation and fatigue deviation (Sxy) had decent results (R^2_{test}≥65%). No additional parameters besides those available in a standard type test were used.

3. *How accurate can inverse problem solving predict asphalt mixtures from their OIA-properties?*
   - *3.1. Which asphalt mixture modelling techniques are good enough for this method (including the machine learning algorithm)?*

3.2. What additional material characteristics do these modelling techniques require?
3.3. How can the output (asphalt mixture recipes) best be visualised?
3.4. How does the code to predict asphalt mixture recipes from their OIA-properties work?
3.5. How does the accuracy of the predictions compare to the forward problem?

The gradient boosted decision tree algorithm can predict all input parameters with an accuracy $R^2_{test} \geq 80\%$ except the mass% Bin and Surf recycled asphalt ($R^2_{test}=72\%$). There are no additional characteristics necessary for this technique besides what is available in standardised testing. The accuracies are comparable/slightly better than the gradient boosted decision tree algorithm in the forwards problem. In chapter 6.3 a practical example of this technique is shown. There the database was further reduced to 5 input parameters in line with the standardised characterising parameters for Dutch asphalt types. The output parameters were further reduced to the mass percentages binder, sand and filler, and the binder mix penetration. Within the parameters for asphalt type 'OL-C' possible asphalt mixtures were explored, shown visually in Figure 6-4 and Figure 6-5.

7.1.2. Main research question

'How can modern machine learning techniques help in asphalt concrete mixture design?'

Before asphalt concrete mixtures may be used in (Dutch) public road construction the functional properties must first be determined. The properties are determined by standardised type tests. Over the years road construction companies have created databases of asphalt concrete mixture recipes and their standardised type test results. These databases may be used to train machine learning algorithms. By inputting mixture ingredients and training for the asphalt concrete functional properties it is possible to simulate the type test. Different machine learning models have been tested, the accuracy varies per model and per functional property, shown in Table 5-12. The Gradient Boosted Decision Tree model especially achieved noteworthy results.

By inversing the training in- and output the machine learning algorithm solves for an asphalt concrete mixture design when given the desired type test results. This offers the possibility of a new non-iterative design method. The accuracy varies per constituent and is noted in Table 6-1, overall significant accuracies were found.

The methodology is proven successful in practical applications in chapter 6.3 and is ready for implementation in asphalt concrete mixture research environments. It is now possible to predict the functional properties of a new asphalt concrete mixture before the type test is completed.

7.2. Recommendations

The usual recommendations derived from the thesis work are shown in chapter 7.2.1. Nearing the end of the study, this author became aware of a method which quickly and easily assesses a large range of different machine learning algorithms from the SciKit-Learn package. The algorithm generates a list of recommendations for machine learning models per output variable. While it was too late to include this information in the thesis, it can be valuable for future research, especially for studies that do not have access to big databases like in this thesis. Therefor the author has chosen to include the outcome from the algorithm in these final recommendation chapters.

7.2.1. Recommendations for future research

The following recommendations follow directly from the research and literature study.

Only a limited number of machine learning models have been tested, broadening this scope may improve results. There is some research included in paragraph 7.2.2 meant as a steppingstone to this.

Recycled asphalt as a constituent is included in new asphalt mixtures 'as is'. As a constituent it consists of (degraded) bitumen, aggregate, sand and filler. The current categorisation of recycled asphalt is relatively broad while the low carbon footprint of recycled asphalt means new asphalt concrete mixtures

are optimising for high mass% of recycled asphalt. Improving knowledge on the functional categorisation of recycled asphalt will improve the current optimisation trend to high mass% of recycled asphalt. Standardising the functional categorisation of recycled asphalt may in the future allow for more targeted use of this constituent and an improvement in the deviations of our pavement quality.

The dataset used is only from one company and specifically for the Dutch market. This could easily be expanded to more datasets and the European market thanks to standardised testing.

Chapter 6.3 shows the use of machine learning algorithms in a practical scenario. This can easily be expanded upon. One interesting example may be the optimisation of specific constituents in a known mixture to achieve certain results. This would need a hybrid model with some input parameters and all output parameters known.

7.2.2. Recommendations for easy machine learning algorithm comparisons

The research question 'Which machine learning algorithm works best for the forwards problem' is deliberately not part of the thesis as this would broaden the scope too much. However, during the end phase of the thesis project the author became aware of an algorithm that allows for an easy comparison of all the machine learning algorithms in the SciKit-Learn package (the gradient boosted decision tree algorithm is not part of this package). This method is too easy and delivers too much insight into the differences between the different output parameters to skip. Due to time constraints the results of this analysis are not used to build upon further in this thesis. So, while the logical place for an analysis between machine learning algorithms would be before chapter 5, because this part stands alone with the specific goal to help future research it stands here as the last chapter.

	Stiffness	**ε6**	**f_c**	**ITSR**	**Healing**
GradientBoostingRegressor	91,7%	99,7%	93,4%	93,5%	92,3%
ExtraTreesRegressor	93,7%	99,6%	87,0%	93,3%	94,9%
XGBRegressor	85,7%	99,9%	94,2%	89,3%	93,1%
RandomForestRegressor	92,2%	98,3%	82,0%	85,4%	91,5%
DecisionTreeRegressor	90,9%	99,0%	80,9%	88,4%	89,4%
BaggingRegressor	92,2%	96,1%	69,2%	82,6%	93,6%
AdaBoostRegressor	89,0%	96,4%	78,0%	84,6%	84,5%
HistGradientBoostingRegressor	94,1%	92,6%	71,2%	79,0%	86,4%
LGBMRegressor	94,7%	93,2%	67,6%	76,6%	86,8%
ExtraTreeRegressor	93,8%	99,5%	93,6%	44,2%	80,0%
KNeighborsRegressor	88,0%	87,5%	36,4%	64,1%	88,3%
HuberRegressor	80,9%	79,1%	32,2%	68,7%	86,0%
TransformedTargetRegressor	83,3%	83,1%	22,3%	72,4%	85,2%
LinearRegression	83,3%	83,1%	22,3%	72,4%	85,2%
Ridge	83,5%	83,3%	21,5%	72,5%	85,2%
LassoLarsIC	83,5%	83,1%	20,7%	72,4%	84,7%
RidgeCV	83,5%	83,3%	18,5%	72,4%	85,2%
BayesianRidge	83,5%	83,2%	16,2%	71,9%	84,3%
SGDRegressor	83,6%	83,1%	10,0%	72,0%	83,7%
PoissonRegressor	81,9%	90,7%	8,2%	71,9%	79,0%
LassoCV	83,5%	82,4%	3,4%	72,5%	84,2%
LassoLarsCV	83,4%	82,1%	1,9%	72,4%	83,9%
OrthogonalMatchingPursuitCV	77,2%	83,2%	-7,4%	72,5%	83,1%
LarsCV	79,3%	82,1%	0,4%	72,4%	73,3%
ElasticNetCV	58,2%	82,7%	3,4%	72,6%	84,2%
TweedieRegressor	68,7%	70,5%	15,1%	59,0%	75,4%
GammaRegressor	67,1%	70,6%	14,0%	58,9%	75,2%
PassiveAggressiveRegressor	77,8%	81,7%	-15,2%	55,7%	66,3%

ElasticNet	74,7%	76,7%	-0,7%	60,5%	38,7%
Lasso	83,3%	83,2%	-0,7%	64,7%	-1,4%
Lars	83,3%	83,1%	-23,8%	72,4%	6,2%
OrthogonalMatchingPursuit	71,7%	41,3%	-7,4%	50,6%	62,1%
NuSVR	0,4%	9,9%	77,7%	33,2%	92,2%
LassoLars	83,4%	76,0%	-0,7%	-0,5%	-1,4%
SVR	0,4%	7,0%	-1,1%	34,2%	91,9%
RANSACRegressor	66,9%	50,8%	-124,1%	32,8%	63,3%
DummyRegressor	-0,3%	-0,5%	-0,7%	-0,5%	-1,4%
QuantileRegressor	-0,2%	-13,5%	-9,7%	-2,5%	-2,6%
GaussianProcessRegressor	35,7%	34,2%	61,8%	-403,1%	43,8%
LinearSVR	-824,2%	-39,9%	35,9%	37,4%	84,8%
MLPRegressor	-840,9%	-386,2%	50,9%	-2757,9%	89,9%
KernelRidge	-678,9%	-587,4%	-283,4%	-5752,9%	-626,5%

Table 7-1: R^2_{total} of the SK Learn regressors for the main 5 functional properties

References

Alshamsi, K. S. (2006). *Development of a mix design methodology for asphalt mixtures with analytically formulated aggregate structures.* Louisiana: Louisiana State University.

Bennert, T., Sheehy, E., Blight, R., & Gresavage, S. (2013). Enhancing the Durability of Asphalt Pavements. *Implementation of performance-based mixture design to enhance the durability of asphalt pavements in New Jersey* (pp. 50-65). Washington: Transportation Research Board.

Bijleveld, F. (2010). *Op weg naar professionalisering: Op basis van mechanische eigenschappen het bepalen van temperatuur- en tijdsvensters voor het verdichten van Nederlandse asfaltmengsels.* Enschede: Universiteit Twente.

Bonaquist, R. (2013). Enhancing the Durability of Asphalt Pavements. *Impact of mix design on asphalt pavement durability* (pp. 1-17). Washington: Transportation Research Board.

Boskalis. (2017). *Functioneel verifiëren.* Ede: CROW.

Branthaver, J., Petersen, J., Robertson, R., Duvall, J., Kim, S., Harnsberger, P., . . . Barbour, F. S. (1993). *Binder Characterization and Evaluation vol. 2. Chemistry.* Washington D.C.: Strategic Highway Research Program, National Research Council.

Brown, E. R., Khandal, P., Roberts, F. L., Kim, Y. R., Lee, D.-Y., & Kennedy, T. W. (2009). *Hot Mix Asphalt Materials, Mixture Design and Construction.* Lanham, Maryland, United States: NAPA Research and Education Foundation.

Caputo, P., Abe, A., Loise, V., & Porto, M. C. (2020). The Role of Additives in Warm Mix Asphalt Technology: An Insight into Their Mechanisms of Improving an Emerging Technology. *nanomaterials*, 1-17.

CROW. (2012). *Ontwerpinstrumentarium asfaltverhardingen (OIA).* Ede: CROW.

CROW. (2021, 08 31). *Data van het wegenbouwproces gestructureerd registreren en beheren. (consulted: 1-7-2022)*. Opgehaald van crow.nl: https://www.crow.nl/thema-s/informatiemanagement/bim-bites-best-practices-use-cases-1/bim-bites/bim-bites/2021/data-van-het-wegenbouwproces-gestructureerd-regist

CROW. (sd). *Geschiedenis*. Opgehaald van CROW (consulted: 30-7-2022): https://www.crow.nl/over-crow/crow/geschiedenis

De Bruin, B., Jabobs, M., & Rering, J. (2011). *Bepaling verwerkbaarheids- en verdichtbaarheidsparameters asfalt.* Bunnik: BAM Wegen.

De Jong, E. (2009). ZOAB vorstschade en onderhoud. *VBW-Asfalt nr.1*, 4-8.

Dienst Grote Projecten en Onderhoud. (2016). *Specificaties Ontwerp Asfaltverhardingen.* Utrecht: Rijkswaterstaat.

Dorogush, A., Ershov, D., & Gulin, A. (2017). *Catboost: gradient boosting with categorical features support.* Moskou: Yandex.

Droogers, J. (2018). *Asphalt concrete stiffness prediction based on composition and binder properties.* Delft: TUDelft.

European Committee for Standardization. (2008). *NEN-EN 12697-12: Bituminous mixtures - Test methods for hot mix asphalt - Part 12: Determination of the water sensitivity of bituminous specimens.* Brussels: CEN.

European Committee for Standardization. (2009). *NEN-EN 12591: Bitumen and bituminous binders - Specifications for paving grade bitumens.* Brussels: CEN.

European Committee for Standardization. (2012). *NEN-EN 12697-26: Bituminous mixtures - Test methods for hot mix asphalt - Part 26: Stiffness.* Brussels: CEN.

European Committee for Standardization. (2016). *NEN-EN 12697-24: Bituminous mixtures - Test methods for hot mix asphalt - Part 24: Resistance to fatigue.* Brussels: CEN.

European Committee for Standardization. (2016). *NEN-EN 12697-25: Bituminous mixtures - Test methods - Part 25: Cyclic compression test.* Brussels: CEN.

European Committee for Standardization. (2016). *NEN-EN 13108-1: Bituminous mixtures - Material specifications - Part 1: Asphalt Concrete.* Brussels: CEN.

European Committee for Standardization. (2016). *NEN-EN 13108-20: Bituminous mixtures - Material specifications - Part 20: Type Testing.* Brussels: CEN.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning.* The MIT Press.

Gopalakrishnan, K., Angrawal, A., Ceylan, H., Kim, S., & Choudhary, A. (2013). Knowledge discovery and data mining in pavement inverse analysis. *Transport*, 1-10.

Gorkem, C., & Sengoz, B. (2009). Predicting stripping and moisture induced damage of asphalt concrete prepared with polymer modified bitumen and hydrated lime. *Construction and Building Materials*, 2227-2236.

Griffioen, E. (2019, 02 22). *Welke toekomst heeft Nederland als asfaltland (consulted: 23-7-2022)*. Opgehaald van CROW: https://www.crow.nl/over-crow/nieuws/2019/februari/welke-toekomst-heeft-nederland-als-asfaltland

Jahanbakhsh, H., Karimi, M., & Nejad, F. (2020). Correlation between asphalt concrete induced healing and rheological properties of asphalt binder. *Construction and Buildings Materials*, 1-16.

Kim, Y., & Whitmoyer, S. L. (1994). Healing in Asphalt Concrete Pavements: Is it Real? *Transportation Research Record*, 89-96.

Lytton, R., Uzan, J., & E.G., F. (1993). *Development and Validation of Performance Prediction Models and Specifications for Asphalt and Paving Mixes.* Washington D.C.: Strategic Highway Research Program.

Martini, G. (2019). *Predictive Modelling of Asphalt Concrete Functional Properties Using Multiple Linear Regression and Gradient Boosting.* Delft: TUDelft.

Molenaar, J., & Klarenaar, W. (2008). *Bitumenkarakterisering.* Intron.

Mosegaard, K. S. (2002). Monte Carlo analysis of inverse problems. *Inverse Problems*, R29-R54.

Munera, J., & Ossa, E. (2014). Polymer modified bitumen: Optimization and selection. *Materials and Design*, 91-97.

Nateski, V. (2017). *An overview of the supervised machine learning methods.* Pratizanska: Faculty of Information and Communication Technologies.

Navlani, A. (2018, December 28). *Decision Tree Classification in Python Tutorial*. Opgehaald van Datacamp: https://www.datacamp.com/tutorial/decision-tree-classification-python (Consulted 9-8-22)

nieuwsredactie nu.nl. (2022, 07 17). Asfalt kan dinsdag 50 graden worden: Rijkswaterstaat raadt paraplu aan. *nu.nl*.

Oil Sands Magazine. (2020). Bitumen upgrading explained: From diluted bitumen to synthetic crude. *Oil Sands Magazine*.

Osinanwo, F., Akinsola, J., Awodele, O., Hinmikaiye, J., Olakanmi, O., & Akinjobi, J. (2017). Supervised Machine Learning Algorithms: Classification and Comparison. *International Journal of Computer Trends and Technology*, 128-138.

Pence, B., Fathy, H., & Stein, J. (2011). Recursive maximum likelihood parameter estimation for state space systems using polynomial chaos theory. *Automatica*, 2420-2424.

Poeran, N., Sluer, B., & Telman, J. (2018). Van functioneel verifieren naar functioneel opleveren. *CROW Infradagen 2018* (pp. 1-21). Ede: CROW.

Praveena, M., & Jaiganesh, V. (2017). A Literature Review on Supervised Machine Learning Algorithms and Boosting Process. *International Journal of Computer Applications*, 32-35.

Sengoz, B., & Agar, E. (2007). Effect of asphalt film thickness on the moisture sensitivity characteristics of hot-mix asphalt. *Building and Environment*, 3621-3628.

Shell Bitumen. (2015). *The Shell Bitumen Handbook, 6th edition.* London: ICE Publishing.

Sleuer, B. (2014-2). Functioneel verifiëren en de CROW contractbepalingen 2020. *AsfaltBlij.nl*, 18-19.

Sleuer, B., & Stigter, J. (2014). *Functioneel verifiëren asfaltverhardingen.* Ede: CROW Asfalt Impuls.

Tarantola, A. (2005). *Inverse Problem Theory and Methods for Model Parameter Estimation.* Philadelphia: Society for Industrial and Applied Mathematics.

Underwood, B. S. (2013). Enhancing the durability of asphalt pavements. *Use of models to Enhance the Durability of Asphalt Pavements* (pp. 18-34). Washington: Transportation Research Board.

van de Ven, M., ter Huerne, H., & Voskuilen, J. (2006). Verdichten van asfaltproefstukken in Europa, Wat te doen met (ultra-)dunne deklagen? *CROW Wegbouwkundige Werkdagen.* Ede: CROW.

Van der Poel, C. (1954). A general system describing the vico-elastic properties of bitumens and its relation to routine test data. *Journaul of Applied Chemistry*, 221-236.

Van der Ven, M., & De Jong, E. (2013). Vulstof is meer dan vulling. *Asfaltblij*.

VBW Asfalt, Benelux Bitume. (2006). *Asfaltkunde.* Apeldoorn: KOAC-NPC.

Werkgroep Asfalt Impuls. (2021). *Grip op Bitumen.* Ede: CROW.

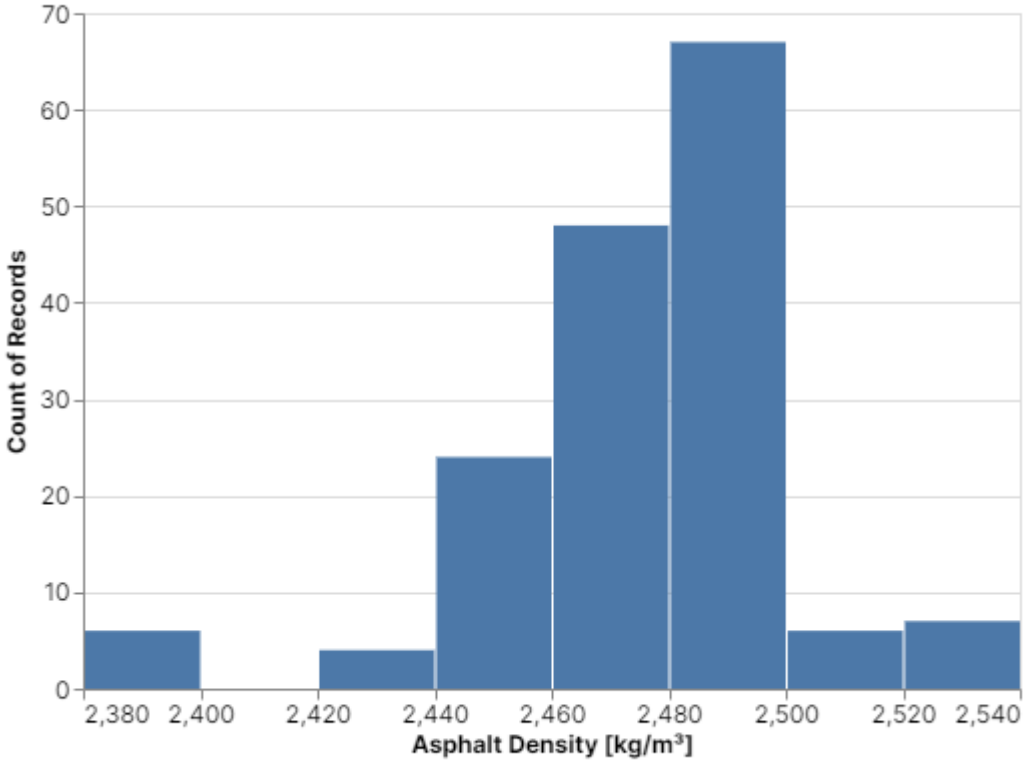
Woldekidan, M. (2011). *Response Modelling of Bitumen, Bituminous Mastic and Mortar.* Delft: TUDelft.

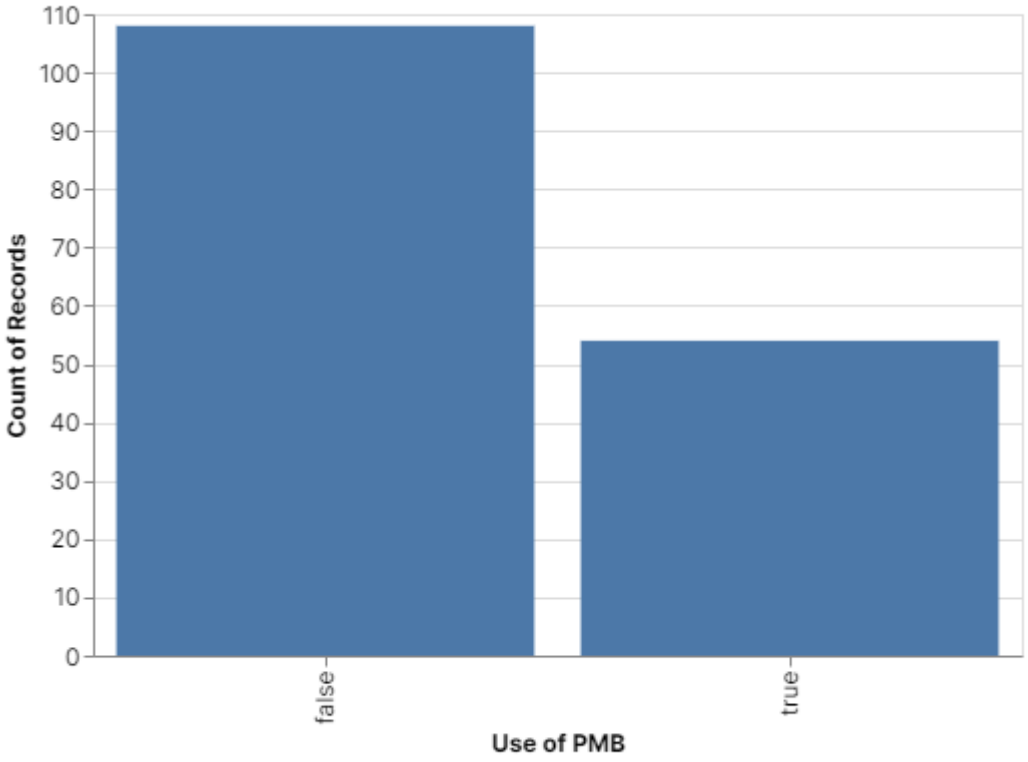
Yilmaz, A., & Sargin, S. (2012). Water effect on deteriorations of asphalt pavements. *TOJSAT*, 1-6.

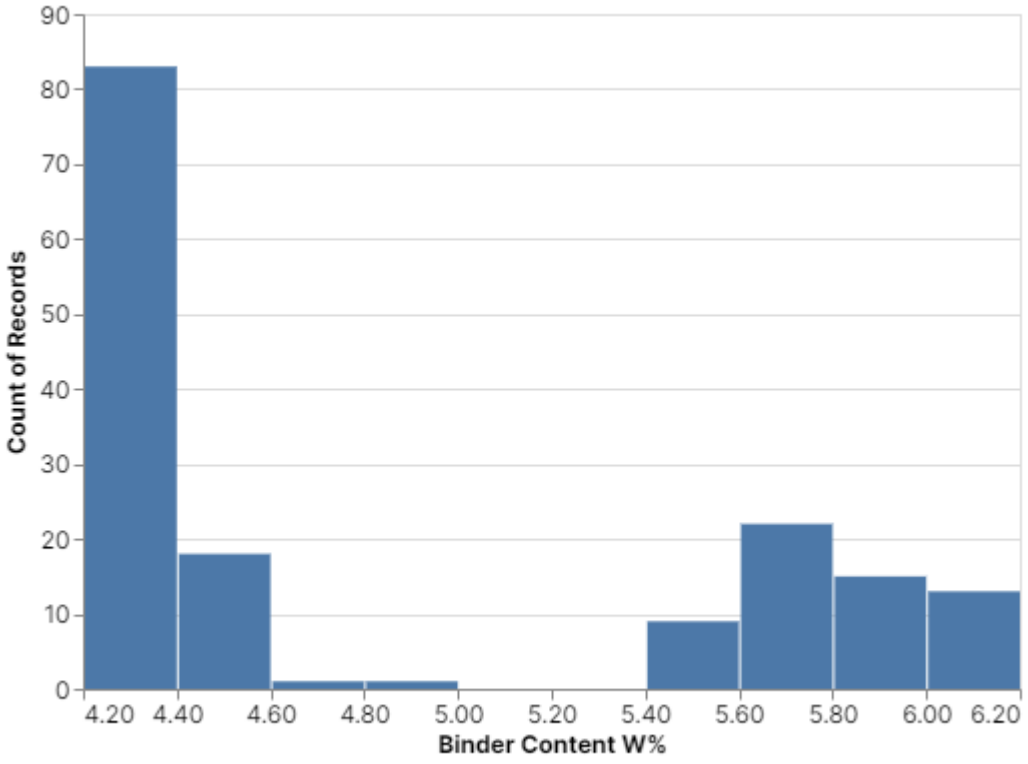
Zuk, P., & Zuk, P. (2022). National energy security or acceleration of transition? Energy policy after the war in Ukraine. *Joule*, 703-712.

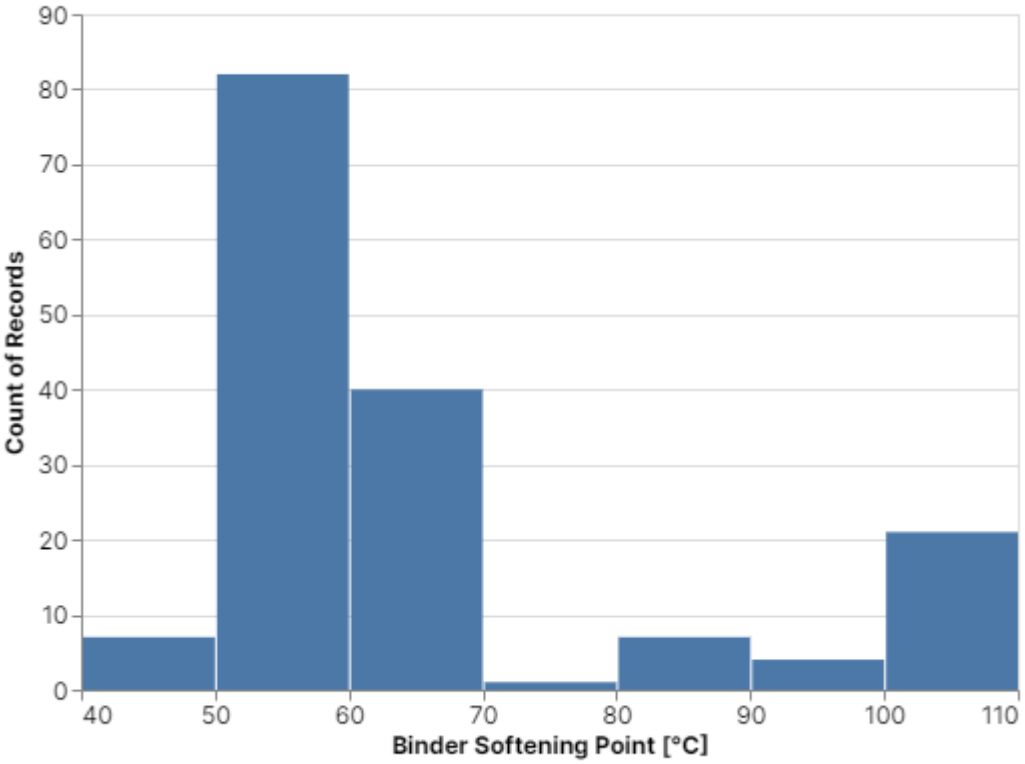
Annex I. Database Distributions

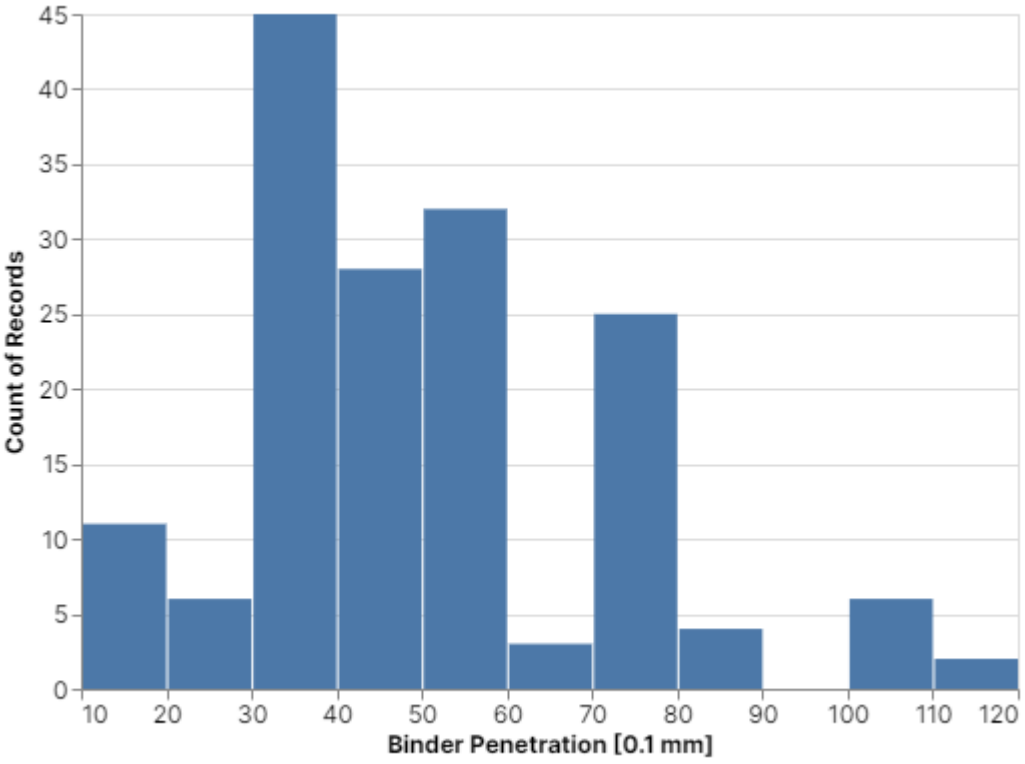
To increase insight in the database used some graphs were made that show the distribution of the different parameters. The graphs are in the same order as Table 4-1 and Table 4-2 .

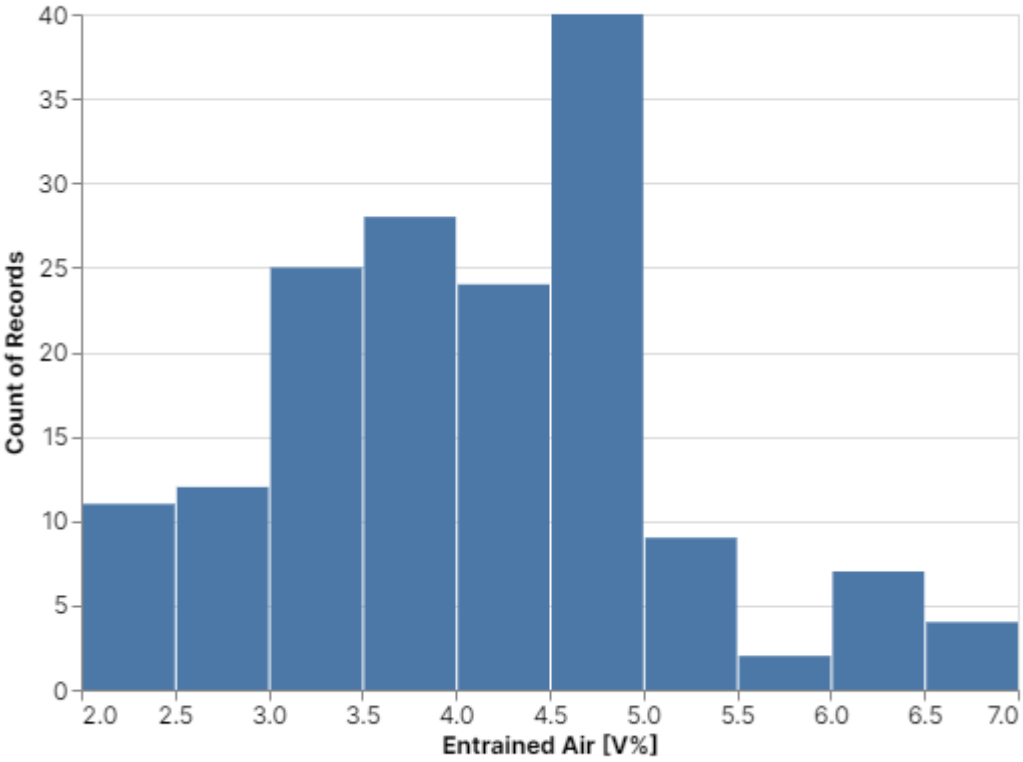


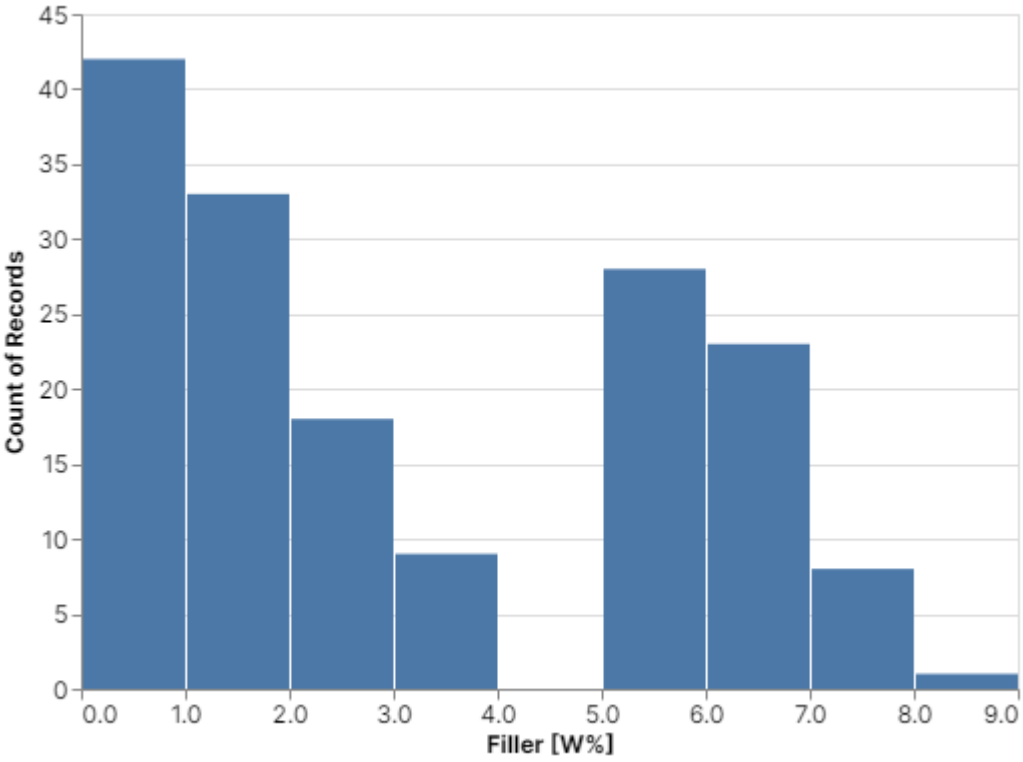

110
100
90
80
70
60
50
40
30
20
10
0
Count of Records
false
true
Use of PMB

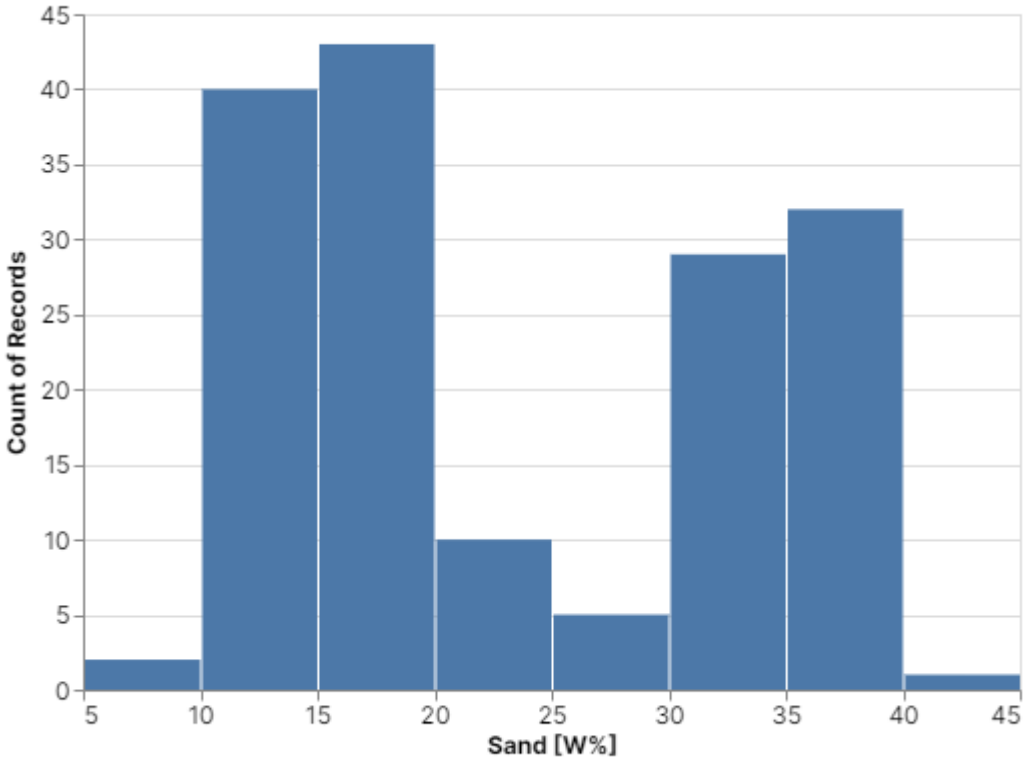

90
80
70
60
50
40
30
20
10
0
Count of Records
4.20
4.40
4.60
4.80
5.00
5.20
5.40
5.60
5.80
6.00
6.20
Binder Content W%

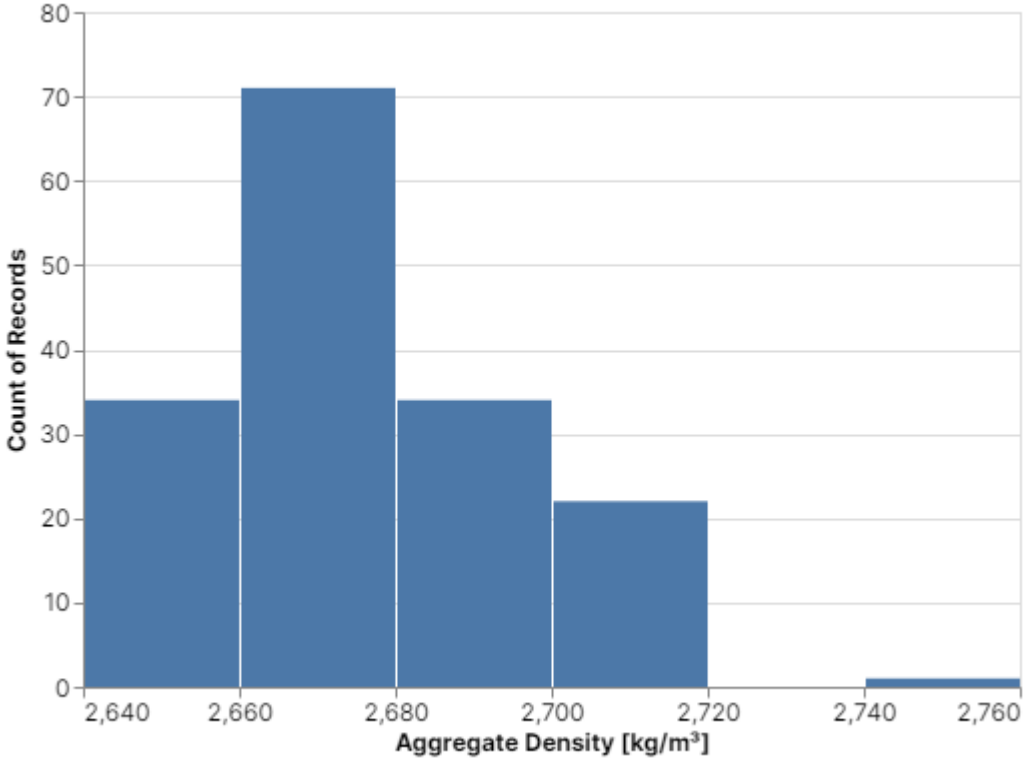

Count of Records
90
80
70
60
50
40
30
20
10
0
40
50
60
70
80
90
100
110
Binder Softening Point [°C]

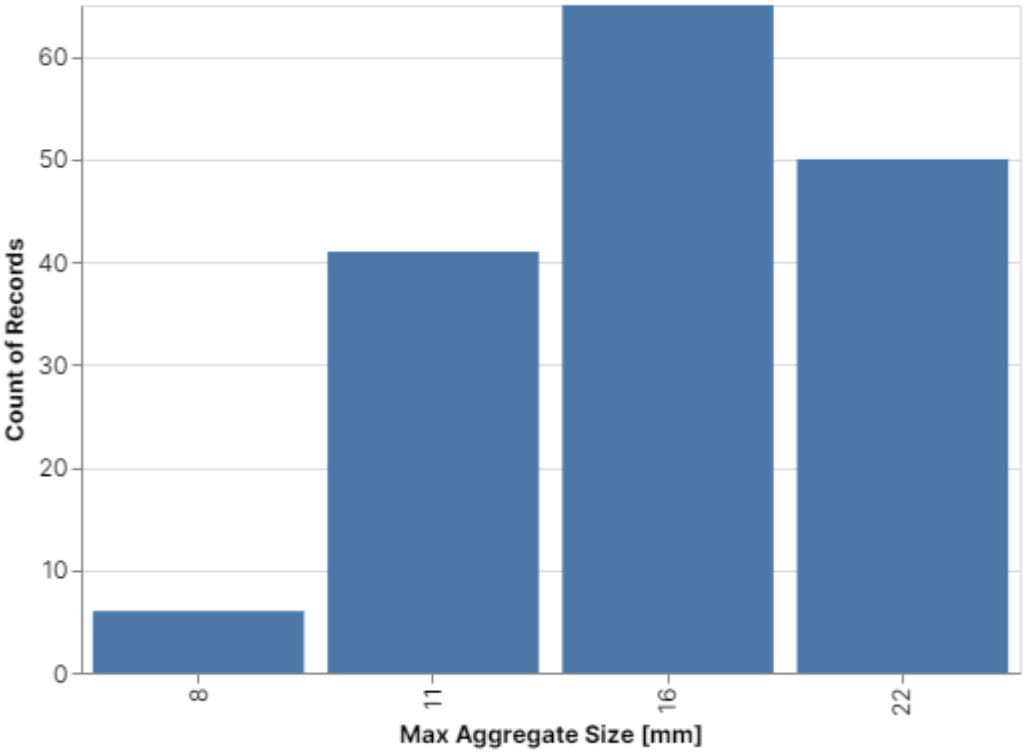

Count of Records
45
40
35
30
25
20
15
10
5
0
10
20
30
40
50
60
70
80
90
100
110
120
Binder Penetration [0.1 mm]

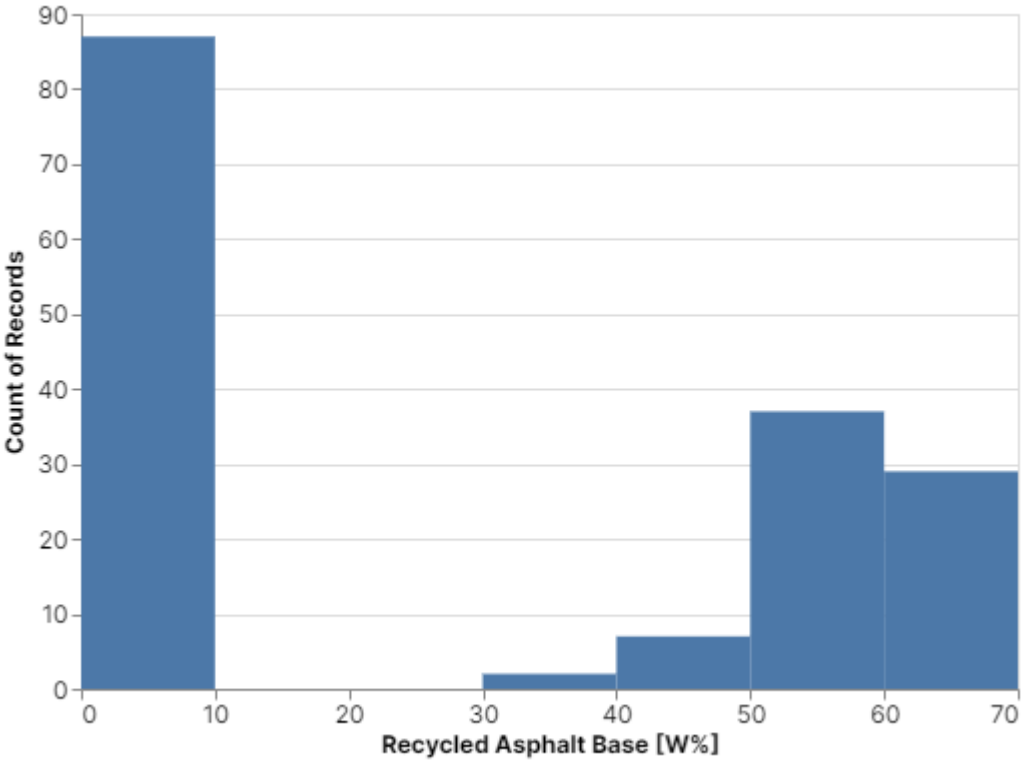

Count of Records
40
35
30
25
20
15
10
5
0
2.0
2.5
3.0
3.5
4.0
4.5
5.0
5.5
6.0
6.5
7.0
Entrained Air [V%]

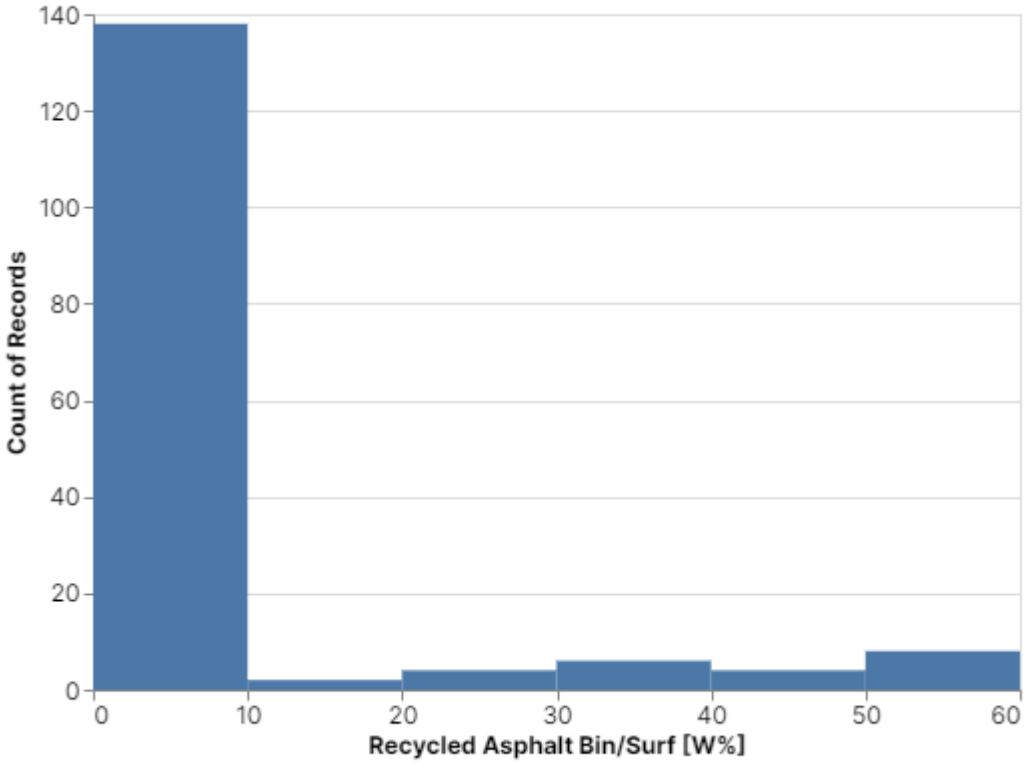

Count of Records
45
40
35
30
25
20
15
10
5
0
0.0
1.0
2.0
3.0
4.0
5.0
6.0
7.0
8.0
9.0
Filler [W%]

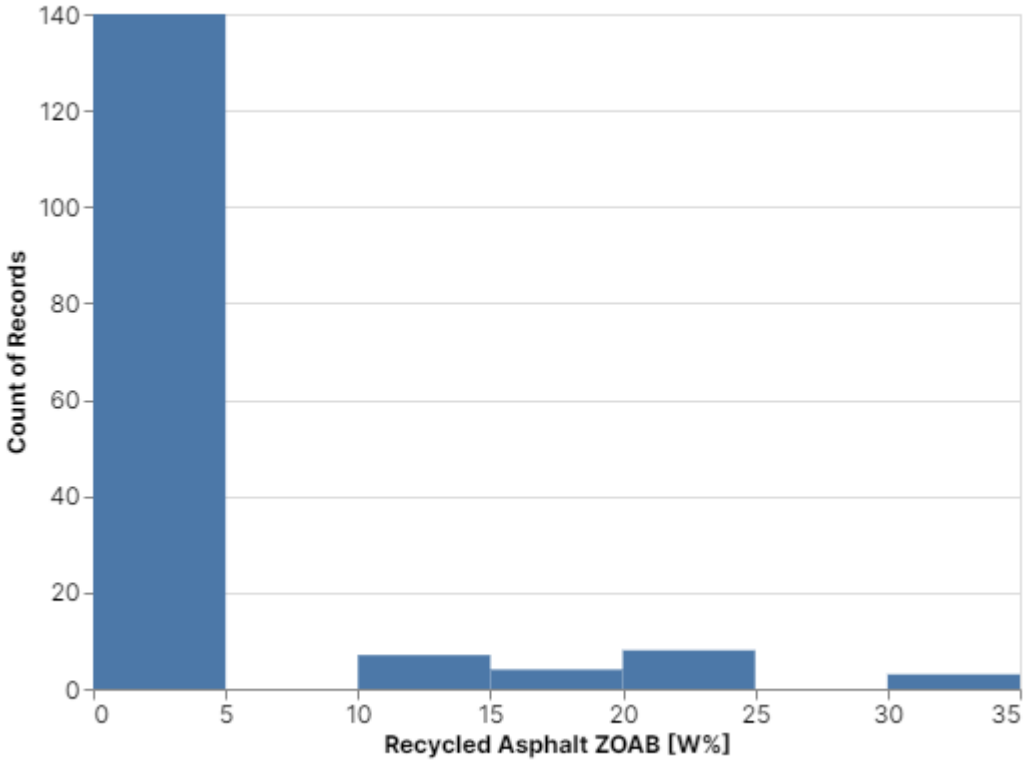

45
40
35
30
25
20
15
10
5
0
Count of Records
5
10
15
20
25
30
35
40
45
Sand [W%]

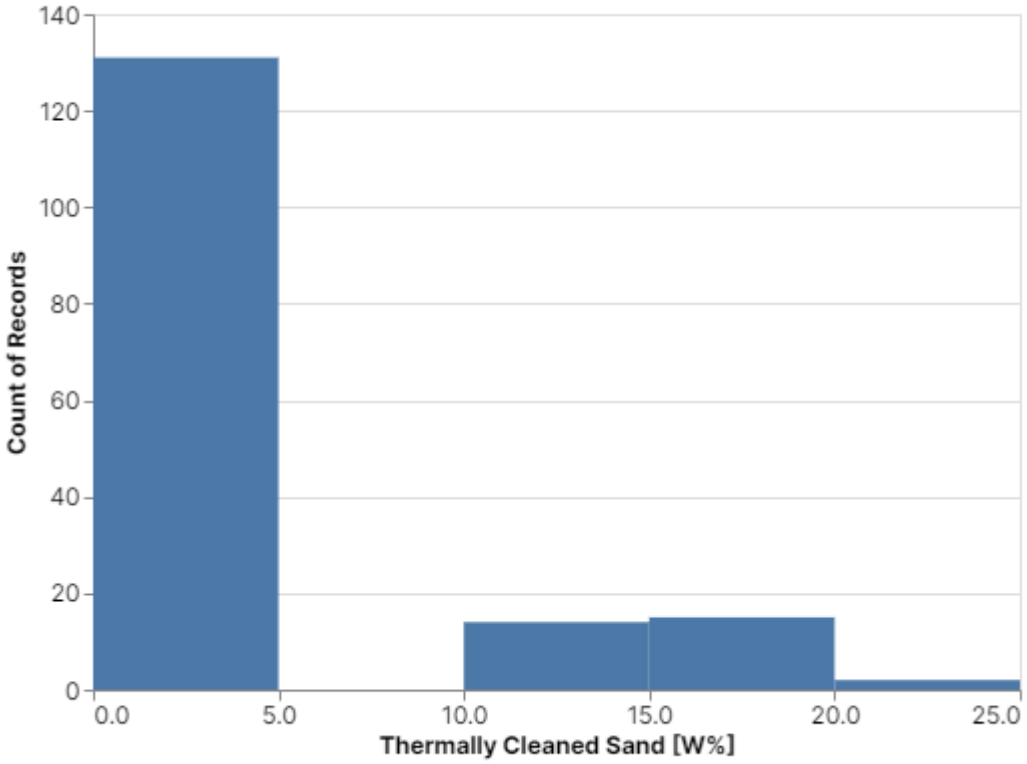

80
70
60
50
40
30
20
10
0
Count of Records
2,640
2,660
2,680
2,700
2,720
2,740
2,760
Aggregate Density [kg/m³]

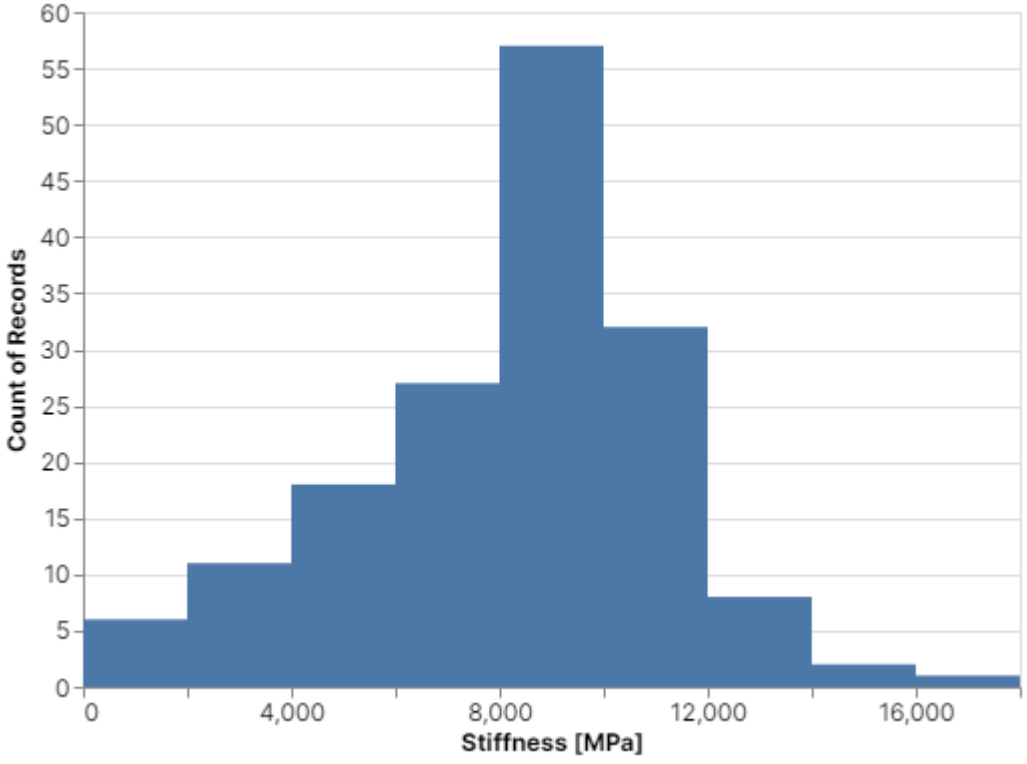

Count of Records
60
50
40
30
20
10
0
8
11
16
22
Max Aggregate Size [mm]

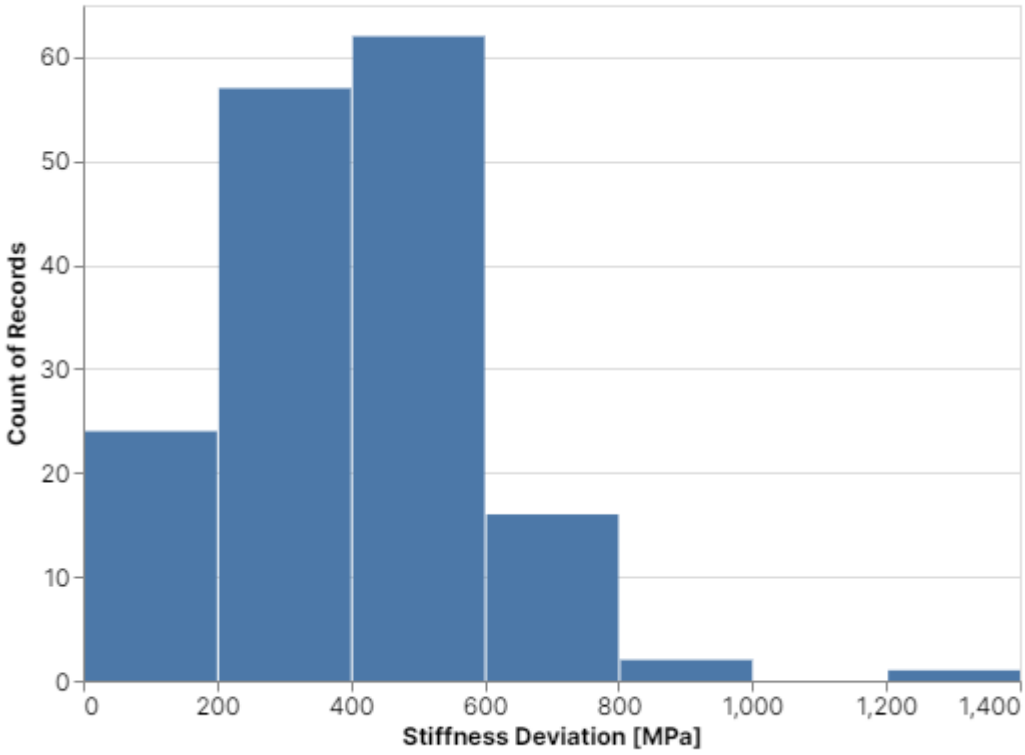

Count of Records
90
80
70
60
50
40
30
20
10
0
0
10
20
30
40
50
60
70
Recycled Asphalt Base [W%]

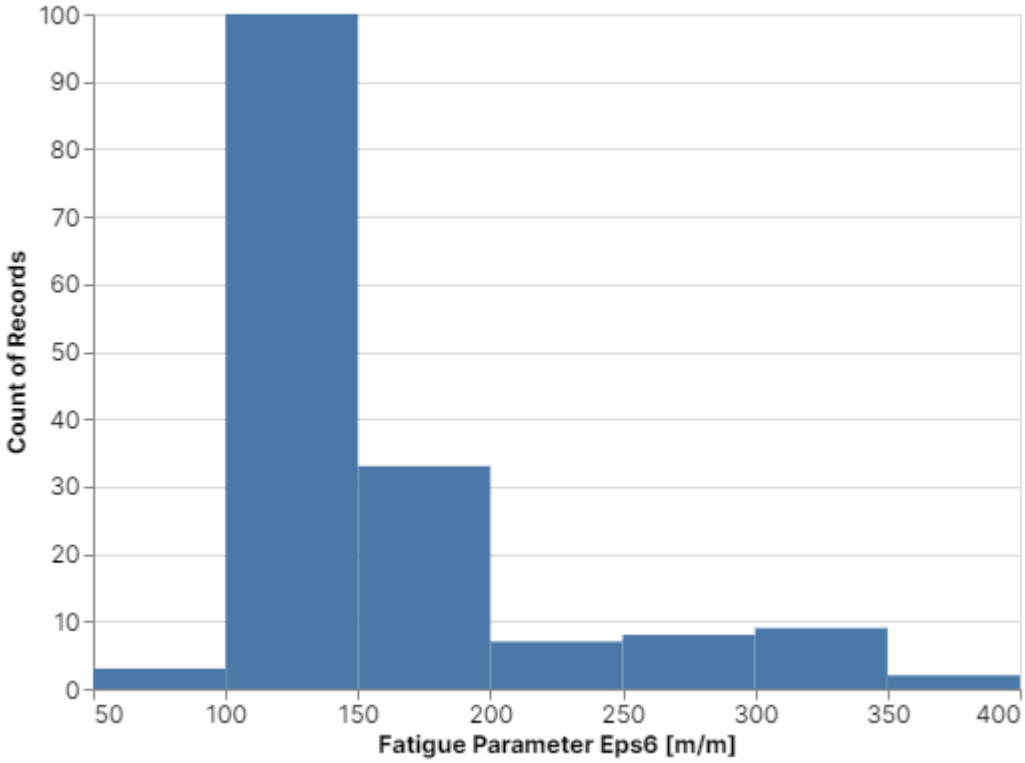

140
120
100
80
60
40
20
0
Count of Records
0
10
20
30
40
50
60
Recycled Asphalt Bin/Surf [W%]

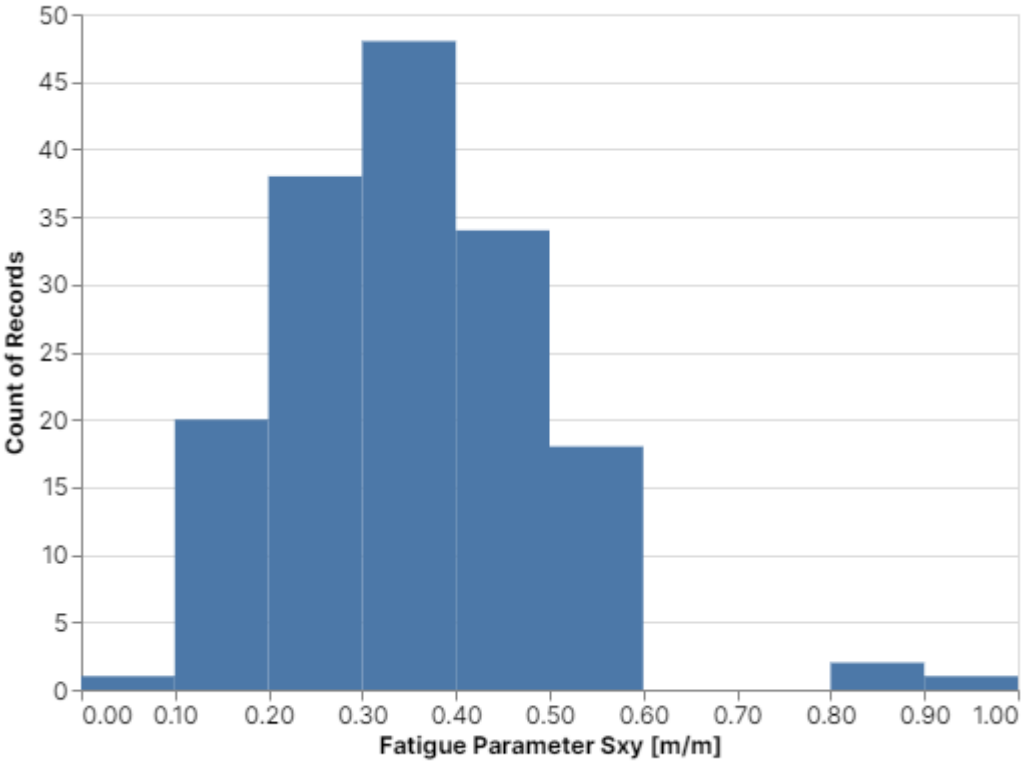

140
120
100
80
60
40
20
0
Count of Records
0
5
10
15
20
25
30
35
Recycled Asphalt ZOAB [W%]

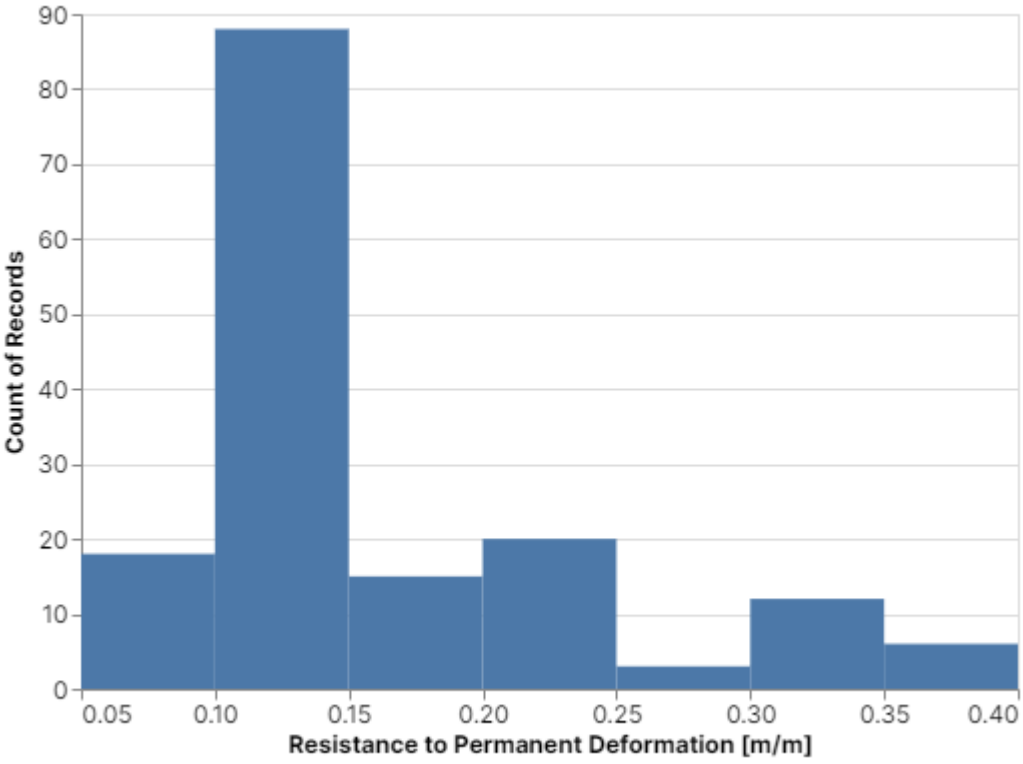

140
120
100
80
60
40
20
0
Count of Records
0.0
5.0
10.0
15.0
20.0
25.0
Thermally Cleaned Sand [W%]

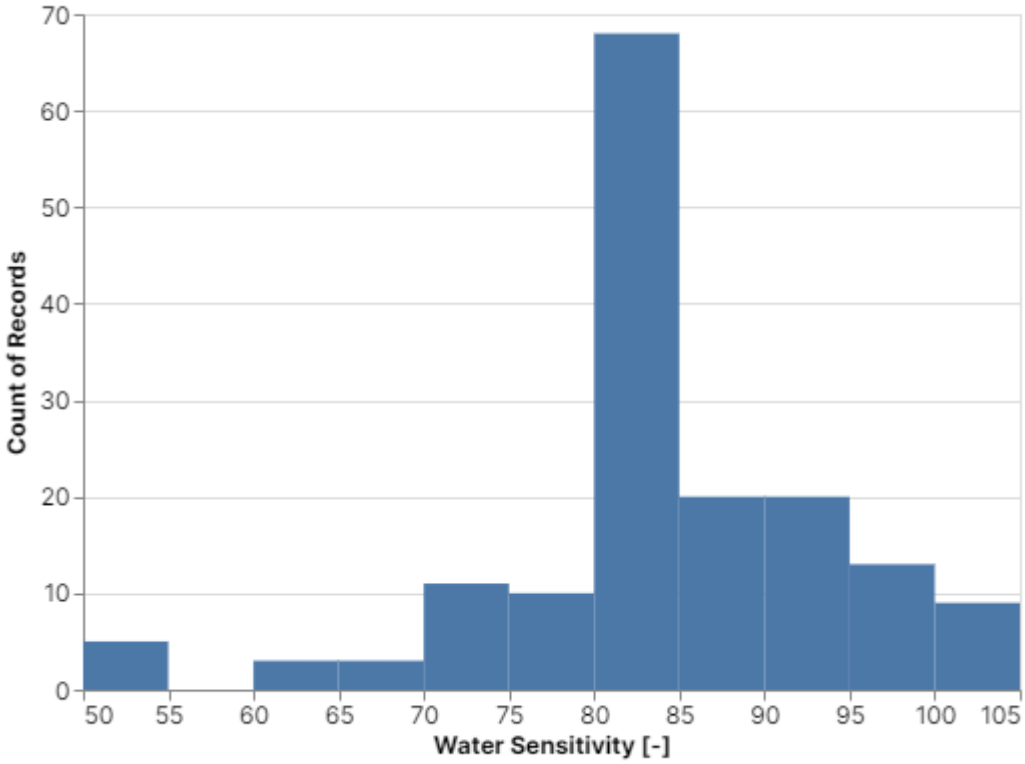

60
55
50
45
40
35
30
25
20
15
10
5
0
Count of Records
0
4,000
8,000
12,000
16,000
Stiffness [MPa]

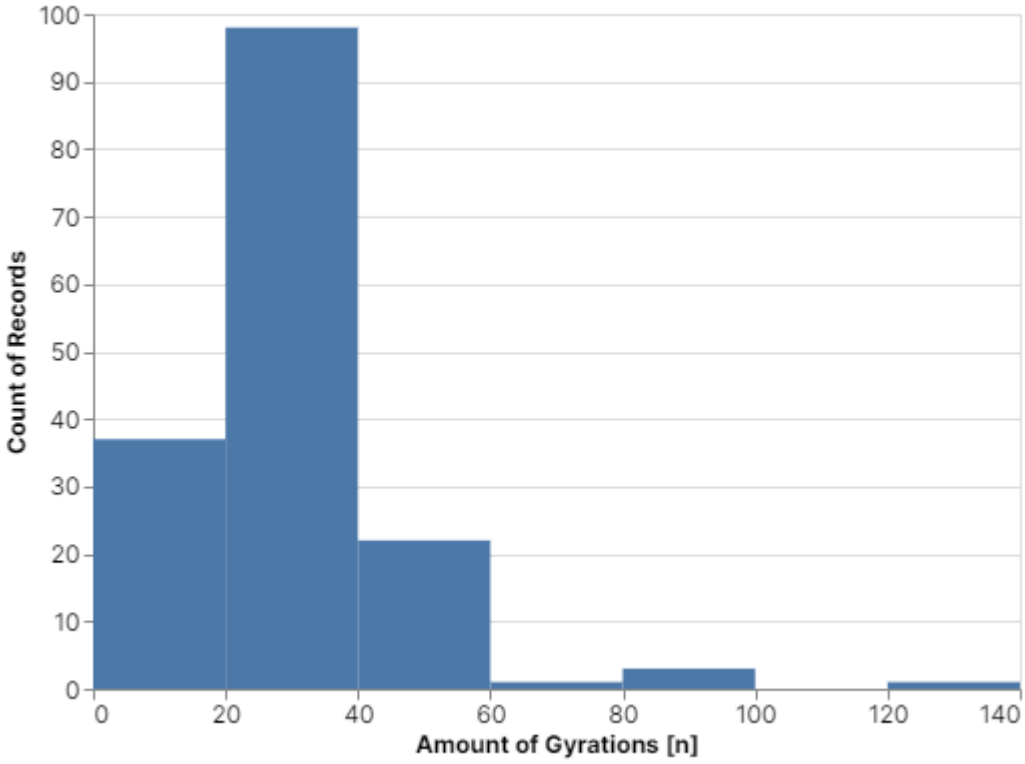

Count of Records
60
50
40
30
20
10
0
0
200
400
600
800
1,000
1,200
1,400
Stiffness Deviation [MPa]

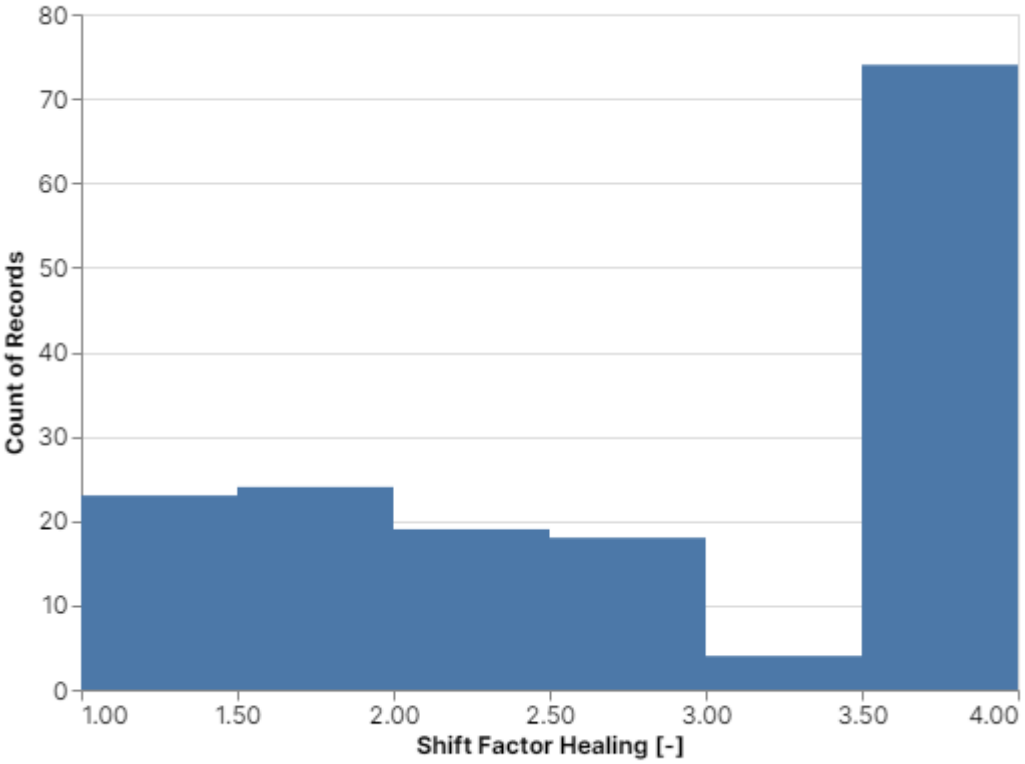

Count of Records
100
90
80
70
60
50
40
30
20
10
0
50
100
150
200
250
300
350
400
Fatigue Parameter Eps6 [m/m]


Count of Records
50
45
40
35
30
25
20
15
10
5
0
0.00
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80
0.90
1.00
Fatigue Parameter Sxy [m/m]


Count of Records
90
80
70
60
50
40
30
20
10
0
0.05
0.10
0.15
0.20
0.25
0.30
0.35
0.40
Resistance to Permanent Deformation [m/m]


Count of Records
70
60
50
40
30
20
10
0
50
55
60
65
70
75
80
85
90
95
100
105
Water Sensitivity [-]


Count of Records
100
90
80
70
60
50
40
30
20
10
0
0
20
40
60
80
100
120
140
Amount of Gyrations [n]


80
70
60
50
40
30
20
10
0
Count of Records
1.00
1.50
2.00
2.50
3.00
3.50
4.00
Shift Factor Healing [-]

Annex II. Python code example

Data driven analysis of common asphalt concrete property prediction methods and a solution to the inverse problem

1. Data processing

1.1 Load in the data

```
import matplotlib.pyplot as plt

def PlotOutputFigure(YTotal, YTrain, YTest, YModelTotal, YModelTrain, YModelTest, ModelNam

    figure, axis = plt.subplots(4, 2, figsize=(10,12), facecolor='w')

    i = 0
    j = 0

    for SingleOutput in YTotal:
        axis[j,i].scatter(YTrain[SingleOutput],YModelTrain[SingleOutput],alpha=0.5,linewid
        axis[j,i].scatter(YTest[SingleOutput],YModelTest[SingleOutput],alpha=0.5,linewidth
        lineStart = YTotal[SingleOutput].min()
        lineEnd = YTotal[SingleOutput].max()
        axis[j,i].plot([lineStart, lineEnd],[lineStart, lineEnd],color = 'b')
        axis[j,i].set_xlim(0, YTotal[SingleOutput].max()*1.1)
        axis[j,i].set_ylim(0, YModelTotal[SingleOutput].max()*1.1)
        axis[j,i].set_xlabel('Measured')
        axis[j,i].set_ylabel(ModelName+'-predicted')
        i += 1
        if i > 1:
            i = 0
            j += 1

    axis[0,0].title.set_text('Stiffness [MPa]')
    axis[0,1].title.set_text('Deviation in stiffness [MPa]')
    axis[1,0].title.set_text('eps6')
    axis[1,1].title.set_text('Sxy')
    axis[2,0].title.set_text('Resistance to permanent deformation')
    axis[2,1].title.set_text('ITSR')
```

```
    axis[3,0].title.set_text('Amount of gyrations')
    axis[3,1].title.set_text('Healing')
    plt.subplots_adjust(left=0.1,
                        bottom=0.1,
                        right=0.9,
                        top=0.9,
                        wspace=0.2,
                        hspace=0.4)

    PlotSaveFileName = 'figures/' + ModelName + '.png'
    plt.savefig(PlotSaveFileName)


def PlotLinksFigure(X, X_Train, X_Test, YModelTotal, YModelTrain, YModelTest, ModelName, x
    XFigureTotal = X[xDesired]
    XFigureTrain = X_Train[xDesired]
    XFigureTest = X_Test[xDesired]
    figure, axis = plt.subplots(7, 2, figsize=(8,21), facecolor='w')

    i = 0
    j = 0

    for SingleOutput in XFigureTotal:
        print()
        axis[j,i].scatter(YModelTrain[yDesired],XFigureTrain[SingleOutput],alpha=0.5,linew
        axis[j,i].scatter(YModelTest[yDesired],XFigureTest[SingleOutput],alpha=0.5,linewid
        # axis[j,i].set_xlim(0, YModelTotal[yDesired].max()*1.1)
        # axis[j,i].set_ylim(0, XFigureTotal[SingleOutput].max()*1.1)
        axis[j,i].set_xlabel(yDesired)
        axis[j,i].set_ylabel(SingleOutput)
        i += 1
        if i > 1:
            i = 0
            j += 1

    PlotSaveFileName = 'figures/' + ModelName + '.png'
    plt.savefig(PlotSaveFileName)
```

```
import os
import pandas as pd
pd.options.mode.chained_assignment = None  # default='warn'
import numpy as np
import import_ipynb
# import ThesisDefinitions as td
import matplotlib.pyplot as plt

!ls /datasets/datasetcsv
DataDir = '/work'

DatafileName = 'Dataset_CSV3.csv'
DataResultsName = 'Results.xlsx'
DatafilePath = os.path.join(DataDir, DatafileName)
DataResultsPath = os.path.join(DataDir, DataResultsName)

DF_Main = pd.read_csv(DatafilePath, encoding='ISO-8859-1', delimiter=';', decimal=',')
```

```
Dataset_CSV.csv
```

1.2 Define input and output

```
SF_Input = [
    'Label',
    'DICHTHEID_MENGSEL_BEPAALD',
    'BerekendBindmiddelgehalte',
    'TemperatuurRingAndBallMengsel',
    'PenMengsel',
    'HOLLE_RUIMTE',
    'DichtheidMineraalAggregaatBerekend',
    '0/0',
    '0/2',
    'WaardeInMM',
    'Onderlaag',
    'Tussen- en deklaag',
    'Zoab',
    'Thermisch gereinigd zand',
    ]

SF_Output = [
    'STIJFHEID_4_PB',
    'StandaardDeviatieStijfheid',
    'ParameterVermoeiingEps6',
    'ParameterVermoeiingSxy',
    'WEERSTAND_TEGEN_PERMANENTE_VERVORMING',
    'WATERGEVOELIGHEID',
    'AANTAL_GYRATIES',
    'PraktijkShiftFactorHealing',
    ]
```

1.3 Preprocessing of the Stiffness and Fatigue measurements

```
import matplotlib.pyplot as plt
from matplotlib.pyplot import boxplot
DF_SF = DF_Main[SF_Input+SF_Output]
DF_SF['Label'] = DF_SF['Label'].replace({'PMB': True, 'PEN': False})
SF = SF_Output + ['Label', 'DICHTHEID_MENGSEL_BEPAALD', 'PenMengsel']
for x in SF:
    DF_SF = DF_SF[pd.to_numeric(DF_SF[x], errors='coerce').notnull()]
DF_SF['Label'] = DF_SF['Label'].astype('boolean')
DF_SF = DF_SF.fillna(0)
```

1.4 Splitting data in training- and testing data

```
from sklearn.model_selection import train_test_split

test_ratio = 0.3 #70/30 split train/test
seed = 101

DF_MLR = DF_SF

DF_MLR = DF_MLR.sample(frac=1, random_state=seed)
nrows = round(DF_MLR.shape[0]*test_ratio)
DF_MLR_test = DF_MLR.iloc[0:nrows, :]
DF_MLR_train = DF_MLR.iloc[nrows:, :]

X = DF_MLR[SF_Input]
Y = DF_MLR[SF_Output]

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=test_ratio, random_sta
```

2. Forward models

```
from sklearn.metrics import r2_score
from scipy.stats import pearsonr

from openpyxl import Workbook
# create a workbook as .xlsx file
def create_workbook(path):
   workbook = Workbook()
   workbook.save(path)
if __name__ == "__main__":
   create_workbook("Results.xlsx")
```

2.1 Multi Linear Regression Analysis

```
from sklearn.linear_model import LinearRegression as MLR

coeff_df = pd.DataFrame(X.columns, columns=[''])
R2_df = pd.DataFrame(['R2 training', 'R2 test', 'R2 total'], columns=[''])
Y_MLR = pd.DataFrame()
Y_MLR_train = pd.DataFrame()
Y_MLR_test = pd.DataFrame()
MLM_PCC_df = pd.DataFrame(columns = SF_Output, index = SF_Input)

MLR_Model = MLR()
MLR_Model.fit(X_train, Y_train)
```

```
MLR_Model.score(X_train, Y_train)

Y_MLR = pd.DataFrame(MLR_Model.predict(X), columns=Y.columns)
Y_MLR_train = pd.DataFrame(MLR_Model.predict(X_train), columns=Y.columns)
Y_MLR_test = pd.DataFrame(MLR_Model.predict(X_test), columns=Y.columns)

coeff_df = pd.DataFrame(MLR_Model.coef_, columns=X.columns, index=Y.columns)

for SingleOutput in Y:
    R2_MLR = r2_score(Y[SingleOutput], Y_MLR[SingleOutput])
    R2_MLR_train = r2_score(Y_train[SingleOutput], Y_MLR_train[SingleOutput])
    R2_MLR_test = r2_score(Y_test[SingleOutput], Y_MLR_test[SingleOutput])
    R2_df[SingleOutput] = [R2_MLR_train, R2_MLR_test, R2_MLR]
    for x in SF_Input:
        MLM_PCC_df[SingleOutput][x] = pearsonr(X_train[x], Y_train[SingleOutput])[0]

with pd.ExcelWriter(DataResultsPath, mode="a", engine="openpyxl", if_sheet_exists="replace
    coeff_df.to_excel(writer, sheet_name='MLR_coeff', startrow = 0, index = False)
    R2_df.to_excel(writer, sheet_name="MLR_R2", startrow=0, index=False)
    MLM_PCC_df.to_excel(writer, sheet_name='MLM_PCC', startrow=0)
```

```
from sklearn.linear_model import Ridge

coeff_df = pd.DataFrame(X.columns, columns=[''])
R2_df = pd.DataFrame(['R2 training', 'R2 test', 'R2 total'], columns=[''])
Y_Ridge = pd.DataFrame()
Y_Ridge_train = pd.DataFrame()
Y_Ridge_test = pd.DataFrame()
Ridge_PCC_df = pd.DataFrame(columns = SF_Output, index = SF_Input)

Ridge_Model = Ridge(alpha=0.5)
Ridge_Model.fit(X_train, Y_train)
Ridge_Model.score(X_train, Y_train)

Y_Ridge = pd.DataFrame(Ridge_Model.predict(X), columns=Y.columns)
Y_Ridge_train = pd.DataFrame(Ridge_Model.predict(X_train), columns=Y.columns)
Y_Ridge_test = pd.DataFrame(Ridge_Model.predict(X_test), columns=Y.columns)

coeff_df = pd.DataFrame(Ridge_Model.coef_, columns=X.columns, index = Y.columns)

for SingleOutput in Y:
    R2_Ridge = r2_score(Y[SingleOutput], Y_Ridge[SingleOutput])
    R2_Ridge_train = r2_score(Y_train[SingleOutput], Y_Ridge_train[SingleOutput])
    R2_Ridge_test = r2_score(Y_test[SingleOutput], Y_Ridge_test[SingleOutput])
    R2_df[SingleOutput] = [R2_Ridge_train, R2_Ridge_test, R2_Ridge]
    for x in SF_Input:
        Ridge_PCC_df[SingleOutput][x] = pearsonr(X_train[x], Y_train[SingleOutput])[0]

with pd.ExcelWriter(DataResultsPath, mode="a", engine="openpyxl", if_sheet_exists="replace
    coeff_df.to_excel(writer, sheet_name='Ridge_coeff', startrow = 0, index = False)
    R2_df.to_excel(writer, sheet_name='Ridge_R2', startrow=0, index=False)
    Ridge_PCC_df.to_excel(writer, sheet_name='Ridge_PCC', startrow=0)
```

2.2 Decision trees

```
from sklearn import tree
# coeff_df = pd.DataFrame(X.columns, columns=[''])
n = len(SF_Output)
R2_df = pd.DataFrame(['R2 training', 'R2 test', 'R2 total'], columns=[''])
Y_DT = pd.DataFrame()
Y_DT_train = pd.DataFrame()
Y_DT_test = pd.DataFrame()

# Create a decision tree Classifier or regressor?
DT_Model = tree.DecisionTreeRegressor(max_depth = 8, min_samples_leaf=n)
DT_FeatureImportance_df = pd.DataFrame(columns = SF_Output, index = SF_Input)


for SingleOutput in Y:
    y = Y[SingleOutput]
    y_train = Y_train[SingleOutput]
    y_test = Y_test[SingleOutput]

    DT_Model.fit(X, y)
    DT_Model.score(X_train, y_train)

    Y_DT[SingleOutput] = DT_Model.predict(X)
    Y_DT_train[SingleOutput] = DT_Model.predict(X_train)
    Y_DT_test[SingleOutput] = DT_Model.predict(X_test)

    R2_DT = r2_score(Y[SingleOutput], Y_DT[SingleOutput])
    R2_DT_train = r2_score(Y_train[SingleOutput], Y_DT_train[SingleOutput])
    R2_DT_test = r2_score(Y_test[SingleOutput], Y_DT_test[SingleOutput])
    R2_df[SingleOutput] = [R2_DT_train, R2_DT_test, R2_DT]

    DT_FeatureImportance_df[SingleOutput] = DT_Model.feature_importances_


with pd.ExcelWriter(DataResultsPath, mode="a", engine="openpyxl", if_sheet_exists="replace
    R2_df.to_excel(writer, sheet_name="DT_R2", startrow=0, index=False)
    DT_FeatureImportance_df.to_excel(writer, sheet_name="DT_FI", startrow=0, index=True)
```

2.3 Gradient Boosted Decision Tree

```
from catboost import *
import pickle

# n = len(SF_Output)
R2_df = pd.DataFrame(['R2 training', 'R2 test', 'R2 total'], columns=[''])
Y_GB = pd.DataFrame()
Y_GB_train = pd.DataFrame()
Y_GB_test = pd.DataFrame()
```

```
X['Label'] = X['Label'].astype('int')
X_train['Label'] = X_train['Label'].astype('int')
X_test['Label'] = X_test['Label'].astype('int')

GB_Model_per = CatBoostRegressor()
GB_FeatureImportance_df = pd.DataFrame(columns=Y.columns, index=X.columns)

for SingleOutput in Y:
    y = Y[SingleOutput]
    y_train = Y_train[SingleOutput]
    y_test = Y_test[SingleOutput]

    cat_column_names = ['Label', 'WaardeInMM']

    train_pool = Pool(X_train, y_train, cat_features=cat_column_names)
    test_pool = Pool(X_test, y_test, cat_features=cat_column_names)

    GB_Model_per.fit(train_pool, eval_set=test_pool, use_best_model=True)
    GB_Model_per.score(X_train, y_train)

    Y_GB[SingleOutput] = GB_Model_per.predict(X)
    Y_GB_train[SingleOutput] = GB_Model_per.predict(X_train)
    Y_GB_test[SingleOutput] = GB_Model_per.predict(X_test)

    R2_GB = r2_score(Y[SingleOutput], Y_GB[SingleOutput])
    R2_GB_train = r2_score(Y_train[SingleOutput], Y_GB_train[SingleOutput])
    R2_GB_test = r2_score(Y_test[SingleOutput], Y_GB_test[SingleOutput])
    R2_df[SingleOutput] = [R2_GB_train, R2_GB_test, R2_GB]
    print(SingleOutput)
    filename = 'models/GB_Model/'+SingleOutput+'.sav'
    pickle.dump(GB_Model_per, open(filename, 'wb'))

    GB_FeatureImportance_df[SingleOutput] = GB_Model_per.get_feature_importance()

with pd.ExcelWriter(DataResultsPath, mode="a", engine="openpyxl", if_sheet_exists="replace
    R2_df.to_excel(writer, sheet_name="GB_R2", startrow=0, index=False)
    GB_FeatureImportance_df.to_excel(writer, sheet_name="GB_FI", startrow=0, index=True)

X['Label'] = X['Label'].astype('boolean')
X_train['Label'] = X_train['Label'].astype('boolean')
X_test['Label'] = X_test['Label'].astype('boolean')
```

2.4 Downloading trained models

```
import pickle

MLR_Model.fit(X, Y)
DT_Model.fit(X, Y)

filename = 'models/MLR_Model.sav'
pickle.dump(MLR_Model, open(filename, 'wb'))
filename = 'models/DT_Model.sav'
pickle.dump(DT_Model, open(filename, 'wb'))
```

```
from lazypredict.Supervised import LazyClassifier, LazyRegressor

create_workbook("regressors.xlsx")
create_workbook("classifiers.xlsx")

for SingleOutput in Y:
    y = Y[SingleOutput]
    y_train = Y_train[SingleOutput]
    y_test = Y_test[SingleOutput]

    reg = LazyRegressor(verbose=0, ignore_warnings=True,custom_metric=None)
    models, predictions=reg.fit(X_train, X_test, y_train, y_test)

    # single_result = pd.DataFrame(models, columns=["Model", R-Squared", "Adjusted R-Squar
    single_result = pd.DataFrame(models)

    with pd.ExcelWriter("regressors.xlsx", mode="a", engine="openpyxl", if_sheet_exists="o
        single_result.to_excel(writer, sheet_name=SingleOutput, index=True)
```

3 Inverse problem solution

```
from sklearn.model_selection import train_test_split

test_ratio = 0.3 #70/30 split train/test
seed = 101

DF_Inv = DF_SF

DF_Inv = DF_Inv.sample(frac=1, random_state=seed)
nrows = round(DF_Inv.shape[0]*test_ratio)
DF_Inv_test = DF_Inv.iloc[0:nrows, :]
DF_Inv_train = DF_Inv.iloc[nrows:, :]

X = DF_Inv[SF_Output]
Y = DF_Inv[SF_Input]

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=test_ratio, random_sta
```

3.1 Inverse solution with Gradient Boosted Decision Tree machine learning

```
from catboost import *
from sklearn.metrics import r2_score

n = len(SF_Output)
R2_df = pd.DataFrame(['R2 total','R2 training', 'R2 test'], columns=[''])
Y_IGB = pd.DataFrame()
```

```
Y_IGB_train = pd.DataFrame()
Y_IGB_test = pd.DataFrame()

Y['Label'] = Y['Label'].astype('int')
Y_train['Label'] = Y_train['Label'].astype('int')
Y_test['Label'] = Y_test['Label'].astype('int')

IGB_Model_Regr = CatBoostRegressor()
IGB_Model_Class = CatBoostClassifier()
cat_column_names = ['Label', 'WaardeInMM']
IGB_FeatureImportance_df = pd.DataFrame(columns = SF_Input, index = SF_Output)

for SingleInput in Y:
    y = Y[SingleInput]
    y_train = Y_train[SingleInput]
    y_test = Y_test[SingleInput]

    train_pool = Pool(X_train, y_train)
    test_pool = Pool(X_test, y_test)

    if SingleInput in cat_column_names:
        IGB_Model_Class.fit(train_pool, eval_set=test_pool, use_best_model=True)
        Y_IGB[SingleInput] = IGB_Model_Class.predict(X)
        Y_IGB_train[SingleInput] = IGB_Model_Class.predict(X_train)
        Y_IGB_test[SingleInput] = IGB_Model_Class.predict(X_test)
        IGB_FeatureImportance_df[SingleInput] = IGB_Model_Class.get_feature_importance()
        print('Classed')
    else:
        IGB_Model_Regr.fit(train_pool, eval_set=test_pool, use_best_model=True)
        IGB_Model_Regr.score(X_train, y_train)
        Y_IGB[SingleInput] = IGB_Model_Regr.predict(X)
        Y_IGB_train[SingleInput] = IGB_Model_Regr.predict(X_train)
        Y_IGB_test[SingleInput] = IGB_Model_Regr.predict(X_test)
        IGB_FeatureImportance_df[SingleInput] = IGB_Model_Regr.get_feature_importance()
        print('Regged')



    R2_IGB = r2_score(Y[SingleInput], Y_IGB[SingleInput])
    R2_IGB_train = r2_score(Y_train[SingleInput], Y_IGB_train[SingleInput])
    R2_IGB_test = r2_score(Y_test[SingleInput], Y_IGB_test[SingleInput])
    R2_df[SingleInput] = [R2_IGB_train, R2_IGB_test, R2_IGB]

with pd.ExcelWriter(DataResultsPath, mode="a", engine="openpyxl", if_sheet_exists="replace
    R2_df.to_excel(writer, sheet_name="IGB_R2", startrow=0, index=False)
    IGB_FeatureImportance_df.to_excel(writer, sheet_name="IGB_FI", startrow=0, index=True)

Y['Label'] = Y['Label'].astype('boolean')
Y_train['Label'] = Y_train['Label'].astype('boolean')
Y_test['Label'] = Y_test['Label'].astype('boolean')
```

3.2 Practical application example

```
from sklearn.model_selection import train_test_split
```

```
Pr_Input = [
    'HOLLE_RUIMTE',
    'WATERGEVOELIGHEID',
    'STIJFHEID_4_PB',
    'WEERSTAND_TEGEN_PERMANENTE_VERVORMING',
    'ParameterVermoeiingEps6',
    ]

Pr_Output = [
    'BerekendBindmiddelgehalte',
    '0/0',
    '0/2',
    'PenMengsel',
    ]

test_ratio = 0.3 #70/30 split train/test
seed = 101

DF_Pr = DF_SF

DF_Pr = DF_Pr.sample(frac=1, random_state=seed)
nrows = round(DF_Pr.shape[0]*test_ratio)
DF_Pr_test = DF_Pr.iloc[0:nrows, :]
DF_Pr_train = DF_Pr.iloc[nrows:, :]

X = DF_Pr[Pr_Input]
Y = DF_Pr[Pr_Output]

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=test_ratio, random_sta
```

```
from catboost import *
from sklearn.metrics import r2_score
import pickle

n = len(Pr_Output)
R2_df = pd.DataFrame(['R2 total','R2 training', 'R2 test'], columns=[''])
Y_PrGB = pd.DataFrame()
Y_PrGB_train = pd.DataFrame()
Y_PrGB_test = pd.DataFrame()

PrGB_Model_Regr = CatBoostRegressor()
cat_column_names = []
PrGB_FeatureImportance_df = pd.DataFrame(columns = Pr_Output, index = Pr_Input)

for SingleInput in Y:
    y = Y[SingleInput]
    y_train = Y_train[SingleInput]
    y_test = Y_test[SingleInput]

    train_pool = Pool(X_train, y_train)
    test_pool = Pool(X_test, y_test)

    PrGB_Model_Regr.fit(train_pool, eval_set=test_pool, use_best_model=True)
    PrGB_Model_Regr.score(X_train, y_train)
    Y_PrGB[SingleInput] = PrGB_Model_Regr.predict(X)
    Y_PrGB_train[SingleInput] = PrGB_Model_Regr.predict(X_train)
```

```
    Y_PrGB_test[SingleInput] = PrGB_Model_Regr.predict(X_test)
    PrGB_FeatureImportance_df[SingleInput] = PrGB_Model_Regr.get_feature_importance()

    R2_PrGB = r2_score(Y[SingleInput], Y_PrGB[SingleInput])
    R2_PrGB_train = r2_score(Y_train[SingleInput], Y_PrGB_train[SingleInput])
    R2_PrGB_test = r2_score(Y_test[SingleInput], Y_PrGB_test[SingleInput])
    R2_df[SingleInput] = [R2_PrGB_train, R2_PrGB_test, R2_PrGB]

    filename = 'models/Pr/Model_'+str(SingleInput.replace('/', ''))+'.sav'
    pickle.dump(PrGB_Model_Regr, open(filename, 'wb'))

with pd.ExcelWriter(DataResultsPath, mode="a", engine="openpyxl", if_sheet_exists="replace
    R2_df.to_excel(writer, sheet_name="Pr_R2", startrow=0, index=False)
    PrGB_FeatureImportance_df.to_excel(writer, sheet_name="Pr_FI", startrow=0, index=True)
```

```
from numpy import arange

A = arange(2,7.1,0.1)
B = [70]
C = arange(4500,11500,500)
D = [1.4]
E = [100]

X_predict = [A, B, C, D, E]

total_length = len(A)*len(B)*len(C)*len(D)*len(E)

X_predict=[]
for a in A:
    for b in B:
        for c in C:
            for d in D:
                for e in E:
                    X_predict.append([a, b, c, d, e])

Y_predict = pd.DataFrame(columns=Pr_Output, index=range(0,total_length,1))
for SingleInput in Y:
    y = Y[SingleInput]
    filename = 'models/Pr/Model_'+str(SingleInput.replace('/', ''))+'.sav'
    PrGB_model = pickle.load(open(filename, 'rb'))
    i=0
    for x_predict in X_predict:
        y_predict = PrGB_model.predict(x_predict)
        Y_predict[SingleInput][i] = y_predict
        i+=1

Predictions = pd.concat([pd.DataFrame(data=X_predict, columns=Pr_Input), Y_predict], axis=

with pd.ExcelWriter(DataResultsPath, mode="a", engine="openpyxl", if_sheet_exists="replace
    Predictions.to_excel(writer, sheet_name='Pr_Results', startrow = 0, index = False)

Predictions = Predictions.rename(columns={'STIJFHEID_4_PB': 'Stiffness', 'WATERGEVOELIGHEI
```